Kernel Trick Embedded Gaussian Mixture Model

Dheeraj Kurukunda (22B0935) - Anumalasetty Varshith (22B0907)

CS 736 : Medical Image Computing - Spring 2023-24

Abstract

The primary focus of this project is the utilization of the Kernel Trick within Gaussian mixture models for data segmentation. The Kernel Trick enables the effective separation of non-linear data by transforming it into a higher-dimensional space where linear separation becomes feasible.

1 Introduction

Kernel trick is an efficient method for nonlinear data analysis early used by SVM's. It could be used in any algorithm that could be cast in the term of dot products. One such famous Algo is Kernel Principal Component Analysis (Kernel PCA).

This project is based on based on a paper[4] which proposed a method to embed kernel trick into Expectation-Maximization (EM) algorithm, and deduce a new parameter estimation algorithm for Gaussian Mixture Model (GMM) in the feature space. The entire model is called kernel Gaussian Mixture Model (kGMM).

1.1 Kernel Trick

We can implicitly map input data into a high dimension feature space via a nonlinear function:

$$\Phi : X \to H, \quad (x \mapsto \phi(x)) \tag{1}$$

And a similarity measure is defined from the dot product in space H as follows:

$$k(x_1, x_2) = \phi(x_1) \cdot \phi(x_2) \tag{2}$$

Even if we know the map Φ this avoids expensive computation cost in feature space by employing the kernel function k instead of directly computing dot product in H.

Some famous kernel functions are

- Radial basis function (RBF) kernel $k(x_1, x_2) = exp\left(-\gamma||x_1 - x_2||^2\right)$
- Polynomial kernel $k(x_1, x_2) = (1 + x_1 \cdot x_2)^d$

2 GMM Based On EM algorithm

It is a probabilistic model with with M classes has probability distribution

$$p(x|\Theta) = \sum_{i=1}^{M} \alpha_i G_i(x|\theta_i) \tag{3}$$

Where each α_i is weight for a partucular class with $\sum_{i=1}^{M} \alpha_i$ and each multi-variate Gaussian is distribution of particular class

$$G_l(x|\theta_l) = \frac{1}{(2\pi)^{d/2}|\Sigma_l|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_l)^T\Sigma_l^{-1}(x-\mu_l)\right) \tag{4}$$

where $\theta_l = (\mu_l, \Sigma_l)$.
The update steps for l^{th} class parameters in EM algorithm with N data points are **E step:**

$$\alpha_l^{(t)} = \frac{1}{N}\sum_{l=1}^{M} p(l|x_i, \Theta^{(t-1)}) \tag{5}$$

$$\mu_l^{(t)} = \frac{\sum_{i=1}^{N} x_i p(l|x_i, \Theta^{(t-1)})}{\sum_{i=1}^{N} p(l|x_i, \Theta^{(t-1)})} \tag{6}$$

$$\Sigma_l^{(t)} = \frac{\sum_{i=1}^{N} (x_i - \mu_l^{(t)})(x_i - \mu_l^{(t)})^T p(l|x_i, \Theta^{(t-1)})}{\sum_{i=1}^{N} p(l|x_i, \Theta^{(t-1)})} \tag{7}$$

M step:

$$p(l|x_i, \Theta^{(t-1)}) = \frac{\alpha_l^{(t-1)} G_l(x_i|\theta_l^{(t-1)})}{\sum_{l=1}^{M} \alpha_l^{(t-1)} G_l(x_i|\theta_l^{(t-1)})} \tag{8}$$

$\Theta^{(t-1)}$ and $\Theta^{(t)}$ represents the parameters at iteration $t-1$ and at iteration t respectively. Specifically, $\Theta^{(t)} = \{\alpha(t), \ldots, \alpha(t); \theta(t), \ldots, \theta(t)\}$ are the parameters of the tth iteration.

3 Kernel Principal Component Analysis

In Kerenl PCA, first we find Kernel Matrix K, whose ijth entry is given below

$$K_{ij} = \langle \phi(x_i), \phi(x_j) \rangle = k(x_i, x_j) \tag{9}$$

In the computation we need centered data in feature space

$$\widetilde{\phi}(x_i) = \phi(x_i) - \frac{1}{N}\sum_{j=1}^{N} \phi(x_j) \tag{10}$$

We actually will only need centered Kernel Matrix with ijth entry

$$K_{ij} = \langle \widetilde{\phi}(x_i), \widetilde{\phi}(x_j) \rangle = k(x_i, x_j) \tag{11}$$

We can compute centered Kernel matrix [2] from original Kernel matrix as

$$\widetilde{K} = K - E_N \cdot K - K \cdot E_N + E_N \cdot K \cdot E_N \tag{12}$$

Where E_N is NxN matrix with each entry 1/N

The reason for computing this centered Kernel matrix is that, the eigen values of this Matrix are same as eigen values of Covariance Matrix (Σ_f) [1]

$$\Sigma_f = \frac{1}{N}\sum_{i=1}^{N} \widetilde{\phi}(x_i) \cdot \widetilde{\phi}(x_i)^T \tag{13}$$

If β_k and V_k are eigenvectors of $\widetilde{K}$ and Σ_f respectively with eigenvalue λ_k

$$\Sigma_f V_k = \lambda_k V_k, \quad \widetilde{K}\beta_k = \lambda_k \beta_k \tag{14}$$

The most important, in feqature space the projection of centered ith data point ($\widetilde{\phi}(x_i)$) onto kth eigen vector V_k (y_{ik}) is equal to dot product of β_k and ith column of $\widetilde{K}$. (Provided, we have done some special normalization on β_k)

$$y_{ik} = \widetilde{\phi}(x_i) \cdot V_k = \beta_k \cdot \Gamma_i \tag{15}$$

where Γ_i is ith column of $\widetilde{K}$

4 Kernel Trick Embedded GMM

The notations that are in paper are as shown in below Table 1

Table 1. Notation List

$p_{li}^{(t)} = p(l\|\phi(x_i), \Theta^{(t)})$	Posterior that $\phi(x_i)$ belongs to the l^{th} component.
$w_{li}^{(t)} = \sqrt{(p_{li}^{(t)} / \sum_{j=1}^{M} p_{ji}^{(t)})}$	$(w_{li}^{(t)})^2$ represents ratio that $\phi(x_i)$ is occupied by the l^{th} Gaussian component.
$\mu_l^{(t)} = \sum_{i=1}^{N} \phi(x_i)(w_{li}^{(t)})^2$	Mean vector of the l^{th} Gaussian component.
$\tilde{\phi}_l^{(t)}(x_i) = \phi(x_i) - \mu_l^{(t)}$	Centered image of $\phi(x_i)$.
$\Sigma_l^{(t)} = \sum_{i=1}^{N} \tilde{\phi}(x_i)\tilde{\phi}(x_i)^T (w_{li}^{(t)})^2$	Covariance matrix of the l^{th} Gaussian component.
$(\mathbf{K}_l^{(t)})_{ij} = \big((w_{li}\phi(x_i) \cdot w_{lj}\phi(x_j)\big)$	Kernel matrix
$(\tilde{\mathbf{K}}_l^{(t)})_{ij} = \big((w_{li}\tilde{\phi}(x_i) \cdot w_{lj}\tilde{\phi}(x_j)\big)$	Centered kernel matrix.
$(\mathbf{K}_l^{(t)})'_{ij} = \big((\phi(x_i) \cdot w_{lj}\phi(x_j)\big)$	Projecting kernel matrix.
$(\tilde{\mathbf{K}}_l^{(t)})'_{ij} = \big(\tilde{\phi}(x_i) \cdot w_{lj}\tilde{\phi}(x_j)\big)$	Centered projecting kernel matrix.
$\lambda_{le}^{(t)},\ V_{le}^{(t)}$	Eigenvalue and eigenvector of $\Sigma_l^{(t)}$.
$\lambda_{le}^{(t)},\ \beta_{le}^{(t)}$	Eigenvalue and eigenvector of $\tilde{\mathbf{K}}_l^{(t)}$.

Here instead of just centered Kernel matrix, we have weighted centered Kernel Matrix ($\widetilde{K}_l$) and single weighted centered Kerenel matrix ($\widetilde{K}'_l$)for each class l. Similar to centering in kPCA, here we have centering equations as

$$\widetilde{K}_l^{(t)} = K_l^{(t)} - W_l^{(t)}K_l^{(t)} - K_l^{(t)}W_l^{(t)} + W_l^{(t)}K_l^{(t)}W_l^{(t)} \tag{16}$$

$$\widetilde{K}_l^{'(t)} = K_l^{'(t)} - W_l^{'(t)}K_l^{(t)} - K_l^{'(t)}W_l^{(t)} + W_l^{'(t)}K_l^{(t)}W_l^{(t)} \tag{17}$$

where $W_l^{(t)}$ and $W_l^{'(t)}$ are given by

$$W_l^{(t)} = w_l^{(t)}(w^{(t)})^T \tag{18}$$

$$W_l^{'(t)} = 1_N(w^{(t)})^T \tag{19}$$

where 1_N is N-dimensional column vector with all entries equal to 1

Now the main objective

$$G_l(\phi(x)|\theta_l) = \frac{1}{(2\pi)^{N/2}|\Sigma_l|^{1/2}} \exp\left(-\frac{1}{2}(\widetilde{\phi}(x_i))^T\Sigma_l^{-1}(\widetilde{\phi}(x_i))\right) \tag{20}$$

We know that inverse of a matrix has same eigenvectors as original matrix, with inverse eigenvalues. So if we do eigenvalue decomposition of Σ_l^{-1} we get

$$\Sigma_l^{-1} = V\sigma V^T \tag{21}$$

where σ is diagonal matrix with entries $1/\lambda_i$ and V is matrix of eigenvalues. Now the value inside exponent becomes [3]

$$(\widetilde{\phi}(x_i))^T\Sigma_l^{-1}(\widetilde{\phi}(x_i)) = (\widetilde{\phi}(x_i))^T V\sigma V^T(\widetilde{\phi}(x_i)) \tag{22}$$

$$= Y_i^T\sigma Y_i \tag{23}$$

$$= \sum_{k=1}^{N} \frac{(y_{ik})^2}{\lambda_k} \tag{24}$$

where Y_i is matrix of projection of ith data point onto eigenvectors of Σ_l^{-1}, which are eigenvectors of Σ_l.

We can compute value of y_{ik} using β_k and $\widetilde{K}_l^{'(t)}$ as

$$y_{ik} = \widetilde{\phi}(x_i) \cdot V_k = \beta_k \cdot \Gamma_i \tag{25}$$

where β_k is kth eigen vector of $\widetilde{K}_l^{(t)}$ and Γ_i is ith column of $\widetilde{K}_l^{'(t)}$ Also the determinant of

$$|\Sigma_l| = \prod_{k=1}^{N} \lambda_k \tag{26}$$

So the complete equation decomposes to

$$G_l(\phi(x)|\theta_l) = \frac{1}{(2\pi)^{N/2}(\prod_{k=1}^{N}\lambda_k)^{N/2}} exp\left(-\frac{1}{2}\sum_{k=1}^{N}\frac{(y_{ik})^2}{\lambda_k}\right) \tag{27}$$

4.1 Probability distribution approximation

Most of the times some eigenvalues would be very very small or even zero. So we make an approximation by considering only largest d_ϕ (Whose value is data dependent) eigenvalue corresponding eigenvectors and consider only projections onto them.

The approximated probability distribution is given by

$$\begin{aligned} G_l(\phi(x)|\theta_l) = &\left[\frac{1}{(2\pi)^{d_\phi/2}(\prod_{k=1}^{N}\lambda_k)^{d_\phi/2}} exp\left(-\frac{1}{2}\sum_{k=1}^{d_\phi}\frac{(y_{ik})^2}{\lambda_k}\right)\right] \\ &\times \left[\frac{1}{(2\pi)^{(N-d_\phi)/2}(\prod_{k=1}^{N}\lambda_k)^{(N-d_\phi)/2}} exp\left(-\frac{\epsilon^2}{2\rho}\right)\right] \end{aligned} \tag{28}$$

Where ϵ^2 is sum of squares of all other (N - d_ϕ) projections. Since we know that sum of all N squares of projections is

$$\sum_{k=1}^{N}(y_{ik})^2 = \langle\widetilde{\phi}(x_i), \widetilde{\phi}(x_i)\rangle = (\widetilde{K}_l^{(t)})_{ii} \tag{29}$$

$$\epsilon^2 = \sum_{k=d_\phi+1}^{N}(y_{ik})^2 = (\widetilde{K}_l^{(t)})_{ii} - \sum_{k=1}^{N}(y_{ik})^2 \tag{30}$$

And ρ is average of all other (N - d_ϕ) eigenvalues. We know that sum of all eigenvalues is trace of matrix, so

$$\rho = \frac{1}{(N-d_\phi)}\sum_{k=d_\phi+1}^{N}\lambda_k = trace(\widetilde{K}_l^{(t)}) - \sum_{k=1}^{d_\phi}\lambda_k \tag{31}$$

The approximation is very similar to

$$\sum_{k=d_\phi+1}^{N}\frac{(y_{ik})^2}{\lambda_k} = \frac{\sum_{k=d_\phi+1}^{N}(y_{ik})^2}{\sum_{k=d_\phi+1}^{N}\lambda_k} \tag{32}$$

only that we multiply by an extra (N - d_ϕ)

4.2 Closed form Update steps of EM algorithm

E step:

$$\alpha_l^{(t)} = \frac{1}{N}\sum_{l=1}^{M}p(l|x_i, \Theta^{(t-1)}) \tag{33}$$

$$w_{il}^{(t)} = \sqrt{\frac{p(l|x_i, \Theta^{(t-1)})}{\sum_{l=1}^{M}p(l|x_i, \Theta^{(t-1)})}} \tag{34}$$

$$\text{Update all matrices } W_l^{(t)}, W_l^{'(t)}, K_l^{(t)}, K_l^{'(t)}, \widetilde{K}_l^{(t)}, \widetilde{K}_l^{'(t)} \tag{35}$$

$$\text{Compute eigenvalues, eigenvectors, projections } \lambda_k, \beta_k, \epsilon^2, \rho, y_{ik} \tag{36}$$

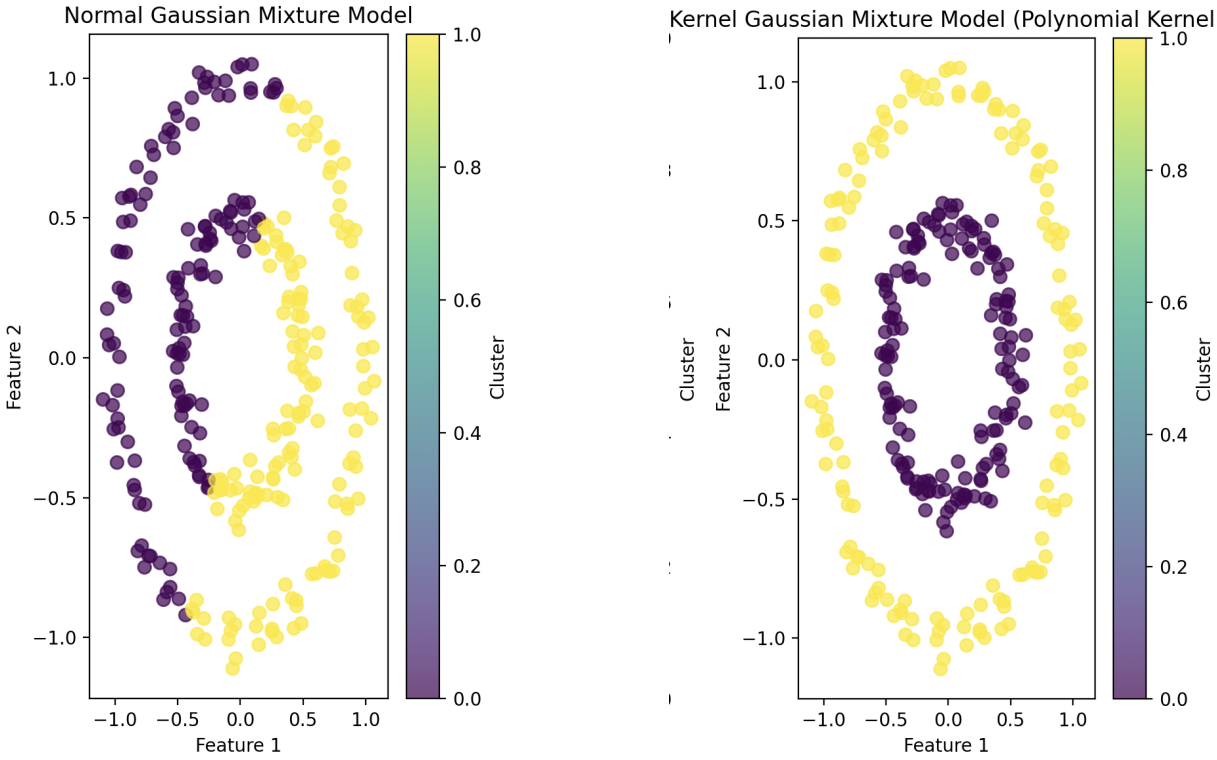
$$\text{Compute } G_l(\phi(x)|\theta_l) \tag{37}$$

M step:

$$p(l|x_i, \Theta^{(t-1)}) = \frac{\alpha_l^{(t-1)}G_l(x_i|\theta_l^{(t-1)})}{\sum_{l=1}^{M}\alpha_l^{(t-1)}G_l(x_i|\theta_l^{(t-1)})} \tag{38}$$

5 Results

5.1 Two circles with same center, different radii



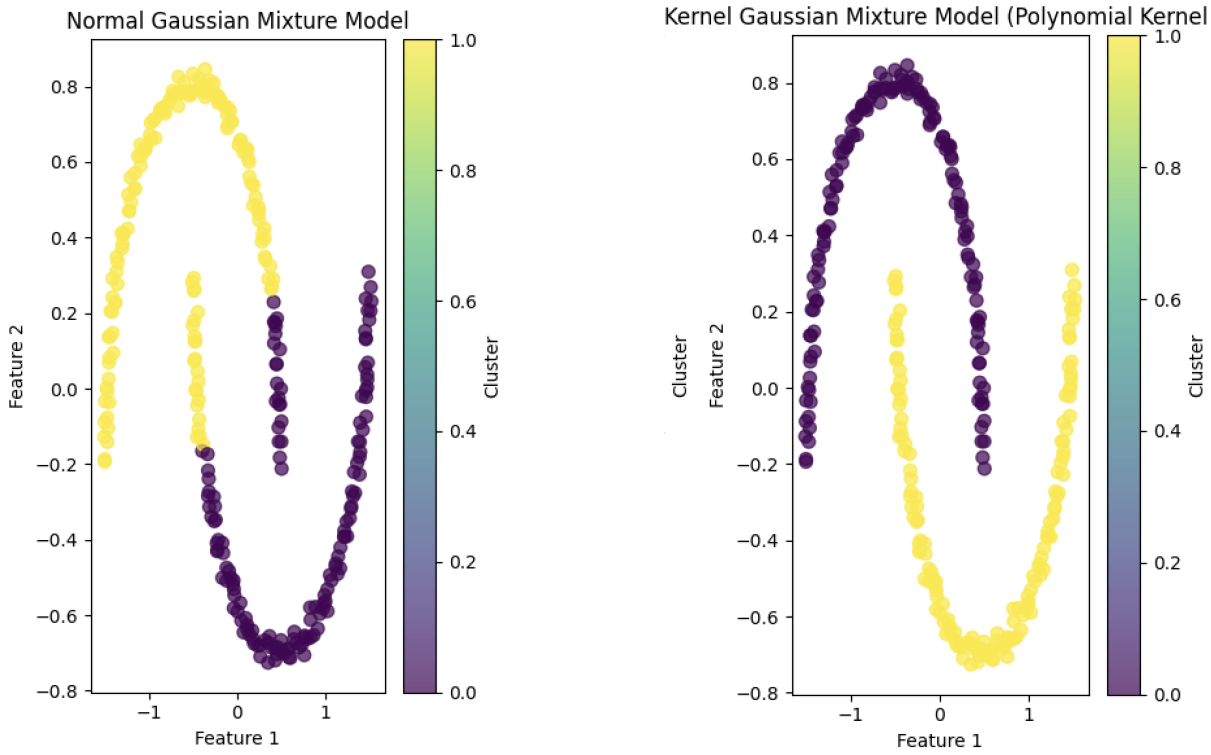
(a) Segmentation by GMM

(b) Segmentation by KGMM

Figure 1: Variation between GMM and KGMM

- We have used polynomial kernel of degree two.
- Normal GMM could only split into eliptical classes, which is not the case with kernel GMM.

5.2 Two moons of intersecting mouths



(a) Segmentation by GMM

(b) Segmentation by KGMM

Figure 2: Variation between GMM and KGMM

- We have used Radial basis function (RBF) kernel with gamma 7.
- Normal GMM could only split into eliptical classes, which is not the case with kernel GMM.

5.3 Silhouette analysis in feature space

Silhouette analysis is a method used to evaluate the goodness of clustering results. It provides a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

The silhouette value for each data point i is calculated using the following formula:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

where:

1. $s(i)$ is the silhouette value for data point i,
2. $a(i)$ is the average distance from i to other points in the same cluster,
3. $b(i)$ is the minimum average distance from i to points in a different cluster (i.e., the nearest neighboring cluster).

Kernel trick for distance in feature space

- The distance in feature space is given by

$$||\phi(x) - \phi(y)||^2 = ||\phi(x)||^2 + ||\phi(y)||^2 - 2\langle\phi(x), \phi(y)\rangle \tag{39}$$

$$= k(x, x) + k(y, y) - 2 * k(x, y) \tag{40}$$

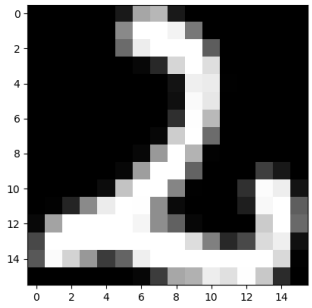
- For above moon dataset by doing Silhouette analysis, the results are as shown in below table

Method	Minimum value	Maximum value	Average	Standard deviation
Kernel GMM	0.045	0.154	0.138	0.019
GMM	-0.254	0.655	0.470	0.203

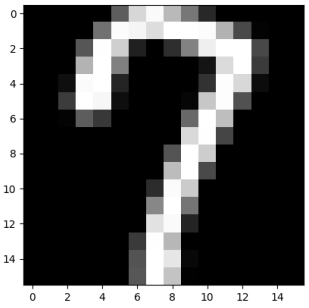
Table 1: Comparison results on USPS data set

- We found it intersting that Kernel GMM performs worse in this metric.
- Because in feature space even though the moon Clusters are linearly seperable, clusters might be too close.
- Where as in the input space though GMM doesn't seperate them the way we want(i.e into 2 moons), the 2 ellipses it drew in input space are infact more compact compared to that of Kernel GMM in feature space.
- So Silhouette analysis is a good metric for comparing 2 clusterings only if they work in the same dimensional space.

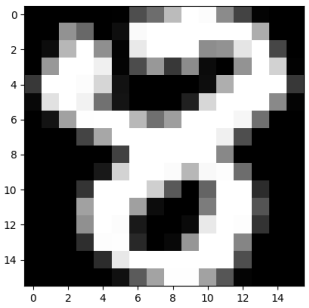
5.4 USPS digit clustering



(a) Handwritten image of 2



(b) Handwritten image of 7



(c) Handwritten image of 8

Figure 3: Three images side by side

- We took 100 points for each of the numbers 2, 7, 8
- Used Gaussian kernel wth gamma 0.0005 for clustering
- The best error rates are as follows

Method	error in class 2	error in class 7	error in class 8	Overall Best Error rate
GMM	3.70 %	3.38 %	3.96 %	8.67 %
Kernel GMM	16.53 %	35.55 %	16.86 %	22.33 %

Table 2: Comparison results on USPS data set

- The paper described that for classes greater than two, kGMM described needs more modification for better work

References

[1] Alberto García-González, Antonio Huerta, Sergio Zlotnik, Pedro Díez. *A kernel Principal Component Analysis (kPCA) digest with a new backward mapping (pre-image reconstruction) strategy*. 2021.

[2] Jie Yu. *A nonlinear kernel Gaussian mixture model based inferential monitoring approach for fault detection and diagnosis of chemical processes*. 2012.

[3] Moghaddam, B. and Pentland, A. *Probabilistic visual learning for object detection*. 1995.

[4] Wang, Jingdong and Lee, Jianguo and Zhang, Changshui. *Kernel Trick Embedded Gaussian Mixture Model*. Springer Berlin Heidelberg, 2003.