4th International Conference on Automation, Control Engineering and Computer Science (ACECS - 2017)
Proceedings of Engineering and Technology – PET
Vol.19, pp.62-66

# Phoneme recognition from tongue and lips forms using SVM, SOM and FCM techniques

Aouatif Bencheikh [1], Abdeljabbar Cherkaoui[1]
[1] Laboratory of innovative technologies,
National School of Applied Sciences in Tangier, Morocco
Bencheikh.aouatif@gmail.com , cherkaoui.lti@gmail.com

**Abstract**: This work is part of our project entitled: "Reconstitution speech by ultrasound imaging and video of the vocal apparatus: towards a silent spoken communication". Our objective is to produce a normally articulate speech and not vocalized, from the vocal apparatus activity during a silent articulation. The system to conceive is based on tongue and lips images in silent articulation. So, it becomes necessary to conduct the recognition based on smaller speech units (typically phonemes).

In this paper, we propose an approach of phoneme recognition from tongue and lips forms. We employ image processing techniques to extract features from images of tongue and lips forms. Then, we evaluate these extracted features using clustering and classification algorithms. Thus, we study the behavior of the SVM (Support vector machine) classifier, and the SOM (Self-Organizing Map) and the FCM (Fuzzy C-Means) methods of clustering; according to their mathematical parameters.

In this paper, we extract features from tongue and lips images. Then, we apply classification and clustering methods to analyze these features. Thus, we conduct a comparison between these methods based on parameters setting, in order to select the most suitable method for our study.

The rest of this paper is organized as follow. In the first section, we present the database matching French language phonemes. In the second section, we expose the SVM classification method [1][2-5] and the mathematical model of this method. In the third section, we present the SOM and FCM clustering methods[6-11]. In the fourth section, we discuss methods parameters setting and the reached results. Then, we conclude and we give our perspectives.

## I. database presentation :

The database is a set of images that matching phonemes of French language:

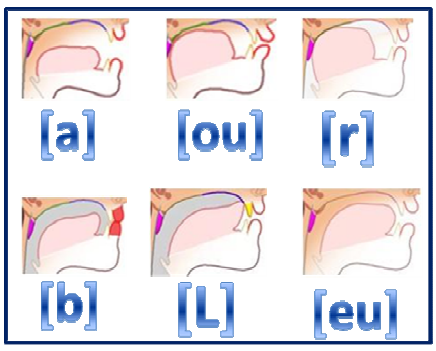


***Fig1 :*** *Example of database images*

## II. Classification method presentation: *Support Vector Machine (*SVM)

The basic concept of SVM is to bring the problem of discrimination to a problem of finding an optimal hyper plane. Two ideas or tricks achieve this goal:

-The first is to define the hyperplane as a solution to an optimization problem with constraints whose objective function is expressed only by using scalar products between vectors and in which the number of constraints "active" or support vector control model complexity.

-searching nonlinear separators surfaces is obtained by the introduction of a kernel function. in the scalar product implicitly inducing non-linear transformation of data to an intermediate space of larger dimension.

**Linear SVM:**

Basic concepts: Hyperplane, margin and support vector

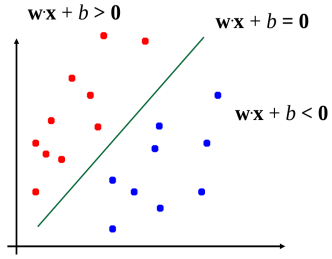


***Fig2:*** *linear classification*

Hyperplane classifies the data correctly and that is as far as possible of all the examples.

$$w..x + b \qquad (1)$$

Margin: distance from the nearest point to the hyperplane

$$d(x) = \frac{|w..x + b|}{\|w\|} \qquad (2)$$

Maximize margin based minimize ||w || under constraints

Support vectors are the closest points of the separating hyperplane. We obtain the minimization problem under constraints:

Primal problem:

$$\begin{cases} \min \frac{1}{2}\|w\|^2 \\ \forall i, y_i (w.x_i + b) \geq 1 \end{cases} \qquad \textbf{(3)}$$


ISSN 2356-5608

Dual problem:

$$\begin{cases} \max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i \bullet x_j \\ \forall_{i,\alpha_i} \geq 0 \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases} \quad \textbf{(4)}$$

**Non-linear SVM:**

The data space may still be immersed in a space of larger size in which data can be linearly separated by the transformation:

**(x, y) → (x, y, x · y)**

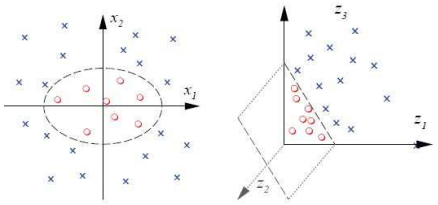


***Fig3:*** *non- linear classification*

Resolution SVM is based only on the scalar product <xi, xj> between the input vectors.

If the training data are immersed in a space of larger dimension via the transformation: $\Phi : x \rightarrow \phi(x)$ (5)

the scalar product becomes: $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ (6)

K: Kernel function.

The types of kernels [12] existing in non-linear separation are:

**Linear:** $k(x, y) = x^T y + c$ (7)

**Polynomial:** $k(x, y) = (\alpha x^T y + c)^d$ (8)

**Gaussian (or radial basis):**

$$k(x, y) = \exp\left( - \frac{\|x - y\|^2}{2\sigma^2} \right) \quad (9)$$

Alternatively, it could also be implemented using:

$$k(x, y) = \exp(-\gamma \|x - y\|^2) \quad (10)$$

**Hyperbolic tangent (or sigmoid)**

$$k(x, y) = \tan gh(\alpha x^T y + c) \quad (11)$$

## III. Clustering methods presentation: SOM, FCM

### *III-1.Self organizing map (SOM)*

In the beginning of the learning, all weights (wi) of the second layer's neurons are set to random values. After that, some input-vector from the set of learning vectors is selected and set to the input of the neural network. At this step, the differences between the input vector and all neurons vectors are calculated as follows (12):

$$\begin{cases} D_{ij} = \left| X^l - W_{ij} \right| = \sqrt{(x_1 - w_{ij1})^2 + \ldots + (x_n - w_{ijn})^2} \quad \textbf{(12)} \\ w_i = w_{ij}\ , \\ i = 1, 2, \ldots, k_x \text{ (kx is the number of row)}, \\ j = 1, 2, \ldots, k_y \text{ (ky is the number of column)}. \end{cases}$$

Next, SOM algorithm will search for the winner neuron using the minimum distance (best matching unit, BMU). BMU is calculated as follows (13):

$$BMU = \text{argmin} \|x(t) - wi(t)\| \quad (13)$$

To increase the similarity with the input vector, weights are adjusted after obtaining the winning neuron. The rule for updating the weight vector is given by (14):

$$W_{ij}(t+1) = W_{ij}(t) + \alpha(t) h_{ij}(t) (X^l(t) - W_{ij}(t)) \quad (14)$$

$\alpha(t)$ : is a learning rate.

$h_{ij}(t)$ : is a neighborhood function and t is the order number of a current iteration.

The learning rate is a training parameter that controls the size of weight vector in learning of SOM. There are many learning rate functions while linear, inverse of time and power series are mostly used in SOM. Thus, Linear, inverse of time and power series are defined in (15), (16) and (17), respectively:

$$\alpha(t) = \alpha(0).\frac{1}{t} \quad (15)$$

$$\alpha(t,T) = \alpha(0).\left(1 - \frac{t}{T}\right) \quad (16)$$

$$\alpha(t,T) = \alpha(0).e^{\frac{t}{T}} \quad (17)$$

T : is the number of iterations.

t : is the order number of a current iteration.

Neighborhood function determines the rate of change of the neighborhood around the winner neuron. Neighborhood function influences the training result of SOM procedure. Therefore, it is important to choose the proper neighborhood function with the data set. Same as learning rate, there are many functions but bubble and Gaussian are widely used in SOM. Bubble function is a constant function while Gaussian function is decreasing function in the defined neighborhood of the winner neuron. Bubble and Gaussian are defined in (18) and (19), respectively.

$$h_{ij} = \begin{cases} \alpha(t), (i, j) \in N_c \\ 0, (i, j) \notin N_c \end{cases} \quad (18)$$

$$hij = \alpha(t).e^{\left( \frac{-\|R_c - R_{ij}\|^2}{2(\eta_{ij}(t))^2} \right)} \quad (19)$$


ISSN 2356-5608

**Nc :** is the index set of neighbor nodes around the node with indices c.

**ηij :** represents the neighbor rank between nodes wc and wij (the radius of neighborhood which is determined the number of neighborhood for SOM procedure).

**Rc** and **Rij** : Two-dimensional vectors which include indexes of wc and wij (number of rows and columns).

### *III-1-3.* Performance measure of SOM

The SOM [13] can be computed using either the sequential training algorithm or the batch training algorithm.

Quantization error (QE) is a technique to measure the quality of SOM and the performance of this type of neural networks. It is computed from the average distance of input vectors, x to the weight vector on the winner node $w_i^c$ of the BMU. A SOM with lower average error is more accurate than a SOM with higher average error [17]. Quantization error is calculated by (20):

$$QE = \frac{\sum_{i=1}^{N} \left\| x_i - w_i^c \right\|}{N} \qquad (20)$$

### *III-2.Fuzzy C-means (FCM)*

FCM [14] algorithm works by assigning membership to each data point corresponding to each cluster center on the basis of distance between the cluster center and the data point. More the data is near to the cluster center more is its membership towards the particular cluster center. Actually, summation of membership of each data point should be equal to one. After each iteration membership and cluster centers are updated according to the following formula (21):

$$v_j = \frac{(\sum_{i=1}^{n} (\mu_{ij})^m \; x_i)}{(\sum_{i=1}^{n} (\mu_{ij})^m)}; \forall j = 1,2,...c \quad \textbf{(21)}$$

Where $\mu_{ij}$ is defined by **(22) :**

$$\mu_{ij} = 1 \Big/ \sum_{k=1}^{c} (d_{ij}/d_{ik})^{(2/m-1)} \qquad \textbf{(22)}$$

**n :** is the number of data points.

**vj :** represents the jth cluster center.

**m :** is the fuzziness index m € [1, ∞].

**c :** represents the number of cluster center.

**µij :** represents the membership of ith data to jth cluster center.

**dij :** represents the Euclidean distance between ith data and jth cluster center.

So, the main objective of fuzzy c-means algorithm is to minimize (23):

$$J(U,V) = \sum_{i=1}^{n} \sum_{j=1}^{c} (\mu_{ij})^m \left\| x_i - v_j \right\|^2 \qquad (23)$$

Where: '||xi – vj||' is the Euclidean distance between ith data and jth cluster center.

## IV. Setting and results

### *IV-1.parameters setting of SVM method:*

***Setting and Results:***

We present best results of simulations for each type : linear, polynomial sigmoid and radial basis SVM according to their mathematical parameters:

*Linear SVM:*

| **Linear SVM** | | | | | | |
|---|---|---|---|---|---|---|
| **Parameters** | | | | | **recognition rate** | |
| c | $w_i$ | s | b | n | 70% :training 30%:test | 80%: training 20%:test |
| 1 | 1.2 | 0 | 1 | - | 80.56% | **83.33%** |

*Table1: parameters setting for linear SVM*

*Sigmoid SVM:*

| ***Sigmoid SVM*** | | | | | | |
|---|---|---|---|---|---|---|
| ***Parameters*** | | | | | ***recognition rate*** | |
| ***s*** | *c* | *g* | *r* | *n* | *70% :training 30%:test* | *80%: training 20%:test* |
| ***0*** | *1* | *1* | *0* | - | *16.67%* | ***16.67%*** |

*Table2: parameters setting for sigmoid SVM*

*Polynomial SVM:*

| **Polynomial SVM** | | | | | | |
|---|---|---|---|---|---|---|
| **Parameters** | | | | | **recognition rate** | |
| s | c | d | r | n | 70% :training 30%:test | 80%: training 20%:test |
| 0 | 1.5 | 2 | 0 | - | 77.78% | **87.50%** |

*Table3: parameters setting for polynomial SVM*

*Radial basis SVM:*

| **Radial basis SVM** | | | | | |
|---|---|---|---|---|---|
| **Parameters** | | | | **recognition rate** | |
| s | c | g | n | 70% :training 30%:test | 80%: training 20%:test |
| 0 | 2 | 1.5 | - | 80.56% | **91.67%** |

*Table4: parameters setting for radial basis SVM*


ISSN 2356-5608

### *Results interpretation:*

We have tested many cases of experimentations using SVM. After studying the cases, we notice that the C parameter has a big effect on accuracy than n. Also we remark that Radial basis SVM is more efficient in our case. Actually, we get 91.67% as the best classification rate, which is a an interesting rate compared to 100%.

So, we can conclude that in our study, we were able to make a setting for Support Vector Machine classification method for phoneme recognition. Thus, we get a performed accuracy using SVM, which means that our approach is modeling well phoneme recognition.

## *IV-3. Parameters setting of SOM method*

### *Setting and results:*

The tables below show the values of the parameters [15] mentioned above (with a linear Initialization)

| | | ***Neighborhood*** | |
|---|---|---|---|
| | | *Gaussian* | *Bubble* |
| **Map size** | [1 2] | 0.5320 | 0.5320 |
| | [3 3] | 0.4504 | 0.4306 |
| | [6 6] | 0.3444 | 0.3233 |
| | [9 6] | 0.3254 | 0.2834 |
| | [10 10] | 0.2662 | 0.2124 |
| | [20 20] | 0.1275 | **0.0492** |

*Table 5: quantization error for batch training phase*

| **Map Size** | **neighborhood** | | | | | |
|---|---|---|---|---|---|---|
| | gaussian | | | Bubble | | |
| | L | IOT | PS | L | IOT | PS |
| **[1 2]** | 0.5329 | 0.5318 | 0.5313 | 0.5606 | 0.5607 | 0.5636 |
| **[3 3]** | 0.4445 | 0.4510 | 0.4438 | 0.4256 | 0.4329 | 0.4257 |
| **[6 6]** | 0.3400 | 0.3851 | 0.3538 | 0.3218 | 0.3554 | 0.3278 |
| **[9 6]** | 0.3176 | 0.3661 | 0.3182 | 0.2853 | 0.3366 | 0.2853 |
| **[10 10]** | 0.2697 | 0.3359 | 0.2734 | 0.2279 | 0.3064 | 0.2355 |
| **[20 20]** | 0.1616 | 0.2879 | 0.1825 | **0.0951** | 0.2571 | 0.1344 |

(L:Linear ; IOT:Inverse of time ; PS: Power series)

*Table 6: quantization error for sequential training phase*

### *Results interpretation:*

From the figures shown above we can see very well the SOM parameters that influence the quantization error:

1-Size of map: the quantization error becomes smaller when the map becomes bigger.

2-neighborhood: we can see that the bubble neighborhood gives results better than Gaussian.

So, the parameters combination that give more efficient results to recognize phoneme from lips and tongue forms are:

-map: big size

-training: batch

-neigborhood: bubble

## *IV-4. Parameters setting of FCM method*

In this experimental section, our objective is to validate measures by calculating di□erent types of validation indexes. For this reason, we will configure our algorithm according to the FCM known parameters.

**Param.c:** The number of clusters or the initial partition matrix.

**param.m:** The weighting exponent which determines the fuzziness of the clusters. It must be given as a scalar greater or equal to one. The default value of m is 2.

**Param.e:** The termination tolerance of the clustering method. The default setting is 0.001

The table below summarizes the setting of FCM applied to our database (representing 17 phonemes of French language).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| parameters | **m** | 2 | 3 | 4 | 10 | 10 | 10 |
| | **c** | 4 | 4 | 6 | 10 | 14 | 17 |
| | **e** | 0.001 | 0.001 | 0.001 | 0.001 | 1e-6 | 1e-6 |
| Validity indexes | **Pc** | 0.5781 | 0.5781 | 0.4992 | 0.4723 | 0.4880 | 0.4837 |
| | **Ec** | 0.7991 | 0.7991 | 1.0437 | 1.2451 | 1.2996 | 1.3651 |
| | **SC** | 1.7277 | 1.7277 | 1.3597 | 1.0775 | 0.7925 | 0.7594 |
| | **S** | 0.0187 | 0.0187 | 0.0152 | 0.0133 | 0.0111 | 0.0108 |
| | **XB** | 4.4142 | 4.4142 | 3.2983 | 3.5232 | 2.9627 | 2.2755 |
| | **DI** | 0.0271 | 0.0271 | 0.0175 | 0.0566 | 0.0716 | 0.0559 |
| | **ADI** | 0.0034 | 0.0034 | 0.0128 | 0.0024 | 2.1713e-05 | 5.8503e-04 |

### *Results interpretation:*

From this experimentation, we can notice that the variation of FCM parameters clearly influences the quality of results (presented by validity indexes). By changing the parameter values one by one, we remark that a big number of clusters give the best validity indexes.

We recall that the number of phonemes (that represent classes) represent our database, so we use17 phonemes; however French language contain 36 phonemes. Thus, if we expand our database the results will be more efficient.

## Conclusion

In this paper, we presented a comparative study between three popular methods: SVM, SOM and FCM according to their mathematical parameters, in order to make a good phoneme recognition based on lips and tongue forms.

We firstly processed by extracting characteristics from input images based on HOG method. Then we applied classification and clustering methods on the same database.

The experimentation results obtained give us an idea about these methods behavior. So, in our case we can conclude that SVM classification method has more advantages than FCM and SOM clustering methods. Also, our proposed approach gives performed classification results.

This work represents a first and important step to start continuous silent speech in real time study, based on lips and tongue images. In future works, we will broaden our database and we will test other image processing techniques for recognition phonemes.


ISSN 2356-5608

## References:


*[1]Cortes, C.; Vapnik, V. (1995). "Support-vector networks". Machine Learning 20 (3): 273–297*

*[2] J.Weston , C.Watkins support vector machine for multi-class pattern recognition(1999)*

*[3]A.David , B.Lerner Support vector machine based image classification for genetic syndrome diagnosis(2004)*

*[4] G.Anthony, H.Greg,M.Tshilidzi Classification of Images Using Support Vector Machines(2007)*

*[5]G. Du et al., Effective and efficient Grassfinch kernel for SVM classification and its application to recognition based on image set, Chaos, Solitons and Fractals (2016)*

*[6]Kohonen, Teuvo (1982). "Self-Organized Formation of Topologically Correct Feature Maps". Biological Cybernetics 43 (1): 59–69.*

*[7] C. Roncancio Valencia. J. Gomez Garcia-Bermejo. E. Zalama Casanova Combined Gesture-Speech Recognition and Synthesis Using Neural Networks (2008)*

*[8] Dunn, J. C. (1973-01-01). "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters". Journal of Cybernetics 3 (3): 32–57.*

*[9] Bezdek, James C. (1981). Pattern Recognition with Fuzzy Objective Function Algorithms.*

*[10] Valafar F. Pattern recognition techniques in microarray data analysis. Annals of the New York Academy of Sciences. 2002 Dec 1;980(1):41-64.*

*[11] Ahmed, Mohamed N.; Yamany, Sameh M.; Mohamed, Nevin; Farag, Aly A.; Moriarty, Thomas (2002). "A Modified Fuzzy C-Means Algorithm for Bias Field Estimation and Segmentation of MRI Data" (PDF). IEEE Transactions on Medical Imaging 21 (3): 193–199.*

*[12] Hofmann, T., B. Schölkopf, and A. J. Smola. "Kernel methods in machine learning." Ann. Statist. Volume 36, Number 3 (2008), 1171-1220.*

*[13] Kohonen, Teuvo; Honkela, Timo (2007). "Kohonen Network".*

*Scholarpedia.*

*[14] J. C. Bezdek, "Pattern Recognition with Fuzzy Objective Function Algorithms", New York: Plenum Press, 1981.*

*[15] J.Vesanto, J.Himberg, E.Alhoniemi , J.Parhankangas "Self-organizing map in Matlab: the SOM Toolbox" In Proceedings of the Matlab DSP Conference 1999, Espoo, Finland, pp. 35−40, 1999*



ISSN 2356-5608