Application of Fuzzy and Possibilistic *c*-Means Clustering Models in Blind Speaker Clustering

Gábor Gosztolya[1], László Szilágyi[2,3]

[1] MTA-SZTE Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged, Tisza Lajos krt. 103, H-6720 Szeged, Hungary, ggabor@inf.u-szeged.hu

[2] Dept. of Control Engineering and Information Technology, Budapest University of Technology and Economics, Hungary

[3] Computational Intelligence Research Group, Dept. of Electrical Engineering, Sapientia University of Transylvania, Tîrgu Mureş, Romania, lalo@ms.sapientia.ro

Abstract: Blind Speaker Clustering is a task within speech technology, where we have a collection of speech recordings (utterances), and the goal is to identify which utterances belong to the same speakers. To aid the clustering process in this task, we performed pre-processing steps such as feature selection and Principal Component Analysis (PCA); still, the choice of clustering method is not a trivial one. To find the best performing algorithm, we tested standard methods such as k-means (or hard c-means, HCM) and fuzzy c-means (FCM) as well as several improved versions of FCM. In the end, we were able to achieve the best performance using probabilistic-possibilistic mixture partitions. The obtained purity score of 83.9% is significantly higher than the baseline score of 46.9%.

Keywords: clustering; fuzzy c-means algorithm; possibilistic c-means algorithm; speech technology

1 Introduction

Automatic Speech Recognition (ASR) seeks to create the correct transcription (written form) of an utterance (a recording containing speech). Traditionally, speech technology researchers focused primarily on ASR (e.g. [1-3]), but in the last few years another area has received growing attention. It is called computational paralinguistics, and it seeks to extract non-verbal information from the speech signal. This area includes tasks such as emotion detection [4, 5], speaker age estimation [6], conflict intensity estimation [7-9], detecting social signals like laughter and filler events [5], and estimating the amount of physical or cognitive load during speaking [10-12]. Several of these tasks attempt to detect phenomena which vary from speaker to speaker. Therefore, in these tasks, if we could identify which utterances (recordings) belong to the same person, it would clearly assist the following classification or regression step. This task, called Blind Speaker Clustering (or just Speaker Clustering), is a viable tool in computational paralinguistics; for example, it was shown that speaker-wise data normalization can lead to a significantly improved classification performance compared to using global normalization techniques [10, 13].

Speaker clustering is an existing, current problem in speech recognition literature (e.g. [14-16]. In most cases, however, it has to be performed along with speaker segmentation ("Who spoke when?"), while now we have only one speaker in an utterance; hence we have to concentrate only on speaker clustering. Note that an important aspect of this task is that we have to work with the utterances of speakers that are unseen to us at training time, so it is clearly a clustering task.

Clustering means that we form groups of those examples which are similar to each other and different from the others, but this definition does not tell us *in which sense* they are different. For example, speech utterances may be similar if they record the speech of the same speaker, or the speakers utter the same sentences, or they were recorded under similar conditions (microphone, background noise), etc. In the actual task, however, we would like to separate the different speakers. A straightforward choice to control the way of clustering is by applying feature selection. If we keep only those attributes which correlate well with the desired property of examples, we can control the type of clusters formed. Still, we have to keep in mind the fact that we also have to avoid choosing a redundant feature subset, as it can also hinder the clustering process.

Besides feature selection, an important choice is that of the clustering method. A straightforward choice is the *k*-means (or hard *c*-means, HCM, [17]) algorithm; however, it is a stochastic algorithm, which is vulnerable to random initialization. To this end, we also test fuzzy *c*-means (FCM [18]) in this task, as well as three of its improved variants ([9, 19, 20]).

2 Blind Speaker Clustering by Feature Selection

Blind Speaker Clustering can be simply viewed as a clustering problem, for which standard clustering methods such as the HCM and FCM algorithms can be readily applied. However, these methods have a weak point, namely that they work in a multi-dimensional space treating all dimensions as equally important (as they rely on the Euclidean distance of the points). This means that they are sensitive to differently-scaled, redundant and irrelevant features.

The first issue can be handled by normalization, i.e. all the features can be normalized (i.e. scaled to a fixed interval, e.g. [0, 1] or [-1, 1]) or standardized (i.e. transformed so as to have zero mean and unit standard deviation). The other two issues can be handled via feature selection; in fact, we can turn the feature selection to our advantage so that we just keep those features which help us create the right kind of clusters (in our case, different speakers).

To be able to perform this, we will need several things. First, to measure which feature set allows us to form better clusters, we will need a set of recordings with their correct classes (now: speakers) annotated. Second, we will have to choose an evaluation metric by which we will rank the results of the different clustering outcomes. Third, we will need to choose (or construct) a feature selection method.

2.1 Performing Feature Selection

A wide range of feature selection algorithms exist (e.g. [21, 22]; most of them, however, have a high computational complexity. To this end, we applied some quite quick and simple pre-processing steps to perform feature selection.

Feature selection has to deal with two phenomena, namely irrelevant and redundant features. In the first case, the problem is that some features do not assist the forming of the desired clusters, or even distract the clustering algorithms (e.g. describe relations that the actual speaker mentioned, and not *who* spoke in that given utterance). Yet, the redundant features describe the same phenomenon in a very similar way. As most clustering algorithms treat each feature as an equally important dimension, redundant features will have a larger importance overall, hence will distract the clustering method used.

2.1.1 Handling Irrelevant Features

We handled the issue of irrelevant features by applying a simple feature selection method. We took the feature vectors of two speakers, and calculated the correlation between each feature with the change of speakers. We repeated this for each speaker pair, and the absolute values of the resulting correlation values were averaged out. Then, the features were sorted according to their averaged correlation score in descending order, and we selected the most correlated features. This way, we also had control over the type of clusters formed.

2.1.2 Handling Redundant Features

The issue of redundancy was dealt with by using Principal Component Analysis (PCA, [23]). PCA is a statistical method which transforms our observation vectors into a space described by linearly uncorrelated directions (the principal components) via an orthogonal transformation. That is, the first direction returned by the PCA will point to the direction where the variance of our data is the highest; the further directions will point to the directions which have the largest possible variance, provided that they are orthogonal to all previous directions. By transforming our examples into this coordinate system, and then performing normalization, we can get rid of most of the redundancy in our attributes.

PCA also supplies information about the importance of each new dimension (feature) describing the examples. It is common practice to keep only the first directions which describe at least a given amount (e.g. 90%) of the information stored in the example set, thereby applying PCA as a feature extraction tool [24]. Of course, the amount of information to be retained is not a trivial one, hence we experimented with different thresholds for this value as well.

3 The Employed Clustering Methods

Given a set of feature vectors $\mathbf{X} = \{ \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \}$ describing n objects, the fuzzy c-means (FCM) algorithm can produce a fuzzy partition into a predefined number of clusters c, based on the minimization of the quadratic objective function

$$J_{FCM} = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \left\| \mathbf{x}_k - \mathbf{v}_i \right\|^2, \tag{1}$$

under the probabilistic constraint $\sum_{i=1}^{c} u_{ik} = 1$, $\forall k = 1,\ldots,n$, where $\mathbf{v}_i$ represents the prototype or centroid of cluster i ($i = 1,\ldots,c$), $u_{ik} \in [0,1]$ is the fuzzy membership function showing the degree to which the vector $\mathbf{x}_k$ belongs to cluster i, and $m > 1$ is the fuzzyfication parameter. The minimization of the objective function J_{FCM} is achieved by alternately applying the optimization of J_{FCM} over u_{ik} with $\mathbf{v}_i$ fixed, $i = 1,\ldots,c$, and the optimization of J_{FCM} over $\mathbf{v}_i$ with u_{ik} fixed, $i = 1,\ldots,c$, $k = 1,\ldots,n$ [18]. In each loop, the optimal values are deduced from the zero gradient conditions using Lagrange multipliers, and are computed using the following formulas:

$$u_{ik}^{*} = \frac{\left\| \mathbf{x}_k - \mathbf{v}_i \right\|^{-2/(m-1)}}{\sum_{j=1}^{c} \left\| \mathbf{x}_k - \mathbf{v}_j \right\|^{-2/(m-1)}} \qquad \begin{array}{l} \forall i = 1,\cdots,c \\ \forall k = 1,\cdots,n \end{array} \tag{2}$$

$$\mathbf{v}_i^{*} = \frac{\sum_{k=1}^{n} u_{ik}^{m} \mathbf{x}_k}{\sum_{k=1}^{n} u_{ik}^{m}} \qquad \forall i = 1,\cdots,c\,. \tag{3}$$

According to the alternating optimization (AO) scheme of the FCM algorithm, eqs. (2) and (3) are alternately applied, until the cluster prototypes stabilize. This stopping criterion compares the sum of norms of the variations of the prototype vectors $\mathbf{v}_i$ within the latest iteration, with a predefined small threshold value ε. The algorithm requires proper initialization of the cluster prototypes. In our case, we have assigned randomly chosen input vectors to cluster prototypes, and ensuring that $\forall i,j \in \{1,\dots,c\}$ and $i \neq j$ we have $\mathbf{v}_i \neq \mathbf{v}_j$.

Hard *c*-means [17] is a special case of FCM, when $m \to 1$ and thus the memberships are obtained according to the winner-takes-all rule. Each cluster prototype is obtained as the mean of input vectors assigned to the given cluster. The first time when the partition does not change during an iteration, the convergence is achieved.

3.1 FCM Variants Employed in this Study

Several families and variants of *c*-means clustering models have been introduced recently, which reportedly produce better partitions than FCM in several applications. In the following, we enumerate those applied in our study.

3.1.1 Adding Possibilistic Component to Fuzzy *c*-means Clustering

The **possibilistic *c*-means clustering** (**PCM**) algorithm assigns typicality values to fuzzy membership functions [25]. Thus in PCM, the elements of the partition matrix, denoted by t_{ik} instead of u_{ik} ($i = 1,\dots,c$, $k = 1,\dots,n$), describe how compatible the input vectors are with the clusters represented by the computed cluster prototypes. Typicality values with respect to one cluster do not depend on any of the prototypes of other clusters.

Since PCM often produces coincident clusters, Pal et al. introduced a mixture clustering model called possibilistic-fuzzy c-means (PFCM) clustering that comprises a probabilistic and a possibilistic term [19]. PFCM optimizes the objective function:

$$J_{FPCM} = \sum_{i=1}^{c}\sum_{k=1}^{n}[au_{ik}^{m} + bt_{ik}^{p}]\|\mathbf{x}_k - \mathbf{v}_i\|^2 + \sum_{i=1}^{c}\eta_i\sum_{k=1}^{n}(1-t_{ik})^p, \qquad (4)$$

where the fuzzy membership functions u_{ik} ($i = 1,\dots,c$, $k = 1,\dots,n$) are constrained by the probabilistic conditions, while the typicality values $t_{ik} \in [0,1]$ ($i = 1,\dots,c$, $k = 1,\dots,n$) are subject to: $0 < \sum_{i=1}^{c} t_{ik} < c$, $\forall i = 1,\dots,c$. The fuzzy exponent m and possibilistic exponent p must be greater than 1, while a and b are tradeoff parameters to set the balance between the probabilistic and possibilistic term. The variables η_i ($i = 1,\dots,c$) are called possibilistic penalty terms and control the variance of the clusters. The optimization formulas applied in each loop of the alternating optimization are:

$$t_{ik}^{*} = \left[1 + \left(\frac{b\|\mathbf{x}_k - \mathbf{v}_i\|^2}{\eta_i} \right)^{1/(p-1)} \right]^{-1} \qquad \begin{matrix} \forall i = 1, \cdots, c \\ \forall k = 1, \cdots, n \end{matrix}, \tag{5}$$

$$\mathbf{v}_i^{*} = \frac{\sum_{k=1}^{n} \left[a u_{ik}^{m} + b t_{ik}^{p} \right] \mathbf{x}_k}{\sum_{k=1}^{n} \left[a u_{ik}^{m} + b t_{ik}^{p} \right]} \qquad \forall i = 1, \cdots, c\,. \tag{6}$$

The probabilistic part of the partition is computed exactly the same way as in FCM, according to Eq. (2). This algorithm was found to be robust in several tests.

3.1.2 Fuzzy-Possibilistic Product Partition *c*-means

The **fuzzy-possibilistic product partition *c*-means** (**FPPPCM**) algorithm was introduced with the goal to eliminate the outlier sensitivity of previous mixture clustering models [20]. This partition also employs a probabilistic and a possibilistic term, but it combines them via multiplication instead of via linear combination. The algorithm optimizes the objective function:

$$J_{FPPPCM} = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^{m} \left[t_{ik}^{p} \|\mathbf{x}_k - \mathbf{v}_i\|^2 + (1 - t_{ik})^{p} \eta_i \right], \tag{7}$$

constrained by the conventional probabilistic and possibilistic conditions mentioned above. The only parameters of FPPPCM are the fuzzy exponent $m > 1$, the possibilistic exponent $p > 1$, and the conventional penalty terms of the possibilistic partition denoted by η_i, $i = 1,\ldots,c$. The optimization formulas that stem from zero gradient conditions using Lagrange multipliers are:

$$t_{ik}^{*} = \left[1 + \left(\frac{\|\mathbf{x}_k - \mathbf{v}_i\|^2}{\eta_i} \right)^{1/(p-1)} \right]^{-1} \qquad \begin{matrix} \forall i = 1, \cdots, c \\ \forall k = 1, \cdots, n \end{matrix}, \tag{8}$$

$$u_{ik}^{*} = \frac{\left[t_{ik}^{p} \|\mathbf{x}_k - \mathbf{v}_i\|^2 + \eta_i (1 - t_{ik})^{p} \right]^{-1/(m-1)}}{\sum_{j=1}^{c} \left[t_{jk}^{p} \|\mathbf{x}_k - \mathbf{v}_j\|^2 + \eta_j (1 - t_{jk})^{p} \right]^{-1/(m-1)}} \qquad \begin{matrix} \forall i = 1, \cdots, c \\ \forall k = 1, \cdots, n \end{matrix}, \tag{9}$$

$$\mathbf{v}_i^* = \frac{\sum_{k=1}^{n} u_{ik}^m t_{ik}^p \mathbf{x}_k}{\sum_{k=1}^{n} u_{ik}^m t_{ik}^p} \qquad \forall i = 1, \cdots, c\,. \tag{10}$$

This algorithm has its main advantage of having a reduced number of parameters. It was found to efficiently reject the effect of outliers while being accurate also in the absence of outliers.

3.1.3 Suppressed FCM

Suppressed FCM was introduced with the intent of combining the quick convergence of HCM with the fine partitions produced by FCM. It manipulates with fuzzy membership functions produced by the FCM algorithm using Eq. (2): for each vector $\mathbf{x}_k$ it looks for the closest cluster prototype, say $\mathbf{v}_w$, applies suppression by multiplying all u_{ik} values by a suppression rate $\alpha \in [0,1]$, and increases u_{wk} by 1 - α to maintain the probabilistic constraint [26]. These modified fuzzy membership values are then fed to Eq. (3) to update the cluster prototypes. The algorithms obtained for various values of α are reportedly quick and accurate in most clustering problems [27].

4 Experimental Setup

Next we will describe the way our experiments were performed: the way clustering accuracy was measured, the database used, the feature set extracted from the examples, and the way the parameters of the clustering methods were set.

4.1 Evaluation Metrics

If the real groups of examples (in our case, the different speakers) are known, we can evaluate a clustering hypothesis generated via an automatic clustering method (*external evaluation*, [28]). However, this is more difficult to do than for classification, as we cannot be sure which resulting cluster corresponds to which group (if any). Perhaps this is why there are several evaluation metrics available for this purpose.

One of the metrics that can be used for clustering evaluation is purity; this metric takes the most frequent class label in each cluster, and calculates the ratio of the elements in the cluster which belong to this class [28-30]. Then, these scores are averaged out for all clusters by weighting them with the number of their elements.

That is, for $\Omega = \{\omega_1,\dots,\omega_c\}$ (the set of resulting clusters), $C = \{\xi_1,\dots,\xi_N\}$ (the set of real groupings) and n elements ($\sum|\omega_j| = \sum|\xi_i| = n$), we calculate

$$Purity(\Omega, C) = \frac{1}{n}\sum_{j=1}^{c}\max_i\left|\omega_j \cap \xi_i\right|. \tag{11}$$

Bad clustering has a purity value close to zero, while a perfect clustering has a purity score of one. It has the drawback that it is easy to achieve high purity scores when the number of clusters (c) is large, but as in our case this is known in advance, we can set $c = N$ (the number of speakers) and handle this problem.

Another possibility is to use entropy [28, 30], which is defined as

$$E(\omega_j) = -\frac{1}{\log N}\sum_{i=1}^{N}\frac{\left|\omega_j \cap \xi_i\right|}{\left|\xi_j\right|}\log\frac{\left|\omega_j \cap \xi_i\right|}{\left|\xi_j\right|} \tag{12}$$

for any $j = 1,\dots,c$, and the entropy of the C clustering will be the sum of the $E(\omega_j)$ values weighted by the number of the elements. That is,

$$Entropy(\Omega, C) = \sum_{j=1}^{c}\frac{\left|\omega_j\right|}{n}E(\omega_j). \tag{13}$$

The better a clustering, the lower the entropy value it has; a perfect clustering has zero entropy.

4.2 The Munich Biovoice Corpus

We performed our experiments on the Munich Biovoice Corpus (MBC, [31]). It contains the utterances of 19 subjects (4 female and 15 male) of three nations (Chinese, German and Italian) both after light and heavy physical load. They had to pronounce sustained vowels as well as reading a short story, which was recorded by two different microphones. Besides the audio recordings, heart rate and skin conductivity was monitored as well. The dataset was later used in the Interspeech ComParE 2014 Physical Load Sub-Challenge [10].

4.3 Experimental Setup

In our experiments we employed the feature set used in [10]. It contained 6373 features overall, extracted by using the tool called openSMILE [32]. The set includes energy, spectral, cepstral (MFCC) and voicing related low-level descriptors (LLDs), as well as a few other LLDs including logarithmic harmonic-to-noise ratio (HNR), spectral harmonicity, and psychoacoustic spectral sharpness.

Similarly to other machine learning areas, separate training and test sets were defined, consisting of 6 speakers each. The feature selection process including the application of PCA was performed on the training set, as well as the parameter setting of the clustering algorithms (*c* for the FCM and its variants and α for s-FCM). Then, the test set was transformed in a similar way to the training one (i.e. using the same (basic) features, then transformed by PCA using the same principal components, and keeping the same number of transformed attributes). Lastly, the transformed test set was clustered using the clustering parameter values obtained on the training set, and the result was evaluated using the purity and entropy metrics.

For the pre-processing steps, we experimented with keeping the 20, 50 and 100 most correlated features; after PCA, we kept 75%, 90%, 95% and 99% of the information.

4.4 Parameter Setting for the Clustering Methods

Although $m = 2$ is the most frequently employed value for the fuzzy exponent, it is not suitable when the number of dimensions is several dozens because it leads all cluster prototypes to the grand mean of the input data. In all algorithms that contain the probabilistic exponent *m*, we tested values in the range of 1.05 to 1.5. For all algorithms that use the possibilistic exponent *p*, we set $p = m$. Possibilistic penalty terms η_i ($i = 1,\ldots,c$) were always chosen equal for all clusters, but their value always depended on the actual number of dimensions *d*. In case of PFCM algorithm, fine results were obtained for $0.6\sqrt{d} \leq \sqrt{\eta_i} \leq 1.25\sqrt{d}$, while FPPPCM performed best for $1.2\sqrt{d} \leq \sqrt{\eta_i} \leq 2\sqrt{d}$. The actual number of dimensions varied between 2 (for 20 features and 75% PCA) and 54 (for 100 features and 99% PCA). For the suppressed FCM algorithm, all suppression rates multiple of 0.1 were considered, but most accurate results were obtained in the range $0.5 \leq \alpha \leq 0.8$.

As HCM has no parameters at all, it required no parameter adjustment. However, as it is not a robust procedure, for this algorithm we performed 100 clusterings for each preprocessing configuration, and averaged out the resulting purity and entropy scores.

5 Results

The resulting purity scores can be seen in Table 1, while the corresponding entropy values are listed in Table 2. The best values for a pre-processing configuration are shown in **bold**. We can see that by increasing the number of features, the quality of clustering also improves, and it also improves if we retain more information after the PCA step. When using 100 features, however, the difference is quite small between keeping 95% or 99% of the information; on the other side, it is pointless using fewer than 50 features, or keeping only 75% of the information after the PCA step, as the resulting purity scores are pretty low.

Table 1

Purity scores achieved with the different preprocessing configurations and clustering algorithms

Inform. Kept after PCA	Clustering Method	Number of Features					
		20		50		100	
		Train	Test	Train	Test	Train	Test
75%	HCM	68.6%	59.2%	77.7%	59.9%	77.9%	59.1%
	FCM	68.6%	**59.4%**	80.5%	61.2%	79.7%	60.4%
	s-FCM	69.1%	58.6%	80.8%	61.7%	79.0%	60.9%
	PFCM	72.7%	57.6%	82.1%	62.5%	80.0%	61.4%
	FPPPCM	73.0%	57.3%	82.3%	**66.2%**	80.8%	**66.4%**
90%	HCM	77.1%	64.8%	84.2%	69.3%	80.3%	74.2%
	FCM	76.9%	64.3%	85.2%	76.6%	85.5%	77.6%
	s-FCM	77.1%	**65.1%**	84.9%	75.5%	85.7%	75.0%
	PFCM	77.1%	**65.1%**	86.0%	76.0%	87.3%	78.1%
	FPPPCM	76.9%	**65.1%**	86.8%	**82.0%**	88.9%	**80.7%**
95%	HCM	73.3%	67.7%	86.0%	71.4%	79.2%	76.3%
	FCM	74.6%	65.9%	86.0%	**80.0%**	85.7%	78.1%
	s-FCM	76.6%	65.9%	85.2%	78.4%	86.5%	77.1%
	PFCM	75.1%	67.2%	87.0%	78.4%	88.3%	77.1%
	FPPPCM	75.9%	**68.5%**	87.8%	76.8%	89.6%	**83.1%**
99%	HCM	73.5%	66.2%	85.2%	75.0%	80.2%	79.5%
	FCM	75.8%	69.5%	86.0%	80.2%	85.5%	79.7%
	s-FCM	75.1%	**70.6%**	86.2%	76.0%	87.0%	82.0%
	PFCM	76.9%	64.8%	86.2%	76.0%	86.8%	82.3%
	FPPPCM	76.1%	66.2%	86.8%	**82.6%**	88.8%	**83.9%**

Regarding the choice of the clustering method, it is clear that HCM performed the worst; the likely reason for this is that it is a stochastic method. There is no great difference among the performances of the other four methods, but generally, FPPPCM performed best both on the training and on the test sets. This seems to indicate that it is not just a method that can be fine-tuned to suit our needs, but the tuned parameter values perform well on another set of examples (e.g. the test set), meaning that the method is a very robust one.

Table 2
Entropy scores achieved with the different preprocessing configurations and clustering algorithms

Inform. Kept after PCA	Clustering Method	Number of Features					
		20		50		100	
		Train	Test	Train	Test	Train	Test
75%	HCM	0.428	**0.544**	0.327	0.577	0.307	0.611
	FCM	0.421	0.548	0.328	0.565	0.292	0.598
	s-FCM	0.425	0.559	0.327	0.547	0.289	0.599
	PFCM	0.358	0.552	0.312	0.555	0.293	0.579
	FPPPCM	0.359	0.546	0.277	**0.510**	0.254	**0.544**
90%	HCM	0.313	0.456	0.265	0.462	0.285	0.450
	FCM	0.309	0.488	0.280	**0.416**	0.256	0.434
	s-FCM	0.313	**0.453**	0.256	0.446	0.262	0.438
	PFCM	0.313	**0.453**	0.256	0.446	0.242	0.422
	FPPPCM	0.312	0.467	0.235	0.467	0.211	**0.372**
95%	HCM	0.336	0.442	0.232	0.434	0.305	0.407
	FCM	0.315	0.494	0.270	0.397	0.259	0.405
	s-FCM	0.314	0.476	0.248	0.397	0.255	0.371
	PFCM	0.309	0.442	0.248	0.397	0.233	0.407
	FPPPCM	0.317	**0.436**	0.238	**0.385**	0.206	**0.335**
99%	HCM	0.340	0.491	0.240	0.406	0.290	0.371
	FCM	0.313	0.451	0.263	0.380	0.259	0.404
	s-FCM	0.312	0.472	0.223	0.404	0.241	0.361
	PFCM	0.306	0.443	0.230	0.404	0.247	0.347
	FPPPCM	0.313	**0.427**	0.254	**0.327**	0.217	**0.328**

In general, the variations of fuzzy *c*-means performed somewhat better than the standard algorithm: the latter achieved its best results with 50 features, while the other three methods were able to utilize the extra information stored in the additional features, thus achieving a clustering, which is of a better quality.

Figure 1 shows the purity scores given by the employed set of clustering algorithm in various scenarios. It is evident that FCM's accuracy drops as the fuzzy exponent m grows beyond a critical value situated around 1.3. Among all tested algorithms, FPPPCM performed the best, while the other fuzzy and possibilistic approaches provided results of approximately same quality, but better than the outcome of HCM. FPPPCM even gave purity scores above 0.85, but that setting never coincided with the best performing scenario on the train data set.

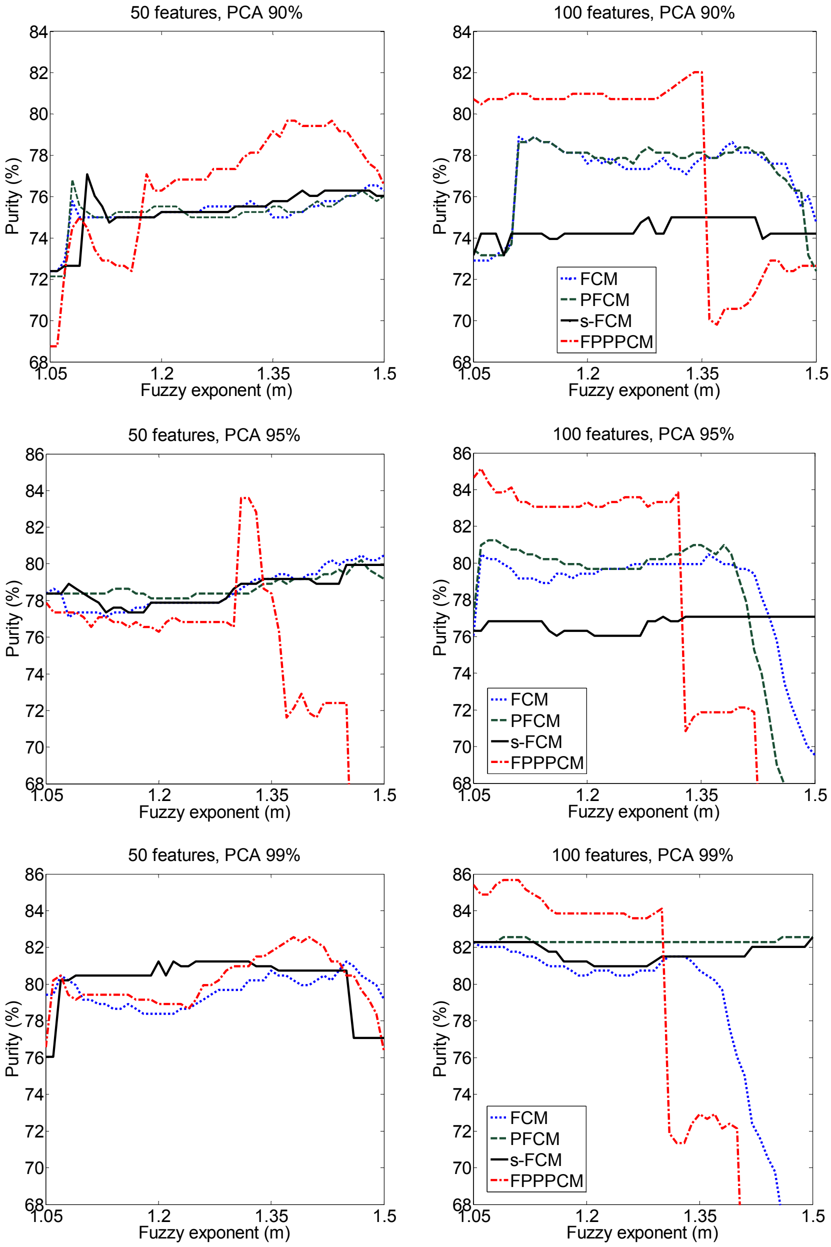

Figure 1

The purity scores obtained plotted against the fuzzy exponent *m* for the different clustering methods and preprocessing configurations on the training set. In case of 50 features and PCA 99%, the black curve fully covers the green one

The results of two clustering configurations can be seen on Figure 2: the left hand side shows the baseline setting, using all the 6373 features with the standard HCM clustering method, while the right hand side shows the FPPPCM method with the optimal configuration, using 100 features and keeping 99% of the information after PCA. In the latter case, clearly most of the utterances belonging to a given speaker could be mapped in the same cluster (see the rectangles near the diagonal). A number of utterances were assigned to wrong speakers (these form small straight lines). Overall, this clustering is of a much higher quality than the baseline one shown on the left hand side, where some speakers were confused by each other (see the boxes off the diagonal). This is reflected by the accuracy values as well: while the baseline setting had a purity score of 46.9% and an entropy value of 0.560 on the test set, we were able to achieve scores of 83.9% and 0.328, purity and entropy, respectively.

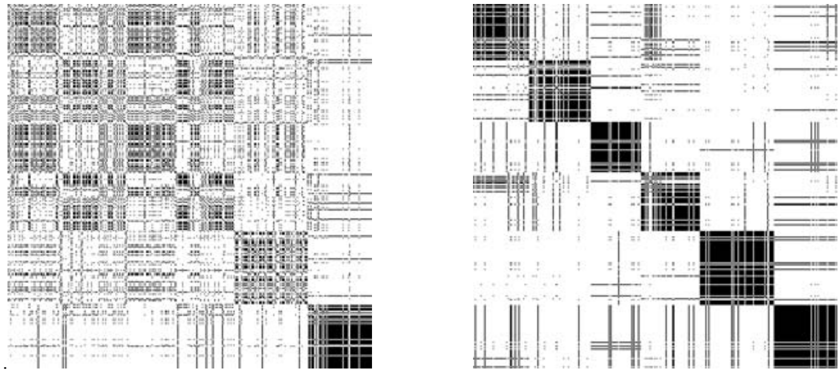

Figure 2

"Confusion matrix" of the test set when using all the features with the HCM method (left), and the best configuration of FPPPCM (right). Each row and column corresponds to one utterance; each point shows whether the corresponding utterances were assigned to the same cluster.

Conclusions

Blind Speaker Clustering (or simply Speaker Clustering) is a task where we have a set of utterances, and our goal is to identify which ones were uttered by the same person. To aid the forming of the desired kinds of clusters, we applied pre-processing steps such as feature selection and Principal Component Analysis (PCA). However, even after these steps it is not trivial to decide which clustering method to apply. Besides the standard algorithms of Hard *c*-means (HCM) and Fuzzy *c*-means (FCM), we tested three further variations of FCM; among them, the fuzzy-possibilistic product partition *c*-means (FPPPCM) proved to be the most effective one, achieving a purity score of 83.9%, which is far above the baseline value of 46.9%.

Acknowledgement

This publication was supported by the Hungarian Government by OTKA under grant no. PD103921.

References

[1] Rabiner, L., Juang, B. H.: Fundamentals of Speech Recognition. Prentice Hall (1993)

[2] Tóth, L., Tarján, B., Sárosi, G., Mihajlik, P.: Speech Recognition Experiments with Audiobooks. Acta Cybernetica 19(4), 695-713 (2010)

[3] Ondáš, S., Juhár, J., Pleva, M., Lojka, M., Kiktová, E., Sulír, M., Čižmár, A., Holcer, R.: Speech Technologies for Advanced Applications in Service robotics. Acta Polytechnica Hungarica 10(5), 45-61 (2013)

[4] Tóth, S., Sztahó, D., Vicsi, K.: Speech Emotion Perception by Human and Machine. Proceedings of COST Action. pp. 213-224, Patras, Greece (2012)

[5] Gosztolya, G., Grósz, T., Busa-Fekete, R., Tóth, L.: Detecting the Intensity of Cognitive and Physical Load using AdaBoost and Deep Rectifier Neural Networks. Proceedings of Interspeech. pp. 452-456, Singapore (Sep 2014)

[6] Dobry, G., Hecht, R., Avigal, M., Zigel, Y.: Supervector Dimension Reduction for Efficient Speaker Age Estimation based on the Acoustic Speech Signal. IEEE Trans. Audio, Speech and Language Processing 19(7), 1975-1985 (2011)

[7] Räsänen, O., Pohjalainen, J.: Random Subset Feature Selection in Automatic Recognition of Developmental Disorders, Affective States, and Level of Conflict from Speech. Proceedings of Interspeech. pp. 210-214, Lyon, France (Sep 2013)

[8] Gosztolya, G., Busa-Fekete, R., Tóth, L.: Detecting Autism, Emotions and Social Signals using AdaBoost. Proceedings of Interspeech. pp. 220-224, Lyon, France (Aug 2013)

[9] Gosztolya, G.: Conflict Intensity Estimation from Speech using Greedy Forward-Backward Feature Selection. Proceedings of Interspeech. Dresden, Germany (2015)

[10] Schuller, B., Steidl, S., Batliner, A., Epps, J., Eyben, F., Ringeval, F., Marchi, E., Zhang, Y.: The Interspeech 2014 Computational Paralinguistics Challenge: Cognitive & Physical Load. Proceedings of Interspeech (2014)

[11] Gosztolya, G.: Estimating the Level of Conflict Based on Audio Information using INVERSE Distance Weighting. Acta Universitas Sapientiae, Electrical and Mechanical Engineering (2014)

[12] Kaya, H., Özkaptan, T., Salah, A. A., Gürgen, F.: Canonical Correlation Analysis and Local Fisher Discriminant Analysis-based Multi-View

Acoustic Feature Reduction for Physical Load Prediction. Proceedings of Interspeech, pp. 442-446, Singapore (Sep 2014)

[13] van Segbroeck, M., Travadi, R., Vaz, C., Kim, J., Black, M. P., Potamianos, A., Narayanan, S. S.: Classification of Cognitive Load from Speech using an i-vector Framework. Proceedings of Interspeech. pp. 671-675, Singapore (Sep 2014)

[14] Ajmera, J., Wooters, C.: A Robust Speaker Clustering Algorithm. Proceedings of ASRU. pp. 411-416 (2003)

[15] Yu, K., Jiang, X., Bunke, H.: Partially Supervised Speaker Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(5), 959-971 (2012)

[16] Han, K. J., Narayanan, S. S.: Agglomerative Hierarchical Speaker Clustering using Incremental Gaussian Mixture Cluster Modeling. Proceedings of Interspeech. pp. 20-23 (2008)

[17] Steinhaus, H.: Sur la division des corps materiels en parties. Bull. Acad. Pol. Sci. C1 III. (IV), 801-804 (1956)

[18] Bezdek, J. C.: Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, New York (1981)

[19] Pal, N. R., Pal, K., Keller, J. M., Bezdek, J. C.: A Possibilistic Fuzzy *c*-means Clustering Algorithm. IEEE Trans. Fuzzy Syst. 13, 517-530 (2005)

[20] Szilágyi, L., Szilágyi, S. M.: Generalization Rules for the Suppressed Fuzzy *c*-means Clustering Algorithm. Neurocomputing 139, 298-309 (2014)

[21] Devijver, P., Kittler, J.: Pattern Recognition, a Statistical Approach. Prentice Hall (1982)

[22] Brendel, M., Zaccarelli, R., Devillers, L.: A Quick Sequential Forward Floating Feature Selection Algorithm for Emotion Detection from Speech. Proceedings of Interspeech. pp. 1157-1160, Makuhari, Japan (2010)

[23] Jolliffe, I.: Principal Component Analysis. Springer-Verlag (1986)

[24] Busa-Fekete, R., Kocsor, A.: Locally Linear Embedding and its Variants for Feature Extraction. Proceedings of SOFA. pp. 216-222 (2005)

[25] Krishnapuram, R., Keller, J. M.: A Possibilistic Approach to Clustering. IEEE Trans. Fuzzy Syst. 1, 98-110 (1993)

[26] Fan, J. L., Zhen, Z. W., Xie, W. X.: Suppressed Fuzzy *c*-means Clustering Algorithm. Patt. Recogn. Lett. 24, 1607-1612 (2003)

[27] Szilágyi, L.: Fuzzy-Possibilistic Product Partition: a Novel Robust Approach to *c*-means Clustering. Proceedings of MDAI. pp. 150-161, Changsha, China (Jul 2011)

[28] Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)

[29] Todd, S. C., Tóth, M. T., Busa-Fekete, R.: A Matlab Program for Cluster Analysis using Graph Theory. Computers & Geosciences 36(6), 1205-1213 (2009)

[30] Endo, Y., Kinoshita, N., Iwakura, K., Hamasuna, Y.: Hard and Fuzzy *c*-means Algorithms with Pairwise Constraints by Non-Metric Terms. Proceedings of MDAI. pp. 145-157, Tokyo, Japan (Oct 2014)

[31] Schuller, B., Friedmann, F., Eyben, F.: The Munich Biovoice Corpus: Effects of Physical Exercising, Heart Rate, and Skin Conductance on Human Speech Production. Proceedings of LREC. pp. 1506-1510, Reykjavik, Iceland (2014)

[32] Eyben, F., Wöllmer, M., Schuller, B.: Opensmile: The Munich Versatile and Fast Open-Source Audio Feature Extractor. Proceedings of ACM Multimedia. pp. 1459-1462 (2010)