GMM on the Motion Data of a Cat Pounce

1. Introduction

This dataset was acquired by taking a 1920x1080 pixel 30 FPS video of two cats: Piccolo and Gohan. The video shows Piccolo standing still and Gohan running and pouncing at a target out of frame. A Gaussian Mixture Model (GMM) will be created from the data. Singular Value Decomposition (SVD) was previously performed on the dataset to understand the variance created by the principal components. The mixture model will be used to identify clusters of data based on component sizes and will generate new sample images using those components.

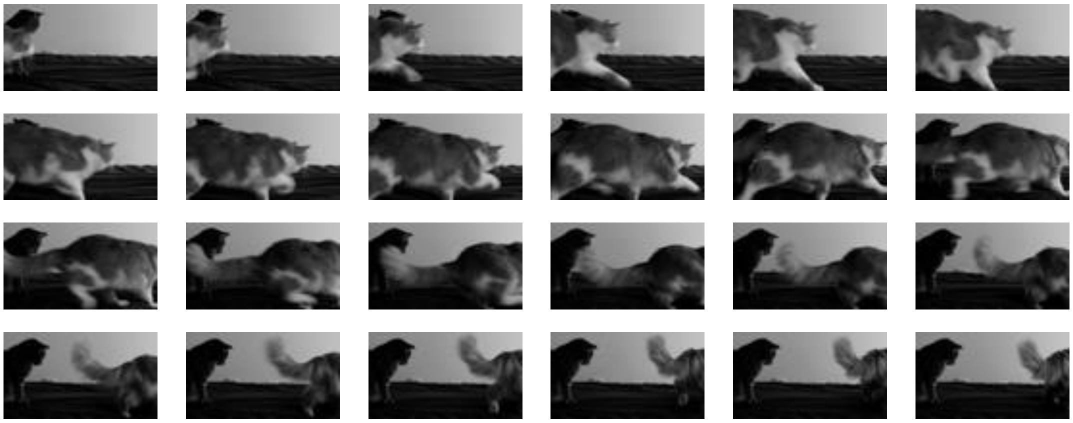

Figure 1. Frames 1 through 24 of the sequence resized to 80x45 pixels (ordered left to right, top to bottom).

2. Method

A GMM is a Probability Distribution Function (PDF) consisting of k components. The GMM uses the k components to create a single PDF which is a combination of the PDFs associated with each component, weighted by the mixture weights, π_i, which determine the proportion each component will contribute to the mixture model. An Expectation Maximization (EM) algorithm will be used to create the GMM from the dataset. The Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) will then be used to choose the best number of components to fit the model with.

I. Expectation Maximization (EM)

Once the mean (μ_k), covariance (Σ_k), and mixture weight (π_k) matrices are initialized from the data and the number of k components is chosen, EM is done in a loop consisting of two steps: the E-step and the M-step.

a. E-Step

Equation 1. Responsibilities function.

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_j^K \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)}$$

The E-step calculates the responsibilities using the updated μ_k, Σ_k, and π_k matrices. Before the E-step is run for the first time, the initial responsibility is calculated from the initialized μ_k, Σ_k, and π_k matrices. This initial responsibility is then passed in to process the M-step. The output from the M-step generates the input to create the new responsibility in the first run of the E-step. This new responsibility will be compared to the previously iterated (or initial) responsibility in order to determine if the algorithm has converged. The responsibilities are used to create a probability that a certain point, x_n, comes from component k.

b. M-Step

Equation 2. Update functions for the mean, covariance, and mixture weights.

$$\mu_k = \frac{1}{N_k}\sum_{n=1}^{N} r_{nk}x_n\,, \Sigma_k = r_{nk}(x_n - \mu_k)(x_n - \mu_k)^{\top}, \pi_k = \frac{N_k}{N}, \text{where } N_k = \sum_{n=1}^{N} r_{nk}$$

The M-step calculates the updated values of μ_k, Σ_k, and π_k from the responsibility previously calculated in the E-step. When the EM algorithm runs the first step, the responsibility which was calculated from the initialized μ_k, Σ_k, and π_k matrices is passed into the M-step update functions to generate the updated μ_k, Σ_k, and π_k matrices. These updated matrices are used to calculate the new responsibility.

c. Iterations

The algorithm is run either until the number of predetermined steps has been reached or if the absolute difference between the previous iteration's responsibilities and updated responsibilities is smaller than a predetermined value, which causes the algorithm to converge and end. If the function converges, this theoretically means the data points have been assigned to the components of the data they are most likely to have come from.

II. Log Likelihood

Equation 3. Log-Likelihood function for $n = 24$ observations, i components

$$\log(\mathcal{L}) = \sum_{n=24}^{N} \log(\mathrm{p}(x_n|\theta) = \sum_{n=24}^{N} \log\left(\sum_{k=i}^{k}(\pi_k \mathcal{N}(\pi_k|\mu_k, \Sigma_k)\right), \text{where } \theta = \{\pi_k, \mu_k, \Sigma_k : k = 1, \dots, K\}$$

The log-likelihood function is used to compute the maximum likelihood estimate (MLE), which finds the optimal value of θ to maximize the likelihood $\mathcal{L}$. The updated μ_k, Σ_k, and π_k values produced by the EM algorithm are passed to this function in order to maximize $\mathcal{L}$. The MLE values produced by this function are used to compute the discrete values that make up the PDF of the GMM, which can be visualized by the contour plots of different components in Figure 5.

3. Data

The dataset contains 24 observations as columns and 3600 features as rows. Labels were generated sequentially for each image. The observations of the data are the 24 columns, which represent each image frame of the video. The features of the data are the 3600 rows, each of which is a 1-dimensional representation of the 80x45 pixel frames of the video. In addition to the preprocessing of reducing the frame sizes down to grayscale 80x45 pixels, the data was mean-centered and its dimensions were reduced with Principal Component Analysis (PCA).

Preprocessing

I. Grayscale, Resizing, and Flattening

The video was converted to grayscale to remove the RGB channels and allow for easier creation of an image matrix containing only the black levels. To decrease the computational time of the code, the original video was trimmed down from 70 to 24 frames (0.8 seconds) of the original video and resized by a factor of 24 (coincidentally) to 80x45 pixels. The 24 images were then flattened by concatenating the rows of pixel data for each image to a single vector of length 3600. The flattened images were then added sequentially to an array of 3600 columns and 24 rows.

II. Mean-Centering

Mean-centering was done along the rows of the data by taking the mean of each row (image) and subtracting it from each value in the row. Mean-centering the data creates results which are easier to interpret by centering the data about the origin and helps simplify the process of reducing the size of the dataset with PCA.

III. PCA Reduction

PCA was used to reduce the data down to two dimensions. This was done to prevent overfitting of the data with only 24 images and a dimension of 3600. Python code was used from the Scikit-learn library as opposed to extracting the principal components through SVD and matrix multiplication. This difference in methods caused the transformed data that will be fit to the GMM to be vertically flipped, as shown in Figure 2. Therefore, any information obtained relating to the motion of the cat Gohan will need to be interpreted in a counterclockwise direction in regard to the labels, as opposed to clockwise with the previous PCA method.

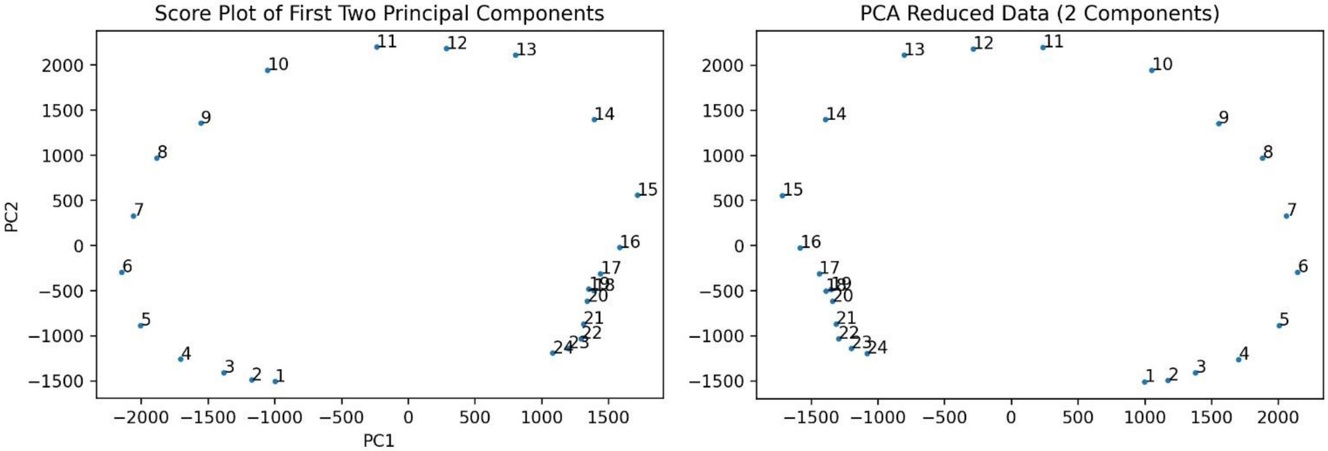

Figure 2. Comparison of transformed data between two different PCA methods.

4. Results

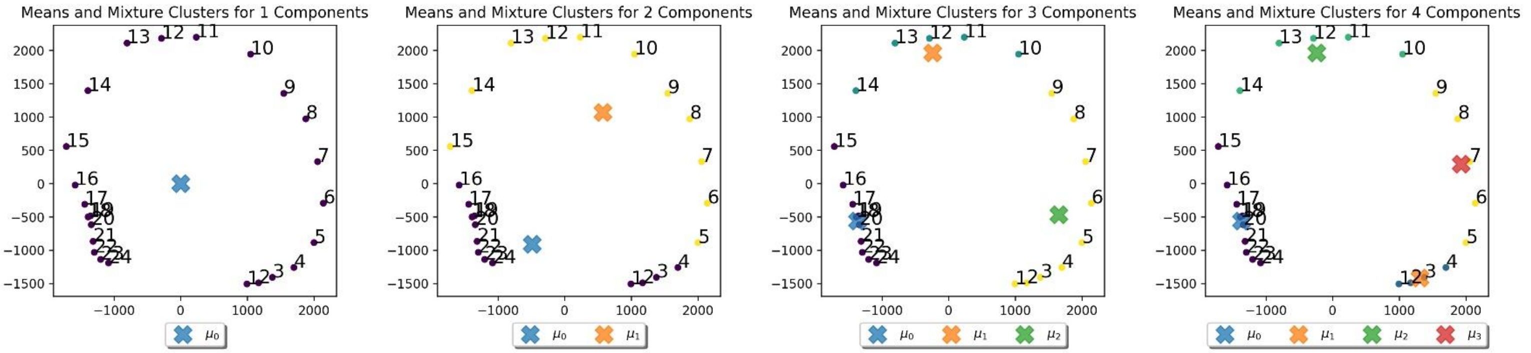

Figure 3. Data points colored by mixture components overlayed with component means for 1-4 components.

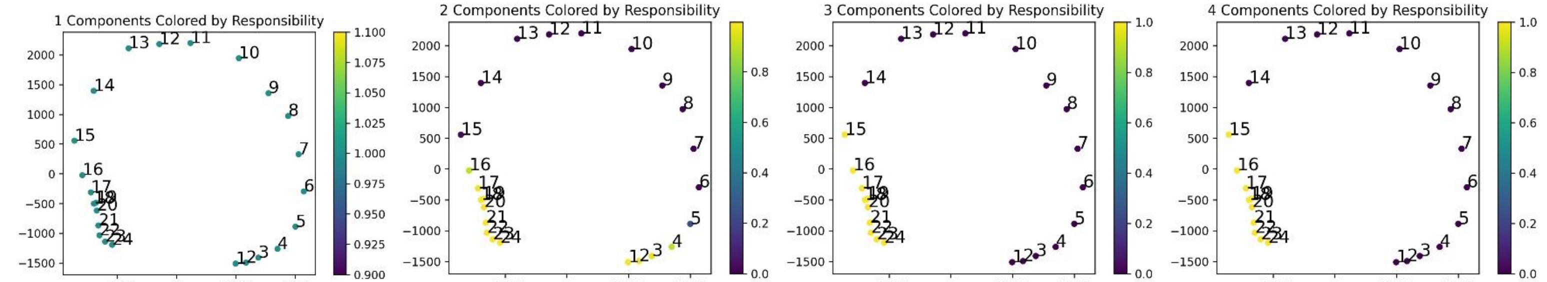

Figure 4. Data points colored by responsibility for 1-4 components.

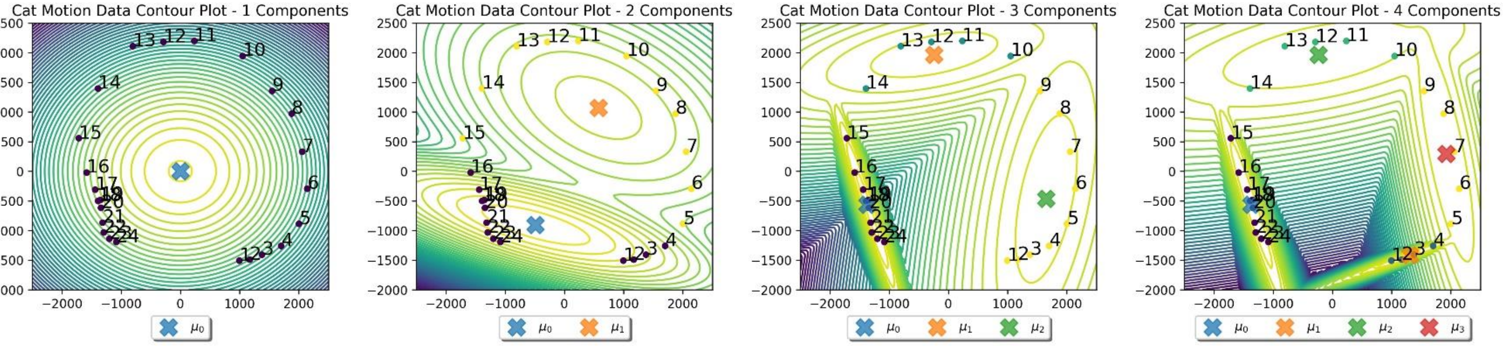

Figure 5. Contour plots overlayed with mixture components and means for 1-4 components.

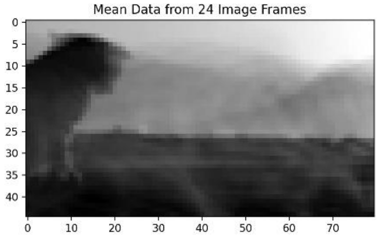

Figure 6. Mean of the dataset.

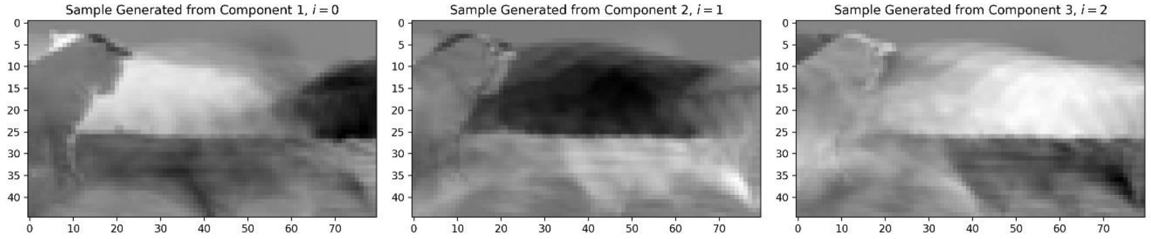

Figure 7. Sample mean-centered data generated from 1-3 components.

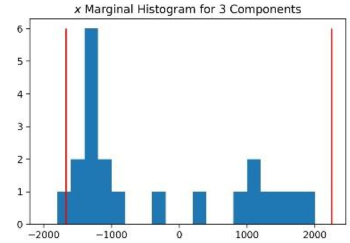

Figure 8. Histogram of the X marginal, 3 components.

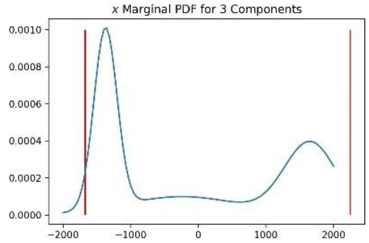

Figure 9. PDF of the X marginal, 3 components.

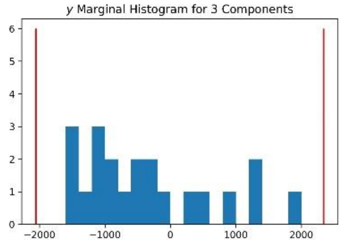

Figure 10. Histogram of the Y marginal, 3 components.

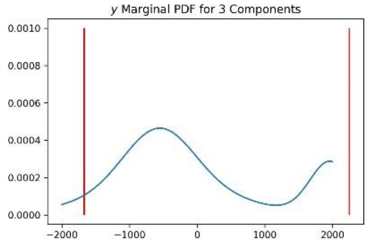

Figure 11. PDF of the Y marginal, 3 components.

4. Results (Continued)

Equation 4. 95% confidence interval for the X marginal.

$$\pi\phi(u|\mu_{x1},\sigma_{x1})+\pi\phi(u|\mu_{x2},\sigma_{x2})=0.025,\pi\phi(u|\mu_{x1},\sigma_{x1})+\pi\phi(u|\mu_{x2},\sigma_{x2})=0.975$$

$$0.025=0.417*0.5\left(1+erf\left(\frac{u-(-1370.384)}{\sqrt{2}*171.031}\right)\right)+0.208*0.5\left(1+\mathrm{erf}\left(\frac{u-(-238.484)}{\sqrt{2}*841.709}\right)\right)+0.375*0.5\left(1+\mathrm{erf}\left(\frac{u-(1654.679)}{\sqrt{2}*384.579}\right)\right),u=-1673.79$$

$$0.975=0.417*0.5\left(1+erf\left(\frac{v-(-1370.384)}{\sqrt{2}*171.031}\right)\right)+0.208*0.5\left(1+\mathrm{erf}\left(\frac{v-(-238.484)}{\sqrt{2}*841.709}\right)\right)+0.375*0.5\left(1+\mathrm{erf}\left(\frac{v-(1654.679)}{\sqrt{2}*384.579}\right)\right),v=2248.55$$

Equation 5. 95% confidence interval for the Y marginal.

$$\pi\phi(u|\mu_{y1},\sigma_{y1})+\pi\phi(u|\mu_{y2},\sigma_{y2})=0.025,\pi\phi(u|\mu_{y1},\sigma_{y1})+\pi\phi(u|\mu_{y2},\sigma_{y2})=0.975$$

$$0.025=0.417*0.5\left(1+erf\left(\frac{u-(-561.807)}{\sqrt{2}*515.929}\right)\right)+0.208*0.5\left(1+\mathrm{erf}\left(\frac{u-(1963.449)}{\sqrt{2}*298.893}\right)\right)+0.375*0.5\left(1+\mathrm{erf}\left(\frac{u-(-465.986)}{\sqrt{2}*1047.613}\right)\right),u=-2054.47$$

$$0.975=0.417*0.5\left(1+erf\left(\frac{u-(-561.807)}{\sqrt{2}*515.929}\right)\right)+0.208*0.5\left(1+\mathrm{erf}\left(\frac{u-(1963.449)}{\sqrt{2}*298.893}\right)\right)+0.375*0.5\left(1+\mathrm{erf}\left(\frac{u-(-465.986)}{\sqrt{2}*1047.613}\right)\right),v=2338.25$$

5. Discussion

Three components seemed to visually fit this dataset better than one, two, or four components. In Figure 5, for three components, the contour plot shows three clusters detected by the GMM which appear to reflect three separate moments in the motion data shown in Figure 1. Starting from the right of the contour plot going counterclockwise, the cluster of points representing frames 1 through 9 are the beginning of the cat Gohan's acceleration. The points representing frames 10 through 14 are Gohan slowing down, and the points representing frames 15 through 24 are Gohan coming to a full stop as he reaches the target. The respective updated mixture weights for these components are $\pi_2 = 0.37509, \pi_1 = 0.20824$, and $\pi_0 = 0.41667$. These values tell us that the third component, Gohan's acceleration, is responsible for approximately 37.509% of the data, the second component, or the magnitude of Gohan's acceleration decreasing, is responsible for approximately 20.824% of the data, and the first component, Gohan decelerating and stopping, is responsible for approximately 41.667% of the data.

6. Evaluation

The plot of AIC and BIC values in Figure 12 appear to show that three components create the best fit for this dataset. It is worth noting that both three and four components produce a lower AIC and BIC value which indicates a possible good fit. However, we can use the contour plots in Figure 5 for four components to see that three components are favorable due to the fact that at four components, the GMM begins to create clusters of points that may be too small to gain meaningful insight from the data. The frames represented by the four points centered around μ_1 do not show any significant change in Gohan's behavior compared to the five points centered around μ_3, as both clusters show Gohan running faster.

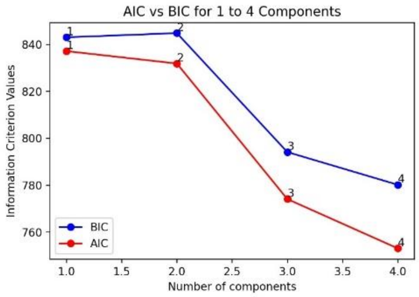

Figure 12. Plot of AIC and BIC values for 1-4 components.

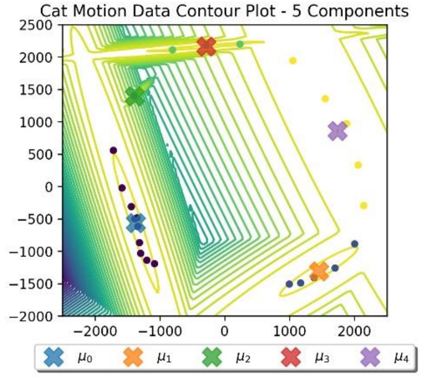

Figure 13. Contour plots overlayed with mixture components and means for 1-4 components.

Creating a GMM for the data produced noteworthy results, but due to the small size of observations in the data, the EM algorithm used to create the model was not as effective as it could have been with larger data. This is due to computational numerical issues with calculating the responsibilities of data points, as the EM algorithm can sometimes choose a small number of points or a single point from the data to be a mixture component after the algorithm is run through for a certain higher number of components. In this case, as shown in Figure 13, running the EM algorithm for 5 components first caused this phenomenon to occur by assigning μ_3 to three points and μ_2 to a single point. The error then arose through trying to calculate the covariance of these small groups of points, which is either relatively small to the point where it is below computational zero (1×10^{-16}) or is equal to zero from the group containing a single point. The algorithm then tries to invert this singular matrix, producing the error. This error may be able to be avoided with a much larger dataset of images either taken at a higher framerate or over a longer period of time.

The three-component GMM is able to generate the three samples of the mean data for each component as shown in Figure 7. These samples are variations of the mean of the data and do not necessarily provide any useful information to extrapolate from, such as creating frames between frames. However, a different data set with many more observations could benefit from a GMM. For example, a GMM could be created from dozens or hundreds of similar images of a single object. The samples generated from this large dataset could be a rudimentary attempt at creating AI images such as the images that can be produced by DALLE-Mini and similar projects.