Odyssey 2018 The Speaker and Language Recognition Workshop
26-29 June 2018, Les Sables d'Olonne, France

Incremental On-Line Clustering of Speakers' Short Segments

Ruth Aloni-Lavi, Irit Opher, Itshak Lapidot,

Afeka Tel-Aviv College of Engineering, ACLP, Israel

rutil@afeka.ac.il, irito@afeka.ac.il, itshakl@afeka.ac.il

Abstract

This paper deals with clustering of speakers' short segments, in a scenario where additional segments continue to arrive and should be constantly clustered together with previous segments that were already clustered. In realistic applications, it is not possible to cluster all segments every time a new segment arrives. Hence, incremental clustering is applied in an on-line mode. New segments can either belong to existing speakers, therefore, have to be assigned to one of the existing clusters, or they could belong to new speakers and thus new clusters should be formed. In this work we show that if there are enough segments per speaker in the off-line initial clustering process, it constitutes a good starting point for the incremental on-line clustering. In this case, incremental on-line clustering can be successfully applied based on the previously proposed mean-shift clustering algorithm with PLDA score as a similarity measure and with k-nearest neighbors (kNN) neighborhood selection.

Index Terms: Speaker clustering, mean shift clustering, Probabilistic Linear Discriminant Analysis score (PLDA), k-Nearest Neighbors (kNN), i-vectors, short segments, incremental clustering.

1. Introduction

Speaker clustering is an unsupervised task of identifying which segments from a set of speech segments belong to the same speaker. It can be an inherent part in speaker diarization task [1], or it can be also a stand alone problem [2]. In this work the segments are well defined by a *push to talk* (PTT) button. A realistic scenario is, for example, correspondence that may arrive from several communication devices, each transmitting their part using PTT mechanism. The on going transmission can arrive from different places and sometimes via different channels. In the most general case, a person might switch their transmition device, or it could happen that the same device can be used by more than one person. Hence, the importance of clustering the segments soley based on the speakers' voices is clear, as other means of tracking a speaker do not necessarily provide accurate and reliable information. The segments we are dealing with are usually short, starting with less than 1sec and up to several seconds. This scenario is similar to the works in [3] that clustered the segments for diarization purpose only that the segments' lengths are defined by a *voice activity detection* (VAD) instead of the PTT mechanism. In our case, after an initial period of data collection, all the segments need to be clustered. However, since data continue to arrive, every new segment should also be assigned either to an existing cluster (that should represent a single speaker), or assigned to a new speaker and create a new cluster. One possibility is to cluster all data each time a new segment arrives, or once a certain number of new segments is accumulated. However, this approach proves to be time consuming, as we are dealing with several dozens of speakers with thousands of segments. We suggest a different approach and apply on-line incremental clustering, meaning we do not cluster all data again and again. It is important to differentiate between our scenario and diarization problems where we work on the whole conversation at a single process and do not spread over time, as we do in the clustering case. In addition, the conversation is taken as a single recording and not separated into segments, i.e. in diarization problems we have to apply a speaker change detection algorithm. The number of speakers is also quite different - in diarization problems we are usually limited to a few speakers in telephony conversations (e.g. 2 or 3), up to about 12 for meetings or up to about 25 for TV shows while in the our clustering scenario it can go up to 60 and even more. Another important distinction is that in diarization tasks the number of participants in many cases may be known in advance, for example in two sided telephony conversations or in professional meetings, while in our clustering scenario the number od speaker is never known in advance, and keeps changing over time as new speakers may be added to the system. Such clustering application can be relevant in taxi station which collect all the speech traffic of all the taxi drivers or homeland security challenges.

Different diarization approaches employ a clustering step as in the mean-shift based [3], *variational Bayesian expectation maximization Gaussian Mixture model* (VBEM-GMM) [4] and a very common *Bayesian information criterion* (BIC) [5]. However, the last two are based on having all the segments available for a single clustering operation, while using mean-shift allows us to add new segments in a natural way with only some degradation in performance.

10.21437/Odyssey.2018-17

There are several works that focus on speaker diarization that address or are relevant to on-line diarization. In [6] and [7] we assumed diarization of two speakers in telephone conversation; In [8–10] the number of speakers is not required in advance. All the methods that do not require knowing the number of speakers in advance, can also be applied to the described clustering problem. However, as we have shown in the past, [2, 11, 12], mean-shift based algorithm with PLDA as a similarity measure performs well for the off-line task, so we decided to continue with this approach and show an easy way to extend it to the described problem of incremental on-line clustering. That way we can explore the limitations of the mean-shift clustering under the new conditions, and compare the on-line incremental clustering performance with the off-line performance.

In this work we investigate such a clustering task where some data was already clustered and new data continues to arrive and needs to be clustered. The new segments may arrive from two "classes": whether a new segment belongs to one of the existing speakers and has to be assigned to the correct speaker cluster; or it is a new speaker's segment and hence a new cluster has to be created. We will show, empirically, that for the task of short speech segments where clustering is based on the mean-shift algorithm, it is possible to perform incremental on-line clustering of the new segments with reasonable degradation in clustering performance, in comparison with the off-line system.

In [2, 11, 12], we presented a short segments clustering method, based on the mean-shift clustering algorithm with *probabilistic linear discriminant analysis* (PLDA). It was tested on segments with average duration of 2.5 seconds, and the number of speakers varied between 3 to 60. It was also shown that PLDA outperforms cosine distance as the similarity measure of the mean-shift algorithm. In addition, we showed that using adaptive bandwidth based on *k-nearest neighbors* (kNN) outperforms the traditional fixed bandwidth approach for neighbors selections, when calculating the new mean. In the current work, we use this off-line clustering system as the baseline for incremental on-line clustering.

The paper is organized as follows: Section 2 presents the standard mean-shift algorithm and its modifications; In section 3 the incremental clustering algorithm is presented; Experiments and results are described in Section 4 and the conclusions are presented in Section 5.

2. Mean-shift clustering algorithm

The mean shift Algorithm is a non-parametric iterative algorithm. It estimates the *probability density function* (*pdf*) of a random variable [13]. The algorithm is inspired by the Parzen window approach to non-parametric density estimation. The algorithm does not require any prior knowledge regarding the number of clusters, and has no assumptions regarding the shape of the clusters. The dense regions in the feature space correspond to local maxima or to the density function modes. As such, for each data point, in order to reach the local maximum of the *pdf*, we perform gradient ascent on the estimated local density, until convergence is reached. Each stationary point represents a mode of the density function. Data points that are associated with the same stationary point are assigned to the same cluster.

2.1. Standard mean shift algorithm

The gradient of the density function $f(\phi)$ is required in order to find the above mentioned modes. Following the mathematical formulation in [3, 13, 14], the mean shift vector $m_h(\phi)$ expression is derived according to eq. 1.

$$m_h(\phi) = \frac{\sum_{i=1}^{J} \phi_i g\left(\left\|\frac{\phi-\phi_i}{h}\right\|^2\right)}{\sum_{i=1}^{J} g\left(\left\|\frac{\phi-\phi_i}{h}\right\|^2\right)} - \phi \qquad (1)$$

where ϕ is the current position of the vector of dimension d. When applying this algorithm to i-vectors' clustering, an i-vector ϕ is selected out of the pool of J i-vectors at the mean-shift algorithm first iteration (the total number of vectors to be clustered). h is the bandwidth and $g(\phi)$ is the kernel. The mean shift vector $m_h(\phi)$ is the difference of the current position (for instance vector ϕ) and the next position presented by the weighed sample mean vector of the neighborhood. The weights in the mean-shift vector formula are given, for simplicity, by the binary outputs (i.e. 0 or 1) of the flat kernel $g(\phi, \phi_i, h)$, as in eq. 2.

$$g(\phi,\phi_i,h) = \begin{cases} 1 & \left\|\frac{\phi-\phi_i}{h}\right\|^2 \le 1 \\ 0 & \left\|\frac{\phi-\phi_i}{h}\right\|^2 > 1 \end{cases} \qquad (2)$$

i.e. $g(\phi, \phi_i, h)$ selects a subset $S_h(\phi)$ of n_ϕ i-vectors in which the Euclidean pairwise distances with ϕ are less or equal to the threshold (bandwidth) h:

$$S_h(\phi) \equiv \{\phi_i : \|\phi - \phi_i\| \le h\} \qquad (3)$$

Then eq. 1 can be rewritten as following:

$$m_h(\phi) = \frac{\sum_{\phi_i \in S_h(\phi)} \phi_i}{\#S_h(\phi)} - \phi \qquad (4)$$

where $\#S_h(\phi)$ is the cardinality of $S_h(\phi)$.

The iterative process of calculating the sample mean followed by data shifting, converges to a mode of the data distribution, as presented at the Algorithm 1.

2.2. Modified mean shift algorithm

Several modifications were done in order to make the standard mean-shift algorithm more appropriate for

Algorithm 1 Mean-shift clustering algorithm

```
Require:
  A set of vectors, $\Phi = \{\phi_j\}_{j=1}^{J}$                ▷ $\phi \in \mathbb{R}^{d\times 1}$
  Neighborhood size $h$                                          ▷ $h \in \mathbb{R}^{+}$
  A cluster merging threshold $Th$                               ▷ $Th \in \mathbb{R}^{+}$
  for $j := 1$ to $J$ step $1$ do
    Set $\hat{\phi}_j = \phi_j$.
    repeat
      Find the subset $S_h\left(\hat{\phi}_j\right)$.
      Calculate $m_h\left(\hat{\phi}_j\right)$, the shift of the vector $\hat{\phi}_j$, using eq. 4.
      $\hat{\phi}_j \leftarrow \hat{\phi}_j + m_h\left(\hat{\phi}_j\right)$
    until Convergence
  Cluster the shifted vectors $\hat{\Phi} = \left\{\hat{\phi}_j\right\}_{j=1}^{J}$ such that the distance between 2 shifted vectors will be less then $Th$.
  Return: Cluster index of each vector.
```

speaker clustering. It is well known that using Euclidean distance as the similarity measure for speech analysis is not optimal. In [3] the authors apply the cosine distance, instead of the Euclidean one, and show that this improved the clustering performance significantly. The same distance measure was also applied for speaker diarization applications, where they performed mean-shift based clustering after removal of non-speech segments. Sapiro et. al. [15] used non flat weights for the shift calculation with cosine distance. In [12] a PLDA similarity was compared with the cosine distance and showed at the beginning some degradation in performances. However, following the work of [16], when the PLDA was trained over short segments while the TV matrix was still trained using long segments, the PLDA-based mean-shift clustering outperformed the cosine distance-based mean-shift clustering. In light of these results, the changes that we applied to the original mean-shift are as follows:

1. A *principal component analysis* (PCA) is used to reduce the dimensionality of the i-vector from d to $q < d$ (matrix **T**).

2. Whitening (matrix **C**) and length normalizations were applied to the low rank i-vectors to have their norm equal to 1. It was shown in [12] that the shift calculation over normalized i-vectors is more stable than when using un-normalized i-vectors.

3. We replaced the Euclidean distance by the *probabilistic linear discriminant analysis* (PLDA) score [17]. In this case we chose the i-vectors that have the highest PLDA-score in regard to the i-vector for which we calculate the shift, $s(\varphi, \varphi_i)$.

4. Instead of using a constant threshold h for selection of neighbors for calculating the mean-shift, we used the *k-nearest neighbors* (kNN) approach to choose the i-vectors for the mean-shift calculation. The only exception was we did not use i-vectors that yielded a negative score. In this case, it is assumed that even if the i-vector is in the k highest scores, it is very far and does not belong to the same speaker. The shift was calculated on the i-vectors before the dimensionality reduction. We denote the set of the closest vectors that are used for the shift calculation as $S_k(\phi)$.

According to the above modifications, the low dimensional i-vectors are as in eq. 5 and the shift calculation is as in eq. 6.

$$\varphi = \frac{\mathbf{CT}\phi}{\|\mathbf{CT}\phi\|} \tag{5}$$

$$m_h(\phi) = \frac{\sum\limits_{\phi_i \in S_k(\phi)} \phi_i}{\#S_k(\phi)} - \phi \tag{6}$$

and the modified mean shift algorithm is presented at Algorithm 2.

Algorithm 2 Modified mean-shift clustering algorithm

```
Require:
  A set of vectors, $\Phi = \{\phi_j\}_{j=1}^{J}$                ▷ $\phi \in \mathbb{R}^{d\times 1}$
  A set of vectors, $\varPhi = \{\varphi_j\}_{j=1}^{J}$          ▷ $\varphi \in \mathbb{R}^{q\times 1}$
  A number of neighbors $k$                                      ▷ $k \in \mathbb{N}$
  A cluster merging threshold $Th$                               ▷ $Th \in \mathbb{R}^{+}$
  for $j := 1$ to $J$ step $1$ do
    Set $\hat{\phi}_j = \phi_j$.
    repeat
      Calculate $\hat{\varphi}_j$ according to eq. 5.
      Find $k$ i-vectors with the highest score with $\hat{\varphi}_j$.
      Find $S_k\left(\hat{\phi}_j\right)$.
      Calculate $m_h\left(\hat{\phi}_j\right)$, the shift of the vector $\hat{\phi}_j$, using eq. 6.
      $\hat{\phi}_j \leftarrow \hat{\phi}_j + m_h\left(\hat{\phi}_j\right)$
    until Convergence
  Cluster the shifted vectors $\hat{\varPhi} = \{\hat{\varphi}_j\}_{j=1}^{J}$ such that the distance between 2 shifted vectors will be less than $Th$.
  Return: Cluster index of each vector.
```

2.3. The off-line clustering algorithm

After we explained the modified mean-shift algorithm, it is possible to describe the speaker clustering algorithm we used for the off-line part.

Before performing clustering, we train in advance the *universal background model* (UBM) and the *total variability* (TV) matrix for i-vectors extraction. Then, we train the PCA matrix **T**, and the whitening transformation matrix **C**. The low rank i-vectors calculated according to eq. 5. Then the PLDA is trained on short segments, the same as in [11, 12, 16].

Using the low dimension normalized i-vectors $\{\varphi\}$, we then train the PLDA model parameters, i.e. the within covariance matrix $\mathbf{W}^{-1}$, the between covariance matrix $\mathbf{B}^{-1}$ and the i-vectors expectation vector μ. Then, given a set of speech segments, we cluster them according to the following steps in Algorithm 3.

Algorithm 3 Off-line clustering algorithm

```
Require:
  A set of speech segments $\{Seg_j\}_{j=1}^{J}$
  A number of neighbors $k$                          $\triangleright k \in \mathbb{N}$
  A cluster merging threshold $Th$                   $\triangleright Th \in \mathbb{R}^+$
    1. for each speech segment $Seg_j$ extract an i-vector
       $\phi_j$ using UBM and $TV$.
    2. Using T and C perform dimensionality reduction
       and spherical normalization according to eq. 5 to
       have the low rank i-vectors $\Phi = \{\varphi_j\}_{j=1}^{J}$.
    3. Apply the modified mean-shift as described in
       Algorithm 2.
  Return: Cluster index for each speech segment.
```

3. Incremental on-line speaker clustering

When applying incremental clustering, the major change is the on-line mode. New speech segments arrive after clustering has already been performed over a set of other speech segments. After performing clustering (either off-line or incrementally), when a new segment arrives, it needs to be clustered. There are two options: the first is to add this segment to an existing cluster; The second option is to create a new cluster representing a new speaker. The decision is always based on the shifted i-vectors. The suggested procedure follows the logic of sequential k-means [18], where the means' update is done incrementally, when each new point is presented to the algorithm. Thus we cluster the new segments without shifting all the previously clustered segments again. This incremental clustering is thus based on the off-line clustering phase, as the new segment's shift uses all original J segments' shifts calculated during the off-line phase, so these shifts were not affected by the new segment that arrived later. Since theoretically, this new segment could have change the original clustering, we need to address this issue empirically, as we could have received a different clustering result if all $J+1$ segments where clustered together as in the off-line mode. We show in section 4 that for the task of short speech segments clustering, the clustering result that should represent the actual segments' speakers identity, is only slightly degraded by using this incremental clustering procedure. The procedure of incremental clustering of additional segments is shown in Algorithm 4.

Algorithm 4 Incremental clustering algorithm

```
Require:
  A set of segments, $\{Seg_j\}_{j=1}^{J}$
  A new segment $Seg_{J+1}$
  A number of neighbors $k$                          $\triangleright k \in \mathbb{N}$
  A cluster merging threshold $Th$                   $\triangleright Th \in \mathbb{R}^+$
    1. Apply algorithm 2 to cluster $\{Seg_j\}_{j=1}^{J}$.
    2. Save the set of the original i-vectors $\Phi = \{\phi_j\}_{j=1}^{J}$ and the shifted i-vectors, $\hat{\Phi} = \left\{\hat{\phi}_j\right\}_{j=1}^{J}$.
    3. Extract the i-vectors $\phi_{J+1}$ for the $Seg_{J+1}$ segment.
    4. For $\phi_{J+1}$
  repeat
      Calculate $\hat{\varphi}_{j+1}$ according to eq. 5.
      Find $k$ i-vectors with the highest score with $\hat{\varphi}_{j+1}$.
      Find $S_k\left(\hat{\phi}_{j+1}\right)$.
      Calculate $m_h\left(\hat{\phi}_{j+1}\right)$, the shift of the vector $\hat{\phi}_{j+1}$,
  using eq. 6.
      $\hat{\phi}_{j+1} \leftarrow \hat{\phi}_{j+1} + m_h\left(\hat{\phi}_{j+1}\right)$
  until Convergence
  Save the shifted i-vector $\hat{\phi}_{J+1}$
    5. Do:
  if $\exists i \bullet \left\|\hat{\phi}_{J+1} - \hat{\phi}_i\right\| < Th$ then
      Add the new speech segment to the same cluster of
  $i$-th segment
  else Create a new cluster
    6. $\Phi \leftarrow \Phi \cup \phi_{i+1}$
    7. $\hat{\Phi} \leftarrow \hat{\Phi} \cup \hat{\phi}_{i+1}$
    8. $J \leftarrow J + 1$
    9. if new segment arrives, Return to step 3
  Return: Cluster indexes for the speech segments,
  $\Phi$ and $\hat{\Phi}$.
```

When we compare the off-line mean-shift and the on-line mean-shift phases we notice one important difference. The off-line version is order independent. The order of the vectors chosen to be clustered does not matter - the final clustering will always be the same. In the on-line case we see the opposite - the order of the vectors may affect the final result. As each time a new vector (segment) arrives, it is added to the pool of vectors in addition to being clustered, the next coming vector can be influenced by the preceded one but not vice versa. This is why, during our experiments (section 4.5), the vectors are presented in a random order.

4. Experiments and results

The experiments were carried out on the NIST 2008 Speaker Recognition Database [19–21]. The test corpus short2-short3-Test7 was used to extract male speakers only for clustering.

4.1. Feature extraction and training data

In our experiments we used *Mel frequency cepstral coefficients* (MFCC), that were extracted using a 25 ms Hamming window. 19 MFCCs features together with log energy were extracted every 10 ms. This was followed by *cepstral mean subtraction* (CMS) and *variance normalization* (VN). Delta and delta delta coefficients were then calculated to produce all together 60-dimensional feature vectors. CMS and VN are not commonly used for diarization. The same holds for the delta features. However, when the application is such as homeland security, where segments supplied for the same speaker may arrive from different acoustic environments and/or through different channels, those normalizations are important and follow a similar procedure to the one used for speaker recognition. We used a male only UBM of 2048 Gaussian mixture components. It was trained using Fisher Part 1; Switchboard II, Phase 2; switchboard Cellular, Parts 1 and 2; and NIST 2004-2006 SREs. Total variability matrix with a low rank of 400 was trained using labeled data from the same databases as for the UBM. In total, 975 unique male speakers with 10705 sessions were used.

The test files include 5 minutes English telephone speech segments that were cut into small homogeneous speech segments. The average number of short segments extracted per speaker was 34. The distribution of the segment length L has minimal length of $L_{min} = 0.7$ s and average length $L_{av} = 2.5$ s and can be approximated using an exponential distribution.

$$L \sim L_{\min} + \exp(\lambda)\,; \quad \lambda = \frac{1}{L_{av} - L_{\min}}$$

The number of segments per speaker has average of $\eta = 34$ and standard deviation of $\sigma = 6.0$. The approximation of the speaker segments S is Gaussian, $S \sim \mathcal{N}\left(\eta, \sigma^2\right)$. Figure 1 shows the distributions of segments' lengths and the number of segments per speaker that were generated.

4.2. Evaluation Criterion

To evaluate the clustering performance we use the same criteria defined in [22]. The purity concept calculates both the *average cluster purity* (ACP) and *average speaker purity* (ASP). ASP is a measure of how well a speaker is limited to only one cluster, while ACP measures how well a cluster is limited to only one speaker. In the ideal case, both ACP and ACP are equal to 1.0. For ease of comparison between systems, the geometrical mean of ASP and ACP is used to obtain an overall evaluation criterion, K. The formulation of the evaluation

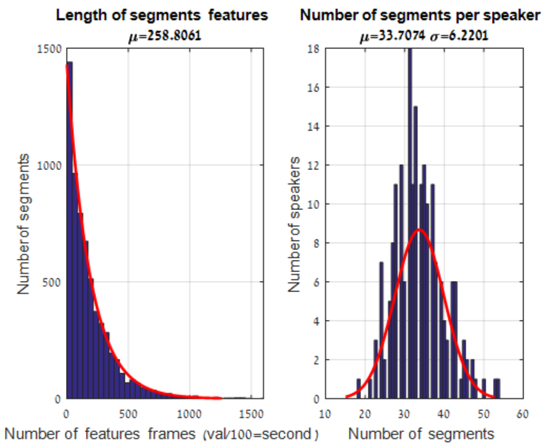

Figure 1: Segments lengths' distribution (on the left), and the distribution of the number of segments per speaker (on the right).

criteria is given in eq. 7, and the notation is as following:

- R - Number of speakers,
- Q - Number of clusters,
- n_{qr} - Total number of i-vectors in cluster q that are associated with speaker r,
- $n_{q.}$ - Total number of i-vectors in cluster q,
- $n_{.r}$ - Total number of i-vectors that are associated with speaker r.

$$\begin{aligned} ACP &= \frac{1}{J}\sum_{q=1}^{Q} p_{q.} \quad ; \quad p_{q.} = \sum_{r=1}^{R} \frac{n_{qr}^2}{n_{q.}^2} \\ ASP &= \frac{1}{J}\sum_{r=1}^{R} p_{.r} \quad ; \quad p_{.r} = \sum_{r=1}^{Q} \frac{n_{qr}^2}{n_{.r}^2} \\ K &= \sqrt{ACP \cdot ASP} \end{aligned} \tag{7}$$

ACP is based on the cluster purities $\{p_{q.}\}_{q=1}^{Q}$ and ASP is based on the speaker purities $\{p_{.r}\}_{q=1}^{Q}$. It is important to note that these purities are not probabilities.

4.3. Different examination modes

At each experiment we had to choose several parameters:

1. P_{sp} - Percentage of the speakers used for the off-line clustering, e.g., for $R = 30$, $P_{sp} = 80\%$, then 24 speakers are chosen randomly for the the off-line mode.

2. P_{seg} - Percentage of the segments chosen for the off-line clustering, e.g., if for speaker r the number of segments that is chosen is $K_r = 40$ and $P_{seg} = 80\%$, then $K_r \cdot P_{seg}/100 = 40 \cdot 0.8 = 32$,

will randomly chosen for the off-line mode, while 8 segments are kept as unseen segments for the incremental on-line mode.

The clustering performance is being examined on four different segment groups and algorithms as defined below.

- **offline:** The clustering results of the off-line clustering only. It gives the starting point for the incremental clustering. It includes only part of the speakers (according to P_{sp}) and only part of the segments (according to P_{seg}).
- **increment only:** The clustering results of the on-line segments subset only. The performance on the segments that were not seen during the off-line clustering. This is complimentary data to the one in the **offline** subset.
- **incremental:** The final results of all the clustering process, the off-line and the on-line modes. It is the combination of the results of the first 2.
- **full:** Clustering results in the case where all the data would have been clustered in an off-line mode. This yields, as expected, the best results that the clustering system can achieve. The gap between the **full** and the **incremental** is the degradation due to applying on-line clustering over part of the data.

4.4. Experimental procedure

As was shown in [2, 11, 12], the off-line clustering performance depends on two parameters. The first is the number of speakers in the data. The fewer the number of speakers the better the clustering. The second is the number of speech segments per speaker. When the amount of segments is relatively small,the degradation in performance is observed. Both parameters should be examined also in the perspective of incremental on-line clustering. We are testing the incremental on-line clustering, both for new segments of an existing speaker, and for a new speaker. The following procedure of experiments is applied: At first we randomly choose R speakers ($R \in \{3, 7, 15, 22, 30, 40, 50, 60\}$) for each clustering session. Then, about $P_{sp}\%$ of the speakers are randomly chosen to be used as the set for off-line clustering phase only, while all other speakers are treated as new speakers whose segments are presented only during the incremental on-line part. Next, we choose the segments, where on average 34 segments are chosen per speaker. Part of the segments of the speakers that participate in the off-line clustering are not shown to the system and are used as new segments in the incremental on-line mode, according to the chosen $P_{seg}\%$. These segments are also chosen randomly. $P_{seg}\%$ for the off-line mode and the rest for the incremental on-line mode. For the speaker that participate in the incremental on-line mode only, the number of segments is chosen in two steps. First, the initial number of segments is chosen (with 34 segments on the average). Then, out of this initial number only $100 - Pseg$ percents are chosen for the experiment.

After the two sets of segments are created, Algorithm 4 is applied. First the "off-line" segments are clustered and then the rest of segments are shown to the system in a random order and clustered using incremental on-line clustering.

4.5. Experiments

In the experimental part we tested two points of view. One is to see how different criteria behave for the four modes described in 4.3. We chose an operational point where the off-line initial condition is moderately good, most of the speakers are presented in the off-line clustering and there is a sufficient number of segments per speaker. In the second point of view, we test only the most popular K criterion and analyzed it in different operational points, starting with the working point in the first experiment and progressing to the more difficult case of having only a few segments per speaker in the off-line mode where most of the speakers are unseen until the incremental on-line phase.

Before starting to analyze the experiments we have to explain the way we present our results. It is important to mention that the horizontal axis is the number of speakers in the **incremental** graph, i.e., the number of speakers in the entire experiment. This means that the actual number of speakers in the **offline** graph is $\#\text{Speakers} \cdot P_{sp}/100$. Looking at the vertical line at each value of $\#\text{Speakers}$ provides the results of **offline**, **increment only**, **incremental** and **full**.

At the first experiment we took a test case with the following parameters: $P_{sp} = 80\%$, $P_{seg} = 80\%$. The results are presented in figure 2 in 4 subplots, for ACP, ASP, K, and the number of estimated clusters. The subplots for ACP, ASP and K are according to eq. 7. The fourth subplot shows the under-clustering of the clustering algorithm. In all the experiments we saw that the mean-shift algorithm yields under-clustering, and produces more clusters than the actual number of speakers (see [2, 11, 12, 15]). In some cases it is important to get an estimate of the under-clustering. As expected, the best results are obtained in the **offline** case. This is expected since only 80% of the speakers participate in this case, and for each of these speakers 80% of the segments are used (which is on the average about 25 segments), and it is a sufficient amount of segments to achieve reasonable clustering. The second best result is obtained for the **full** case, which also makes sense, as it uses all the segments but in the off-line mode; **incremental** clustering is next, and it means that the incremental segments harm the **of-**

fline results compared with both off-line cases, which is expected; The worst result is obtained for the **increment only** as it is applied to all the speakers, using only 20% of the segments per speaker, while clustering the segments one by one and not all together. Nevertheless, even the **increment only** yields ACP, ASP and K over 0.6. The estimation of the number of speakers behaves similarly.

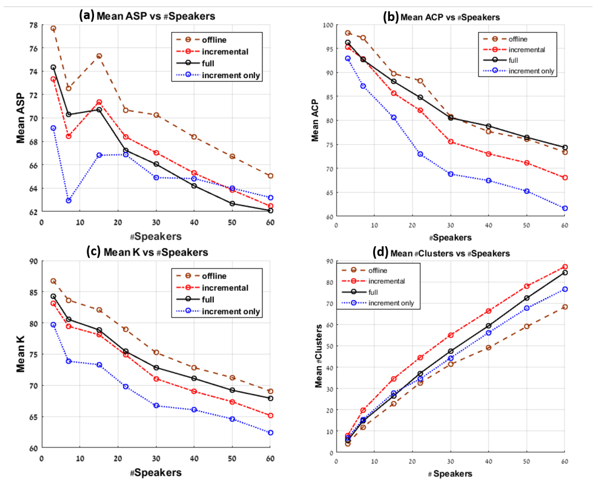

Figure 2: For $P_{sp} = 80\%$ and $P_{seg} = 80\%$ (a) *ASP; (b) ACP; (c) K; (d) Number of clusters.*

As the K value is usually the most informative, as it takes into account both ASP and ACP, in the second experiment we focused only on K, using different values of P_{sp} and P_{seg}. The results are shown in figure 3. Subplot (a) is the same as subplot (c) in figure 2. The **offline** is the best as it has less speakers and quite a lot of segments per speakers ($P_{sp} = 0.8$ and $P_{seg} = 0.8$). The **full** and **incremental** are similar as the additional data is relatively small and most of the clusters are well estimated during the off-line clustering. Naturally, the incremental on-line clustering is the worst but not much worse. In subplot (b) $P_{sp} = 0.8$ and $P_{seg} = 0.2$, the **offline** and **full** are very similar. The reason is probably that as in **offline** there are less speakers, which makes the clustering task easier, but the amount of segments per cluster is small, and that makes the clustering less accurate. These effects are the opposite to the **full** case where more clusters cause degradation while many segments per speaker increase the clustering performance. As most of the data is not seen during the off-line clustering and the clusters are not very accurate, the main clustering influence is due to the on-line clustering, so the results of the **incremental** and **incremental only** are similar. Subplots (c) and (d) show the cases where the amounts of data for both number of speakers and number of segments per speaker, decrease. The **offline** seems to be better than **full** as the number of speakers is much less (only 50% in (c) and 20% in (d)), however, it leads to a very significant degradation for the **incremental** which is very similar to **incremental only** as very few speakers are seen during the off-line clustering and the clusters are not well estimated as there are not many segments per speaker. It can be observed that as the number of segments in the off-line clustering is higher and the incremental part do not have many new speakers, the incremental on-line clustering yields good results. On the other hand, insufficient data at the off-line clustering leads to much poorer incremental on-line clustering.

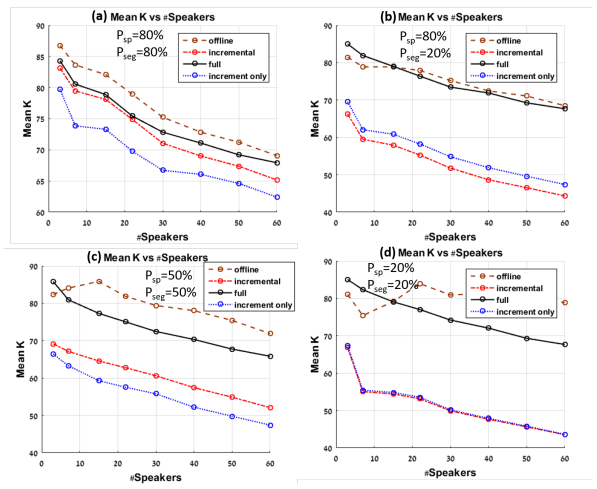

Figure 3: K values for (a) $P_{sp} = 80\%$ and $P_{seg} = 80\%$; (b) $P_{sp} = 80\%$ and $P_{seg} = 20\%$; (c) $P_{sp} = 50\%$ and $P_{seg} = 50\%$; (d) $P_{sp} = 20\%$ and $P_{seg} = 20\%$.

5. Conclusions

In this work we showed empirically that after off-line clustering of short speech segments using the mean-shift algorithm with PLDA as the similarity measure, it is possible to continue to cluster newly arrived segments in an incremental on-line procedure. It works not only for new segments from already seen speakers, but also for segments that belong to new speakers. The algorithm works for a wide range of speakers' number, from 3 to 60. We have observed that it is important to have sufficient amounts of segments per speaker for the off-line clustering phase, so the incremental clustering can rely on a reliable initial clustering. In graphs (c) and (d) of figure 3, we can see how the incremental clustering degrades when the amounts of data for the off-line clustering is reduced. One possible solution, that can be applied in several scenarios, is to perform once in a while the off-line clustering using all speech segments. This way we can improve the initial clustering before applying the off-line clustering on all the collected data that have arrived till this step, while still using incremental clustering in between these time steps in which off-line re-clustering is applied.

As representing short segments by i-vectors is not optimal, we intend in the future to try some other representations, for example the one described in [23], that is based on DNNs.

6. References

[1] S. E. Tranter and D. A. Reynolds, “An overview of automatic speaker diarization systems,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1557–1565, Sept 2006.

[2] I. Salmun, I. Opher, and I. Lapidot, “On the use of plda i-vector scoring for clustering short segments,” in *ODYSSEY 2016 -The Speaker and Language Recognition Workshop*, 2016.

[3] M. Senoussaoui, P. Kenny, T. Stafylakis, and P. Dumouchel, “A study of the cosine distance-based mean shift for telephone speech diarization,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 1, pp. 217–227, Jan 2014.

[4] S. Shum, N. Dehak, R. Dehak, and J. Glass, “Unsupervised methods for speaker diarization: An integrated and iterative approach,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 10, pp. 2015–2028, Oct 2013.

[5] S. Chen and P. Gopalakrishnan, “Speaker, environment and channel change detection and clustering via the bayesian information criterion,” Feb. 1998.

[6] O. Ben-Harush, I. Lapidot, and H. Guterman, “Online diarization of telephone conversations,” in *ODYSSEY 2010 -The Speaker and Language Recognition Workshop*, 2010.

[7] ——, “Incremental diarization of telephone conversations,” in *Proceedings of Interspeech 2010*, 2010.

[8] T. Koshinaka, K. Nagatomo, and K. Shinoda, “Online speaker clustering using incremental learning of an ergodic hidden markov model,” *IEICE Transactions on Information and Systems*, vol. E95.D, no. 10, pp. 2469–2478, 2012.

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

[10] G. Soldi, C. Beaugeant, and N. Evans, “Adaptive and online speaker diarization for meeting data,” in *2015 23rd European Signal Processing Conference (EUSIPCO)*, Aug 2015, pp. 2112–2116.

[11] I. Salmun, I. Opher, and I. Lapidot, “Improvements to plda i-vector scoring for short segments clustering,” in *2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE)*, Nov 2016, pp. 1–4.

[12] I. Salmun, I. Shapiro, I. Opher, and I. Lapidot, “Plda-based mean shift speakers’ short segments clustering,” *Computer Speech and Language*, vol. 45, pp. 411 – 436, September 2017.

[13] K. Fukunaga and L. Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition,” *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, Jan 1975.

[14] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[15] I. Shapiro, N. Rabin, I. Opher, and I. Lapidot, “Clustering short tush-to-talk segments,” in *Proceedings of Interspeech 2015*, September 2015.

[16] O. Kudashev, T. Pekhovsky, and O. Khomitsevich, “Speaker diarization system based on probability linear discriminant analysis,” ITMO University, Tech. Rep., 2015.

[17] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[18] L. Bottou and Y. Bengio, “Convergence properties of the k-means algorithms,” in *Advances in Neural Information Processing Systems 7*. MIT Press, 1995, pp. 585–592.

[19] “Linguistic data consortium,” LDC2011S05, Catalog, 2011, available: https://catalog.ldc.upenn.edu/LDC2011S05.

[20] “Linguistic data consortium,” LDC2011S07, Catalog, 2011, available: https://catalog.ldc.upenn.edu/LDC2011S07.

[21] “Linguistic data consortium,” LDC2011S08, Catalog, 2011, available: https://catalog.ldc.upenn.edu/LDC2011S08.

[22] J. Ajmera, H. Bourlard, I. Lapidot, and I. McCowan, “Unknown-multiple speaker clustering using HMM,” in *In Proceedings of ICSLP-2002*, 2002, pp. 573–576.

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