Clustering Variable Length Sequences by Eigenvector Decomposition using HMM

Fatih Porikli

Mitsubishi Electric Research Laboratories, Cambridge MA 02139, USA,
fatih@merl.com

Abstract. We present a novel clustering method using HMM parameter space and eigenvector decomposition. Unlike the existing methods, our algorithm can cluster both constant and variable length sequences without requiring normalization of data. We show that the number of clusters governs the number of eigenvectors used to span the feature similarity space. We are thus able to automatically compute the optimal number of clusters. We successfully show that the proposed method accurately clusters variable length sequences for various scenarios.

1 Motivation

Although many algorithms exist for unsupervised classification of patterns into clusters, most of these methods require the data space X consists of 'identical length' data points (feature vectors) $x_i = (x_{i1}, ..., x_{iN})$ where N is the dimension of the data space, i.e. $X : \mathcal{R}^N$. Such algorithms include the ordinary implementations of decision trees, neural nets, Bayesian classifiers, ML-estimators, support vector machines, Gaussian mixture models, k-means, and hierarchical approaches, self-organizing maps, etc [4].

However not all classification problems can be formulated into a data space that contains only equal length feature vectors. For instance, lets consider the following scenarios:

Example 1. A data space contains different shapes. We compute a sequence of boundary coordinates for each shape by starting from a certain point on the boundary. Then we obtain sequences such as $s_i = ((x_{i1}, y_{i1}), ..., (x_{iN_i}, y_{iN_i}))$ where (x_{ij}, y_{ij}) is the coordinate of the j^{th} boundary point for the i^{th} shape. In this case, the length of the sequences are not necessarily same since the length of the boundaries may be different, e.g. it is possible that $N_1 \neq N_2$.

Example 2. A basket contains unknown number of not necessarily identical balls labeled as `a` and `b`. At a random time instant we start drawing balls from this basket. We keep record of the symbols on the balls. Then we stop drawing balls at a random time instant. By repeating the experiment we obtain sequences as

$s_1 =$ `abababababababababababababababa`
$s_2 =$ `abababababababababa`
$s_3 =$ `bbbbbbbbbbbbbbbbbbbbbbbbbbbbbb`, etc.

Note that the length of these sequences are different. This process is analogous to chopping a DNA into genes. Finding the common and divergent patterns from a big pool of protein genomes, which come in various sizes, is a challenge of current bioinformatics and requires clustering of variable length sequences.

We will refer the type of the data in these examples as variable length sequences. One way to adapt some of the above scenarios for ordinary classification, which processes constant length sequences, is to normalize the length of the feature vectors. Although commonly used due to its simplicity, normalization of the feature vector length (either by sampling or interpolation) causes severe degradation and aliasing. Besides, some clustering approaches assume that they can compute a centroid and then compare the data points with this centroid, as in k-means. It is not possible to obtain such a centroid for variable length data.

Thus, we propose a clustering algorithm that can classify variable length sequences. Our algorithm also estimates the optimum number of clusters and does not require normalization of the length of the feature vectors. Instead of working directly on the initial values, we transfer the sequences into a parameter space using Hidden Markov Models (HMM), which captures the probabilistic transition properties of sequential data.

In this work, we concentrate on the discrete label sequences. Using the parameter space representations, we compute an affinity matrix that shows the similarity of a given pair of sequences. Then we decompose the affinity matrix into a series of subspaces spanned by the eigenvectors corresponding to the largest eigenvalues. We threshold the decomposed values and partition clusters using simple connected component analysis. For each decomposition, we calculate a validity score indicating the fitness of the current clusters to the data. We determine the optimum number of clusters using the validity score. We give a flow diagram of the method in fig. 1.

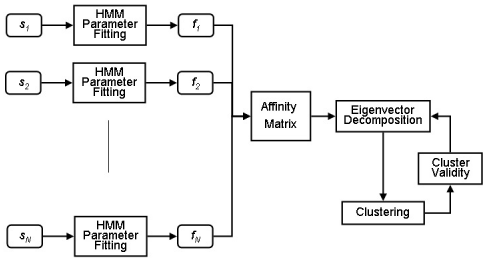

Fig. 1. Flow diagram of clustering sequences.

In the next section, we explain HMM's and affinity matrix. In section 3, we present eigenvector decomposition. In the following sections, we give details of the clustering and discuss simulations.

2 Parameter Space by HMM

We project each sequence s_i into the parameter space that is characterized by a set of HMM parameters. HMM's are richer representations of time series. An HMM is a probabilistic model composed of a number of interconnected states, each of which emits an observable output. A discrete hidden Markov model is defined by a set of states and an alphabet of output symbols [6]. Each state is characterized by two probability distributions: the transition distribution over states and the emission distribution over the output symbols. A random source described by such a model generates a sequence of output symbols as follows: at each time step the source is in one state, and after emitting an output symbol according to the emission distribution of the current state, the source jumps to a next state according to the transition distribution of its current state. Since the activity of the source is observed indirectly, through the sequence of output symbols, and the sequence of states is not directly observable, the states are said to be hidden.

An K-state $\{S_1, S_2, ..., S_K\}$, discrete HMM is represented by:

1. A set of prior probabilities $\pi = \{\pi_i\}$ where $\pi_i = P(q_1 = S_i), 1 \leq i \leq K$.
2. A set of state transition probabilities $H = \{h_{ij}\}$, where $h_{ij} = P(q_{t+1} = S_j | q_t = S_i), 1 \leq i, j \leq K$.
3. A set of output distributions $B = \{b_{ij}\}$, where $b_{ij}(y) = P(O_{t+1} = y | q_t = S_i, q_{t+1} = j), 1 \leq i, j \leq K$.

where q_t and O_t are the state and observation respectively at time t. It is common to denote the an M-mixture of HMM's by (H_m, B_m, π_m), $1 \leq m \leq M$. For discrete HMM, algorithms exist for: 1) computing the probability of observing a sequence, given a model, 2) finding the state sequence that maximizes the probability of the given sequence, when the model is known (the Viterbi algorithm), 3) inducing the HMM that maximizes (locally) the probability of the given sequence (the BaumWelch algorithm, an expectationmaximization algorithm).

For each sequence, we fit a HMM (a discrete model in case the sequence components are labels, a continuous model in case the components are real numbers that reflect certain proportional properties, e.g. magnitude, coordinate, etc). The number of states K, number of models M, and the HMM topology (left-to-right) are assigned same for each sequence. This enables us to compute parameter space distances using the model state transition, observation, and prior matrix differences [8].

Definition 1. *A feature f_i is a set of real numbers that correspond the HMM parameters of a sequence s_i for a given number of states K, number of mixtures M, and left-to-right topology using the sequence as an observation ($f_i : s_i \rightarrow (H_m, B_m, \pi_m)_i$).*

The problem of estimating the correct number of clusters is a difficult one: a full Bayesian solution for obtaining the posterior probability on M, requires a complex integration over the HMM parameter space, as well as knowledge about

the priors on the mixture parameters and about the priors on M itself. Often this integration cannot be solved in closed form, and Monte-Carlo methods and other approximation methods are used to evaluate it. However, these methods are computationally intensive [1].

Now, we can compute our affinity matrix. Given two sequences s_i, s_j, we determine the probability that feature set is generated by s_j and feature set f_j is generated by s_i. This probability indicates the mutual 'fitness' of the given sequences to corresponding HMM's. Thus, the affinity matrix represents the similarity of two sequences. The elements a_{ij} of A are equal to

$$a_{ij} = e^{-d(s_i,s_j)/2\sigma^2} \tag{1}$$

where the distance is defined as

$$d(s_i, s_j) = |P(s_i|f_i) + P(s_j|f_j) - P(s_i|f_j) - P(s_j|f_i)|\,. \tag{2}$$

and σ^2 is a scaler. The affinity matrix components will have values close to 1 if the corresponding sequences fit well to each other's models, and close to 0 otherwise. Note that similarity matrix $A \in \mathcal{R}^{n\times n}$ is a real semi-positive symmetric matrix, thus $A^T = A$.

Next, we explain the details of the eigenvector decomposition process.

3 Eigenvector Decomposition

The decomposition of a square matrix into eigenvalues and eigenvectors is known as eigenvector decomposition. For the affinity matrix A there are n eigenvalues λ with associated eigenvectors $\mathbf{v}$ which satisfy $A\mathbf{v} = \lambda\mathbf{v}$. To find these eigenvalues, we rewrite the previous equation as $(A-\lambda I)\mathbf{v} = 0$ and determinant is computed $det(A - \lambda I) = 0$.

Let $V \equiv [\mathbf{v}_1\ \mathbf{v}_2\ ..\ \mathbf{v}_n]$ be a matrix formed by the columns of the eigenvectors. Let D be a diagonal matrix $diag[\lambda_1, \lambda_2, .., \lambda_n]$. Lets also assume $\lambda_1 \geq \lambda_2 \geq ..\lambda_n$. Then the eigenvalue problem becomes

$$AV = [A\mathbf{v}_1\ ..\ A\mathbf{v}_n] = [\lambda_1\mathbf{v}_1\ ..\ \lambda_n\mathbf{v}_n] = VD \tag{3}$$

and $A = VDV^{-1}$. Since A is symmetric, the eigenvectors corresponding to distinct eigenvalues are real and orthogonal $VV^T = V^TV = I$, which implies $A = VDV^T$.

Iterative Eigenvector Computation The main idea behind iterative computation is the following. Suppose we have some subspace $\mathcal{K}$ of dimension k, over which the projected matrix A has Ritz [7] value θ_k and a corresponding Ritz vector $\mathbf{u}_k$. Let us assume that an orthogonal basis for $\mathcal{K}$ is given by the vectors $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k$ (already determined eigenvectors).

Quite naturally the question arises how to expand the subspace in order to find a successful update for $\mathbf{u}_k$, which will become v_{k+1}. To that end we compute

the defect $\mathbf{r} = A\mathbf{u}_k - \theta_k\mathbf{u}_k$. Then as in [3], we compute $\widetilde{\mathbf{z}}$ from $(D - \theta_k I)\widetilde{\mathbf{z}} = \mathbf{r}$, where D is the diagonal matrix of A as defined above. The vector $\widetilde{\mathbf{z}}$ is made orthogonal to $\mathcal{K}$, and the resulting vector is chosen as the new $\mathbf{v}_{k+1}$ by which $\mathcal{K}$ is expanded. This method find the largest eigenvalues in absolute value. The matrix $(D-\theta_k I)^{-1}$ can be viewed as a preconditioner for the vector $\mathbf{r}$. Although it is tempting to use this preconditioner as an approximation for $(A - \theta_k I)$, it would not lead to an expansion of our search space. To avoid this stagnation, we concentrate on the kth approximation $\mathbf{u}_k$ of the eigenvector $\mathbf{v}$, where $\mathbf{u}_k$ is normalized $||\mathbf{u}_k|| = 1$. The residual $\mathbf{r} = A\mathbf{u}_k - \theta_k\mathbf{u}_k$ is orthogonal to $\mathbf{u}_k$ because $\theta_k = \mathbf{u}_k^T A\mathbf{u}_k$ is the Ritz value associated with $\mathbf{u}_k$. We project the eigenvalue problem $Av = \lambda\mathbf{v}$ on span($\mathbf{u}_k$), and on its orthogonal complement. This leads to two coupled equations for λ and the complement $\mathbf{z}$ of $\mathbf{v}$ orthogonal to $\mathbf{u}_k$: $\lambda = u_k^T A(\mathbf{u}_k + \mathbf{z})$ and $\mathbf{z} \perp \mathbf{u}_k$, $\quad (I - \mathbf{u}_k\mathbf{u}_k^T)(A - \lambda I)(I - \mathbf{u}_k\mathbf{u}_k^T)\mathbf{z} = -\mathbf{r}$. Since λ is unknown, we cannot compute optimal update $\mathbf{z}$ from $\mathbf{u}_k$. However it is reasonable to replace λ by the current approximation θ_k. Thus we obtain $\mathbf{r} \perp \mathbf{u}_k$, $\quad (I - \mathbf{u}_k\mathbf{u}_k^T)(A - \theta_k I)(I - \mathbf{u}_k\mathbf{u}_k^T)\mathbf{z} = -\mathbf{r}$ as a good correction for $\mathbf{u}_k$. Similarly, we compute the approximate solution $\widetilde{\mathbf{z}}$ using this equation, and by making $\widetilde{\mathbf{z}}$ orthogonal to search space, we obtain $\mathbf{v}_{k+1}$. Briefly, we extract an approximate eigenvalue from the search subspace, project it, solve the projected eigenvalue problem, compute the corresponding Ritz value and residual, correct the approximate eigenvector u, and expand the search subspace with the correction vector.

The above iterative prediction is used at the following clustering stage.

4 Clustering

Although eigenvector based clustering [2], [9], [5] is addressed before in the literature, to our knowledge no one has established the relationship between the optimal clustering of the data distribution and the number of eigenvectors that should be used for spanning before. Here we show that the number of eigenvectors is proportional to the number of clusters.

Let a matrix P_k be a matrix in a subspace $\mathcal{K}$ that is spanned by the columns of V such as $P_k = [\mathbf{v}_1\ \mathbf{v}_2\ ..\ \mathbf{v}_k,\ 0]$ where V is the orthogonal basis satisfies $A = VDV^T$.

Now, we define vectors $\mathbf{p}_n$ as the rows of the truncated matrix P_k as

$$P_k = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} = \begin{bmatrix} v_{11} \cdots v_{1k}\ 0 \cdots \\ v_{21} \cdots v_{2k}\ 0 \cdots \\ \vdots \qquad\qquad \vdots \\ v_{n1} \cdots v_{nk}\ 0 \cdots \end{bmatrix} \tag{4}$$

We normalize each row of matrix P_k by $p_{ij} \leftarrow p_{ij}/\sqrt{\sum_j^k p_{ij}^2}$. Then a correlation matrix is computed using the normalized rows by $C_k = P_k P_k^T$. For a given P_k, the value of p_{ij} indicates the degree of similarity between the object i and object

j. Values close to one correspond to a match whereas negative values and values close to zero suggest that objects are different. Let ϵ be a threshold that transfers values of matrix C_k to the binary quantized values of an association matrix W_k as

$$w_{ij} = \begin{cases} 1 \; c_{ij} \geq \epsilon \\ 0 \; c_{ij} < \epsilon \end{cases} \tag{5}$$

where $\epsilon \approx 0.5$. The clustering is then becomes grouping the objects that have association values equal to one $w_{ij} = 1$.

To explain why this works, remember that eigenvectors are the solution of the classical extremal problem $\max \mathbf{v}^T A\mathbf{v}$ constrained by $\mathbf{v}^T\mathbf{v} = 1$. That is, find the linear combination of variables having the largest variance, with the restriction that the sum of the squared weights is 1. Minimizing the usual Lagrangian expression $\mathbf{v}^T A\mathbf{v} - \lambda(\mathbf{v}^T\mathbf{v} - 1)$ implies that $A\mathbf{v} = \lambda\mathbf{v}$. Thus, $\mathbf{v}$ is the eigenvector with the largest eigenvalue.

When we project the affinity matrix columns on the eigenvector $\mathbf{v}_1$ with the largest eigenvalue and span $\mathcal{K}_1$, the distribution of the a_{ij} will have the maximum variance therefore the maximum separation. Keep in mind that a threshold operation will perform best if the separation is high. To this end, if the distribution of values have only two distinct classes then a balanced threshold passing through the center will divide the points into two separate clusters. With the same reasoning, the eigenvector $\mathbf{v}_2$ with the second largest eigenvalue, we will obtain the basis vector that gives the best separation after normalizing the projected space using the $\mathbf{v}_1$ since $\mathbf{v}_1 \perp \mathbf{v}_2$. It is important to note that, each additional eigenvector enables us to divide the space into an extra cluster. Thus, we conclude that;

Lemma 1. *The number of largest eigenvalues (in absolute value) to span the subspace is one less than the number of clusters.*

As opposed to using only the largest or first and second largest eigenvectors (also the generalized second minimum which is the ratio of the first and the second depending the definition of affinity), the correct number of eigenvectors should be selected with respect to the optimum cluster number.

After each eigenvalue computation of matrix A in the iterative algorithm, we compute a validity score α_k using the clustering results as

$$validity : \alpha_k = \sum_{c}^{k} \frac{1}{N_c} \sum_{i,j \in Z_c} p_{ij} \tag{6}$$

where Z_c is set of objects included in the cluster c, N_c number of objects in Z_c. The validity score gets higher values for the better fits. Thus, by evaluating the local maxima of this score we determine the correct cluster number automatically. Thus, we answer one important question of clustering; "what should be the total cluster number?"

The values of the thresholds should still be computed. We obtained projections that gives us the maximum separation but we did not determine the degree

of separation i.e. maximum and minimum values of projected values on the basis vectors. For convenience, we normalize the projections i.e. the *rows* of current projection matrix (V_k) as $\mathbf{p}^T\mathbf{p} = 1$ and then compute the correlation $V_k^T V_k$. Correlation will make rows that their projections are similar to get values close to 1 (equal values will give exactly 1), and dissimilar values to 0. By maximizing the separation (distance) between the points in different clusters on an orthonormal basis, we pushed for the orthogonality of points depending their clusters; $\mathbf{p}_i\mathbf{p}_j \approx 1$ if they are in the same cluster, and $\mathbf{p}_i\mathbf{p}_j \approx 0$ if they are not in the same cluster.

As a summary, the clustering for a given maximum cluster number k^* includes

1. Compute A, approximate eigenvectors using Ritz values $\lambda_k \simeq \theta_k$, find eigenvectors v_k for $k = 1, .., k^*$,
2. Find $P_k = V_k V_k^T$ and Q_k for $k = 1, .., k^*$,
3. Determine clusters and calculate α_k,
4. Compute $\alpha' = d\alpha/dk$ and find local maxima.

The maximum cluster number k^* does not affect the determination of the fittest cluster; it only limits the maximum number of possible clusters that will be searched.

5 Experiments and Discussion

We simulated the proposed method using several label sequences as given in fig. 2. We conducted the following evaluations:

Language Discrimination: We generated three sequences of random integers that are uniformly distributed in the range $[1 : 10]$. Then, we replaced each number in the sequence with its English and Portuguese spellings to obtain the letter sequences given in fig. 2-a (shown in black). Thus, each letter represents a label. We trained 4-states HMM's and applied eigenvector decomposition after computing the affinity matrix. The validity reached maximum for cluster number $k = 2$ and fig. 2-a shows the clustering results. As visible in the bottom part of fig. 2-a (blue and red clusters), the HMM's captured the dynamics of letter ordering that is intrinsic to each language and identified the language clusters accurately. We obtained similar results for different length sequences as well. In the future, we plan o represent each word using a separate label rather that using letters as labels for language related classification tasks.

Pattern Matching: We generated a total of 8 random length sequences that can be partitioned into 4 patterns using two labels (a, b) as given in fig. 2-b, left. After computing 2-state HMM's we obtained the affinity matrix given in fig. 2-c. The validity scores after iterative clustering is shown in fig. 2-d, where it reaches maximum for the cluster number $k = 4$. The corresponding clusters after eigenvector decomposition is given in fig. 2-d. The results prove that the proposed method can accurately detect pattern similarities even though the length of the sequences may differ significantly.

Random Letters: We generated random length, random distributed sequences using labels (c, d, e, f, g, h, i, j) as given in fig. 2-e (1^{st} and 3^{nd} columns). In the first column set, the first 10 sequences consist of uniformly distributed random drawings of from set (c, d, e, f), and similarly the second 10 sequences are made of (e, f, g, h), and the last 10 sequences are generated from (g, h, i, j) to allow partial overlap between each domain. The affinity matrix and validity scores are given in fig. 2-f and fig. 2-g, respectively. The maximum validity happens at $k = 3$. The second column shows the clustering results for this set. The third row in fig. 2-e shows the random label sequences that are generated using the set (e, f, g) (first 15 sequences) and a bigger inclusive set of (e, f, g, h, i) (last 15 sequences). The affinity is depicted in fig. 2-h and the validity scores in fig. 2-i. As obvious in the clustering results, the optimum cluster number is estimated accurately and the eigenvector decomposition partitioned correctly at each time.

In conclusion, the main contributions of this paper are:

- We proposed a new method to compare the variable length sequences using the HMM parameter space.
- We showed that the number of largest eigenvalues (in absolute value) to span subspace is one less than the number of clusters.
- We used the above result as a quality assessment criterion for cluster fit.

References

1. J. Alon, S. Sclaroff, G. Kollios V .Pavlovic, "Discovering clusters in motion time-series data", *Proceedings of Computer Vision and Pattern Recognition*, 2003.
2. G.L. Scott and H. C. Longuet-Higgins, "Feature grouping by relocalisation of eigenvectors of the proxmity matrix" *In Proc. British Machine Vision Conference*, 103-108, 1990.
3. G. Sleijpen and H. Van Der Vorst, "A Jacobi-Davidson iteration method for linear eigenvalue problems", *SIAM J. Matrix Anal. Appl.*, vol. 17, 401425, 1996.
4. A. K. Jain , M. N. Murty , P. J. Flynn, "Data clustering: a review", *ACM Computing Surveys (CSUR)*, 31(3), 264-323, 1999.
5. J. Shi and J. Malik. "Normalized cuts and image segmentation" *In Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 731-737, 1997.
6. L. Rabiner. "A tutorial on hidden markov models and selected applications in speech recognition", *Proceedings of IEEE*, 77(2), 257285, 1989.
7. R. B. Morgan, "Computing interior eigenvalues of large matrices" *Linear Algebra Appl.*, 154/156, 289-309, 1991.
8. P. Smyth, "Clustering sequences with Hidden Markov Models", *Book: Advances in Neural Information Processing Systems, The MIT Press, M.C. Mozer, M.I. Jordan, T. Petsche*, 648, 1997.
9. Y. Weiss, "Segmentation using eigenvectors: a unifying view", *Proceedings IEEE International Conference on Computer Vision*, 975-982, 1999.

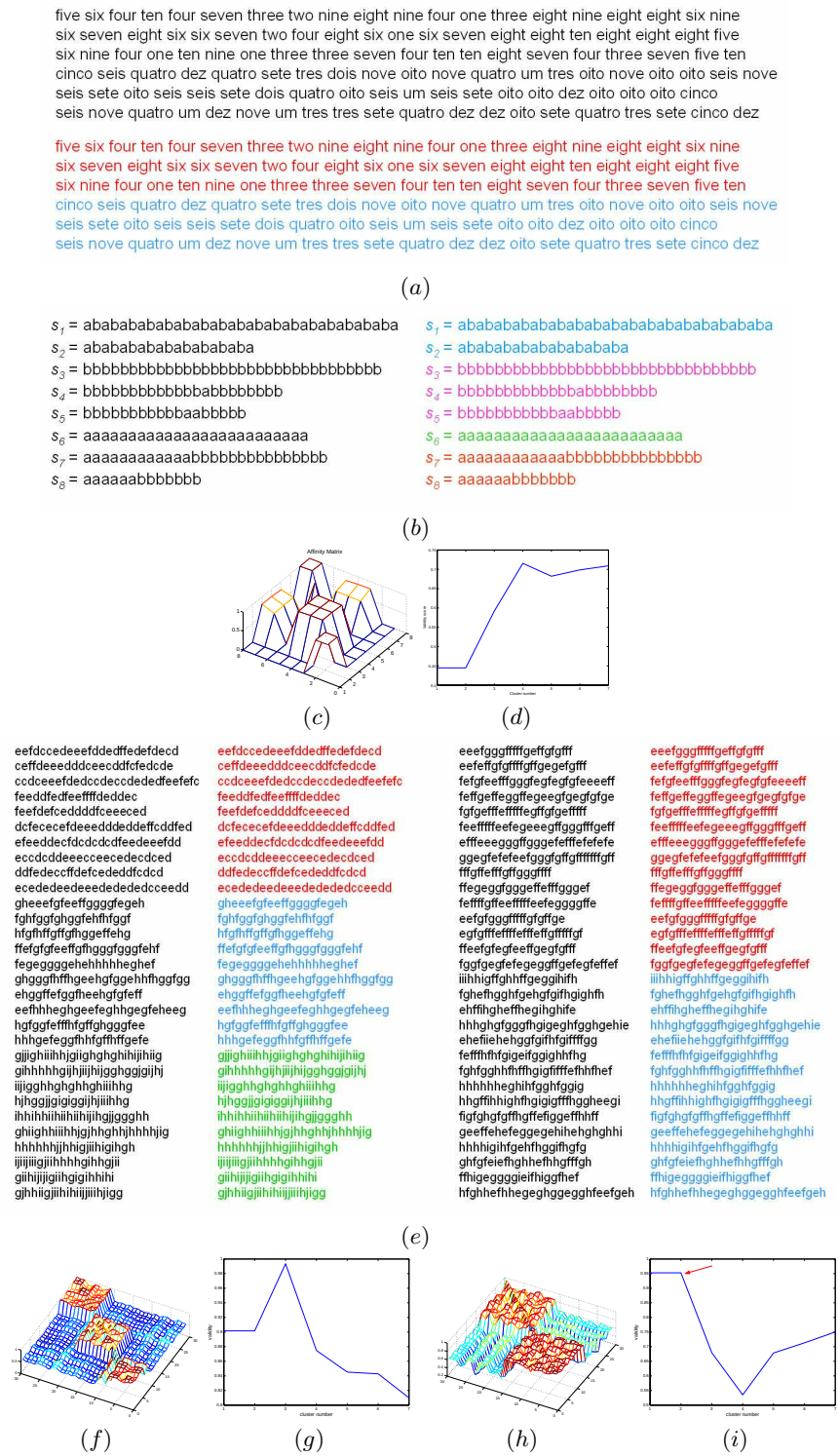

Fig. 2. (a) Language discrimination set (upper) and the clustering results (lower). (b) Pattern detection set (left) the computed clusters (right), and (c) affinity matrix, (d) validity scores for this set. (e) Two random length, random distribution scenarios, (f-g) affinity matrix and validity scores for the 1^{st} column of (e), and (h-i) affinity matrix and validity scores for the 3^{rd} column of (e).