Cats vs. Dogs

Raymond Ahn
Daradipo Bou
Augusto Partida
Justin Sunu

CSU, Long Beach

May 10, 2012

Outline

1 Introduction

2 Methods
- Wavelet Analysis
- Principal Angles
- Kohonen's Novelty Filter
- Kernel Linear Discriminant Analysis

3 Results

4 Conclusion

Introduction

- The goal of this project is to test various algorithms in order to correctly identify images of cats and dogs.
- We ran 4 methods through cross-validation in order to maximize the percentage of correctly identified images.
- And finally, run the final set of test images with each algorithm.

Introduction

- The goal of this project is to test various algorithms in order to correctly identify images of cats and dogs.
- We ran 4 methods through cross-validation in order to maximize the percentage of correctly identified images.
- And finally, run the final set of test images with each algorithm.

Introduction

- The goal of this project is to test various algorithms in order to correctly identify images of cats and dogs.
- We ran 4 methods through cross-validation in order to maximize the percentage of correctly identified images.
- And finally, run the final set of test images with each algorithm.

What is the process of image recognition?

- Find a way to reduce the size of the image in order to maximize efficiency.
- Reconstruct an image using optimal bases so that the most amount of information is captured.
- Extract various information about the image.
- Use various methods to compare images.

What is the process of image recognition?

- Find a way to reduce the size of the image in order to maximize efficiency.
- Reconstruct an image using optimal bases so that the most amount of information is captured.
- Extract various information about the image.
- Use various methods to compare images.

What is the process of image recognition?

- Find a way to reduce the size of the image in order to maximize efficiency.
- Reconstruct an image using optimal bases so that the most amount of information is captured.
- Extract various information about the image.
- Use various methods to compare images.

What is the process of image recognition?

- Find a way to reduce the size of the image in order to maximize efficiency.
- Reconstruct an image using optimal bases so that the most amount of information is captured.
- Extract various information about the image.
- Use various methods to compare images.

What is the process of image recognition?

- Find a way to reduce the size of the image in order to maximize efficiency.
- Reconstruct an image using optimal bases so that the most amount of information is captured.
- Extract various information about the image.
- Use various methods to compare images.

The data

Figure : Images of cats and dogs

Eigencat and Eigendog

Since eigenvectors are an important aspect of all our methods, we feel it is important to explore the eigencat and eigendog.

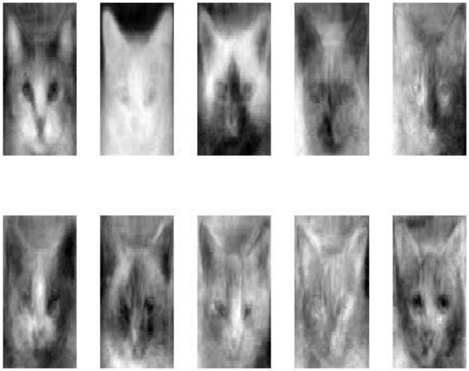

Figure : The first 10 eigencats

Eigencat and Eigendog

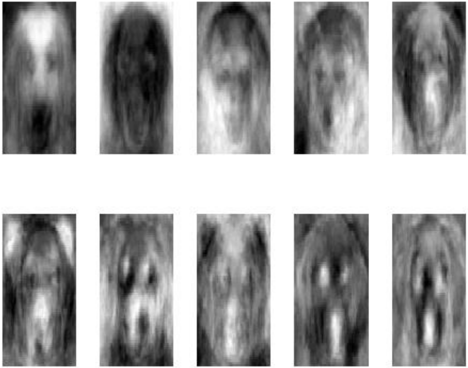

Figure : The first 10 eigendogs

Wavelet Analysis

- Wavelet analysis allows us to extract features from an image in the form of “wavelets".
- For images, we require a 2-D scaling function $\phi(x, y)$, and three 2-D wavelets, $\psi^H(x, y)$, $\psi^V(x, y)$, and $\psi^D(x, y)$.
- These wavelets measure functional variations (intensity variations for images) along different dimensions: ψ^H measures along the columns, ψ^V measures along the rows, and ψ^D measures along the diagonals [2].

Wavelet Analysis

- Wavelet analysis allows us to extract features from an image in the form of “wavelets".
- For images, we require a 2-D scaling function $\phi(x, y)$, and three 2-D wavelets, $\psi^H(x, y)$, $\psi^V(x, y)$, and $\psi^D(x, y)$.
- These wavelets measure functional variations (intensity variations for images) along different dimensions: ψ^H measures along the columns, ψ^V measures along the rows, and ψ^D measures along the diagonals [2].

Wavelet Analysis

- Wavelet analysis allows us to extract features from an image in the form of “wavelets".
- For images, we require a 2-D scaling function $\phi(x, y)$, and three 2-D wavelets, $\psi^H(x, y)$, $\psi^V(x, y)$, and $\psi^D(x, y)$.
- These wavelets measure functional variations (intensity variations for images) along different dimensions: ψ^H measures along the columns, ψ^V measures along the rows, and ψ^D measures along the diagonals [2].

The equations

Define the scaled and translated basis functions as:

$$\begin{aligned} \phi^j_{m,n}(x,y) &= 2^{-j/2}\phi(2^{-j}x-m, 2^{-j}y-n) \\ (\psi^j_{m,n}(x,y))^i &= 2^{-j/2}\psi(2^{-j}x-m, 2^{-j}y-n), \end{aligned}$$

The discrete wavelet transform of an image $f(x,y)$ of size $M \times N$ is then:

$$\begin{aligned} c^j_{m,n} &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\phi^j_{m,n}(x,y) \\ (d^j_{m,n})^i &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)(\psi^j_{m,n}(x,y))^i, \end{aligned}$$

where $i = \{H, V, D\}$.

The equations

Define the scaled and translated basis functions as:

$$\begin{aligned}\phi^j_{m,n}(x,y) &= 2^{-j/2}\phi(2^{-j}x-m, 2^{-j}y-n)\\ (\psi^j_{m,n}(x,y))^i &= 2^{-j/2}\psi(2^{-j}x-m, 2^{-j}y-n),\end{aligned}$$

The discrete wavelet transform of an image $f(x,y)$ of size $M \times N$ is then:

$$\begin{aligned}c^j_{m,n} &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\phi^j_{m,n}(x,y)\\ (d^j_{m,n})^i &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)(\psi^j_{m,n}(x,y))^i,\end{aligned}$$

where $i = \{H, V, D\}$.

Implementation

- In order to implement wavelet analysis, we used MATLAB's "dwt2" function along with the 'Haar' wavelets.
- Since each iteration of DWT creates four images, each of size $M/2 \times N/2$, we only need to do 2 or iterations.
- At the 2$^{\text{nd}}$ iteration, we already have a 16×16 image, and any further decomposition of it yields very pixelated images with very little information.
- Since wavelet analysis is only a means to generate wavelets (images in our case), analysis of the wavelets are done with the following methods.

Implementation

- In order to implement wavelet analysis, we used MATLAB's "dwt2" function along with the 'Haar' wavelets.
- Since each iteration of DWT creates four images, each of size $M/2 \times N/2$, we only need to do 2 or iterations.
- At the 2nd iteration, we already have a 16×16 image, and any further decomposition of it yields very pixelated images with very little information.
- Since wavelet analysis is only a means to generate wavelets (images in our case), analysis of the wavelets are done with the following methods.

Implementation

- In order to implement wavelet analysis, we used MATLAB's “dwt2" function along with the ’Haar’ wavelets.
- Since each iteration of DWT creates four images, each of size $M/2 \times N/2$, we only need to do 2 or iterations.
- At the 2$^{\text{nd}}$ iteration, we already have a 16×16 image, and any further decomposition of it yields very pixelated images with very little information.
- Since wavelet analysis is only a means to generate wavelets (images in our case), analysis of the wavelets are done with the following methods.

Implementation

- In order to implement wavelet analysis, we used MATLAB's "dwt2" function along with the 'Haar' wavelets.
- Since each iteration of DWT creates four images, each of size $M/2 \times N/2$, we only need to do 2 or iterations.
- At the 2$^{\text{nd}}$ iteration, we already have a 16×16 image, and any further decomposition of it yields very pixelated images with very little information.
- Since wavelet analysis is only a means to generate wavelets (images in our case), analysis of the wavelets are done with the following methods.

Decomposition of a cat and dog

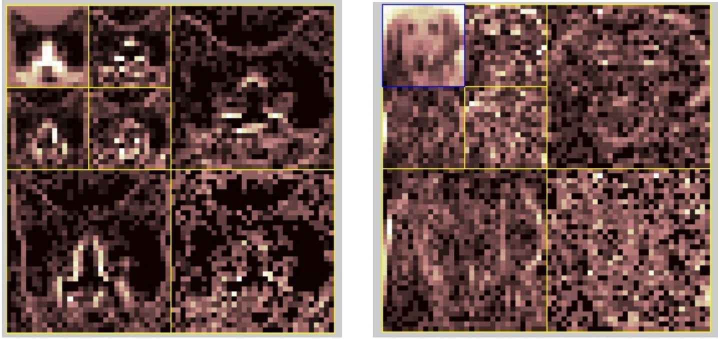

Figure : 2 level decomposition of a cat (left) and dog (right).

Principal Angles - Introduction

- A projection, in $\mathbb{R}^2$, is a transformation of a vector into a different vectors direction.
- Conceptually, in higher dimensions, this is the equivalent of transforming a vector space into another vector space's direction.
- The advantage of performing a projection is that it allows for easier comparisons.

Principal Angles - Introduction

- A projection, in $\mathbb{R}^2$, is a transformation of a vector into a different vectors direction.
- Conceptually, in higher dimensions, this is the equivalent of transforming a vector space into another vector space's direction.
- The advantage of performing a projection is that it allows for easier comparisons.

Principal Angles - Introduction

- A projection, in $\mathbb{R}^2$, is a transformation of a vector into a different vectors direction.
- Conceptually, in higher dimensions, this is the equivalent of transforming a vector space into another vector space's direction.
- The advantage of performing a projection is that it allows for easier comparisons.

Singular Value Decomposition

- Singular Value Decomposition, or SVD, is decomposing a matrix into 3 matrices, $M = U\Sigma V^T$.
- U is a unitary matrix, $U^{-1} = U^T$, of M's eigenvectors.
- U is an orthonormal basis, which means it can be used as a projection matrix.
- Σ is a diagonal matrix, such that the values of the diagonal are the eigenvalues of M.
- V^T is a unitary matrix of M's eigenvectors.

Singular Value Decomposition

- Singular Value Decomposition, or SVD, is decomposing a matrix into 3 matrices, $M = U\Sigma V^T$.
- U is a unitary matrix, $U^{-1} = U^T$, of M's eigenvectors.
- U is an orthonormal basis, which means it can be used as a projection matrix.
- Σ is a diagonal matrix, such that the values of the diagonal are the eigenvalues of M.
- V^T is a unitary matrix of M's eigenvectors.

Singular Value Decomposition

- Singular Value Decomposition, or SVD, is decomposing a matrix into 3 matrices, $M = U\Sigma V^T$.
- U is a unitary matrix, $U^{-1} = U^T$, of M's eigenvectors.
- U is an orthonormal basis, which means it can be used as a projection matrix.
- Σ is a diagonal matrix, such that the values of the diagonal are the eigenvalues of M.
- V^T is a unitary matrix of M's eigenvectors.

Singular Value Decomposition

- Singular Value Decomposition, or SVD, is decomposing a matrix into 3 matrices, $M = U\Sigma V^T$.
- U is a unitary matrix, $U^{-1} = U^T$, of M's eigenvectors.
- U is an orthonormal basis, which means it can be used as a projection matrix.
- Σ is a diagonal matrix, such that the values of the diagonal are the eigenvalues of M.
- V^T is a unitary matrix of M's eigenvectors.

Singular Value Decomposition

- Singular Value Decomposition, or SVD, is decomposing a matrix into 3 matrices, $M = U\Sigma V^T$.
- U is a unitary matrix, $U^{-1} = U^T$, of M's eigenvectors.
- U is an orthonormal basis, which means it can be used as a projection matrix.
- Σ is a diagonal matrix, such that the values of the diagonal are the eigenvalues of M.
- V^T is a unitary matrix of M's eigenvectors.

Principal Angles

- In $\mathbb{R}^2$, you can use the angle between two vectors to determine how similar they are.
- $\cos(\theta) = \frac{u \cdot v}{\|u\| \|v\|}$
- As you go up in dimensions, you start to work with vector spaces, which consists of a multiple vectors. To handle this, you use principal Angles.
- Principal angles is applying the $\cos(\theta)$ formula for all combinations of vectors among the two vector spaces.
- The smaller the θ the more "similar".

Principal Angles

- In $\mathbb{R}^2$, you can use the angle between two vectors to determine how similar they are.
- $\cos(\theta) = \frac{u \cdot v}{\|u\|\|v\|}$
- As you go up in dimensions, you start to work with vector spaces, which consists of a multiple vectors. To handle this, you use principal Angles.
- Principal angles is applying the $\cos(\theta)$ formula for all combinations of vectors among the two vector spaces.
- The smaller the θ the more "similar".

Principal Angles

- In $\mathbb{R}^2$, you can use the angle between two vectors to determine how similar they are.
- $\cos(\theta) = \frac{u \cdot v}{\|u\| \|v\|}$
- As you go up in dimensions, you start to work with vector spaces, which consists of a multiple vectors. To handle this, you use principal Angles.
- Principal angles is applying the $\cos(\theta)$ formula for all combinations of vectors among the two vector spaces.
- The smaller the θ the more "similar".

Principal Angles

- In $\mathbb{R}^2$, you can use the angle between two vectors to determine how similar they are.
- $\cos(\theta) = \frac{u \cdot v}{\|u\|\|v\|}$
- As you go up in dimensions, you start to work with vector spaces, which consists of a multiple vectors. To handle this, you use principal Angles.
- Principal angles is applying the $\cos(\theta)$ formula for all combinations of vectors among the two vector spaces.
- The smaller the θ the more “similar".

Principal Angles

- In $\mathbb{R}^2$, you can use the angle between two vectors to determine how similar they are.
- $\cos(\theta) = \frac{u \cdot v}{\|u\|\|v\|}$
- As you go up in dimensions, you start to work with vector spaces, which consists of a multiple vectors. To handle this, you use principal Angles.
- Principal angles is applying the $\cos(\theta)$ formula for all combinations of vectors among the two vector spaces.
- The smaller the θ the more “similar".

Putting it all together

M is the training sets, and N is the test set.

- Perform *SVD* on the training data, $M = U_M \Sigma_M V_M^T$.
- *Project* the test set onto the training set, $U_M N$.
- Compute *principal angles* between the training sets and projected test set.
- The training set is $\Sigma_M V_M^T$.
- The projected test set is $U_M N$.

Putting it all together

M is the training sets, and N is the test set.

- Perform *SVD* on the training data, $M = U_M \Sigma_M V_M^T$.
- *Project* the test set onto the training set, $U_M N$.
- Compute *principal angles* between the training sets and projected test set.
- The training set is $\Sigma_M V_M^T$.
- The projected test set is $U_M N$.

Putting it all together

M is the training sets, and N is the test set.

- Perform *SVD* on the training data, $M = U_M \Sigma_M V_M^T$.
- *Project* the test set onto the training set, $U_M N$.
- Compute *principal angles* between the training sets and projected test set.
- The training set is $\Sigma_M V_M^T$.
- The projected test set is $U_M N$.

Putting it all together

M is the training sets, and N is the test set.

- Perform *SVD* on the training data, $M = U_M \Sigma_M V_M^T$.
- *Project* the test set onto the training set, $U_M N$.
- Compute *principal angles* between the training sets and projected test set.
- The training set is $\Sigma_M V_M^T$.
- The projected test set is $U_M N$.

Putting it all together

M is the training sets, and N is the test set.

- Perform *SVD* on the training data, $M = U_M \Sigma_M V_M^T$.
- *Project* the test set onto the training set, $U_M N$.
- Compute *principal angles* between the training sets and projected test set.
- The training set is $\Sigma_M V_M^T$.
- The projected test set is $U_M N$.

Kohonen's Novelty Filter

- Kohonen's way to compare the two images and see the difference.
- The process is to pick the best characteristic from each of the test sample by using Singular Value Decomposition.
- The image that we want to test will be projected to the Singular Value Decomposition and subtract from the original, whichever give the smallest two norms (implies smaller difference from the Singular Value Decomposition) will be categorize to be the same class with the image.

Kohonen's Novelty Filter

- Kohonen's way to compare the two images and see the difference.
- The process is to pick the best characteristic from each of the test sample by using Singular Value Decomposition.
- The image that we want to test will be projected to the Singular Value Decomposition and subtract from the original, whichever give the smallest two norms (implies smaller difference from the Singular Value Decomposition) will be categorize to be the same class with the image.

Kohonen's Novelty Filter

- Kohonen's way to compare the two images and see the difference.
- The process is to pick the best characteristic from each of the test sample by using Singular Value Decomposition.
- The image that we want to test will be projected to the Singular Value Decomposition and subtract from the original, whichever give the smallest two norms (implies smaller difference from the Singular Value Decomposition) will be categorize to be the same class with the image.

The equations

$$
\begin{aligned}
X = &\quad SC \times SC' \times TT(:,i) - TT(:,i) \\
Y = &\quad SD \times SD' \times TT(:,i) - TT(:,i) \qquad i = 1, 2, ...K,
\end{aligned}
$$

If $\|X\|_2 \leq \|Y\|_2$ the test consider it to be a cat and it is a dog otherwise.

The reason why the filter pick SC or SD is because $SC \times SC' = I$, where I is the identity matrix. As a result after we apply the filter, we should get the same (or close to the original vector) if they are from the same type.

The equations

$$
\begin{aligned}
X = & \quad SC \times SC' \times TT(:, i) - TT(:, i) \\
Y = & \quad SD \times SD' \times TT(:, i) - TT(:, i) \qquad i = 1, 2, ...K,
\end{aligned}
$$

If $\|X\|_2 \leq \|Y\|_2$ the test consider it to be a cat and it is a dog otherwise.
The reason why the filter pick SC or SD is because $SC \times SC' = I$, where I is the identity matrix. As a result after we apply the filter, we should get the same (or close to the original vector) if they are from the same type.

KLDA - Introduction

- KLDA generalizes LDA since in the transformed space, the principal components are nonlinearly related to the input variables.
- Kernel Linear Discriminant Analysis (KLDA) maps the input space into a high dimensional, nonlinear feature space. This transformation is carried out by a kernel function $\phi : X \rightarrow F$.
- Common kernel function is the RBF (Gaussian): $\phi(\mathbf{x}, \mathbf{y}) = exp\left(-\frac{||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}\right)$ where the assumption is that the classes have a multivariate Gaussian distribution.

KLDA - Introduction

- KLDA generalizes LDA since in the transformed space, the principal components are nonlinearly related to the input variables.
- Kernel Linear Discriminant Analysis (KLDA) maps the input space into a high dimensional, nonlinear feature space. This transformation is carried out by a kernel function $\phi : X \rightarrow F$.
- Common kernel function is the RBF (Gaussian): $\phi(\mathbf{x}, \mathbf{y}) = exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}\right)$ where the assumption is that the classes have a multivariate Gaussian distribution.

KLDA - Introduction

- KLDA generalizes LDA since in the transformed space, the principal components are nonlinearly related to the input variables.
- Kernel Linear Discriminant Analysis (KLDA) maps the input space into a high dimensional, nonlinear feature space. This transformation is carried out by a kernel function $\phi : X \to F$.
- Common kernel function is the RBF (Gaussian): $\phi(\mathbf{x}, \mathbf{y}) = exp\left(-\frac{||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}\right)$ where the assumption is that the classes have a multivariate Gaussian distribution.

KLDA - Intro. cont

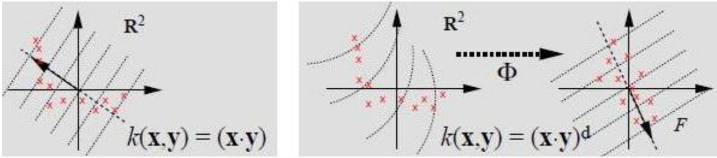

Figure : Feature space transformation.

Implementation

- After mapping data to feature space, then same procedure as LDA to find optimal direction that separates classes:
  - Rayleigh quotient: $J(\mathbf{w}) = \frac{\mathbf{w}^T S_B^{\phi} \mathbf{w}}{\mathbf{w}^T S_W^{\phi} \mathbf{w}}$.
  - Solve generalized eigenvalue problem: $S_B^{\phi}\mathbf{w} = J(\mathbf{w}) S_W^{\phi}\mathbf{w}$.
- Performed leave-one-out cross-validation (LOOCV) on training set to find best parameter σ for kernel function and energy for dimensionality reduction using principal components analysis (PCA) ([1],[3],[4]).

Implementation

- After mapping data to feature space, then same procedure as LDA to find optimal direction that separates classes:
  - Rayleigh quotient: $J(\mathbf{w}) = \frac{\mathbf{w}^T S_B^\phi \mathbf{w}}{\mathbf{w}^T S_W^\phi \mathbf{w}}$.
  - Solve generalized eigenvalue problem: $S_B^\phi \mathbf{w} = J(\mathbf{w}) S_W^\phi \mathbf{w}$.
- Performed leave-one-out cross-validation (LOOCV) on training set to find best parameter σ for kernel function and energy for dimensionality reduction using principal components analysis (PCA) ([1],[3],[4]).

Method of testing

- For each method we run it through cross-validation.
- Furthermore, we used different sets of different numbers of training images in order to find the “optimal" number of training images in order to produce the best results.
- Logically, it would seem that using all images as a training set would be the best, but if we could produce the same results with half as many, then the time it takes will be reduced.
- After all the data is collected, we average the results.

Method of testing

- For each method we run it through cross-validation.
- Furthermore, we used different sets of different numbers of training images in order to find the “optimal" number of training images in order to produce the best results.
- Logically, it would seem that using all images as a training set would be the best, but if we could produce the same results with half as many, then the time it takes will be reduced.
- After all the data is collected, we average the results.

Method of testing

- For each method we run it through cross-validation.
- Furthermore, we used different sets of different numbers of training images in order to find the “optimal" number of training images in order to produce the best results.
- Logically, it would seem that using all images as a training set would be the best, but if we could produce the same results with half as many, then the time it takes will be reduced.
- After all the data is collected, we average the results.

Method of testing

- For each method we run it through cross-validation.
- Furthermore, we used different sets of different numbers of training images in order to find the “optimal" number of training images in order to produce the best results.
- Logically, it would seem that using all images as a training set would be the best, but if we could produce the same results with half as many, then the time it takes will be reduced.
- After all the data is collected, we average the results.

Final results

Our testing data consists of 38 images, 19 cats and 19 dogs. As stated before, we ran our algorithms with a varying number (40,50,60,70) of training images.

For each case, we ran 11 iterations cycling through all the images to make sure all images were included at least once in our training set.

For the methods that require cumulative energy, we used 99%.

Final Results

Method	40	50	60	70
Principal Angles	**.8684**	.8421	**.8684**	**.8684**
Novelty Filter	**.9211**	**.9211**	**.9211**	.8947
LDA	.8158	**.8421**	**.8421**	**.8421**

Table : Best results from using original data.

Final Results

Method	40	50	60	70
Principal Angles	.8421	**.8684**	**.8684**	.8421
Novelty Filter	.8947	**.9211**	.8947	.8947
LDA	.8421	**.8684**	.8421	.8421

Table : Best results from using first wavelet approximation.

Final Results

Method	40	50	60	70
Principal Angles	..8947	.8947	.8947	**.9211**
Novelty Filter	**.9211**	**.9211**	**.9211**	**.9211**
LDA	.7632	**.8158**	.7632	.7632

Table : Best results from using first wavelet horizontal detail.

Final Results

Method	40	50	60	70
Principal Angles	**.9211**	**.9211**	.8947	.8947
Novelty Filter	**.9474**	.9211	.9211	.8947
LDA	.7105	.6579	.6579	**.7368**

Table : Best results from using first wavelet vertical detail.

Final Results

Method	40	50	60	70
Principal Angles	.8421	**.8684**	**.8684**	**.8684**
Novelty Filter	**.9211**	**.9211**	**.9211**	.8947
LDA	**.7105**	.6842	.6579	.6053

Table : Best results from using second wavelet approximation.

Final Results

Method	40	50	60	70
Principal Angles	**.8947**	.8684	.8684	.8684
Novelty Filter	**.8947**	**.8947**	**.8947**	**.8947**
LDA	.7895	**.8684**	.8158	.7895

Table : Best results from using second wavelet horizontal detail.

Final Results

Method	40	50	60	70
Principal Angles	.8421	.8158	.8421	**.8684**
Novelty Filter	**.8421**	.8158	**.8421**	.8158
LDA	.5789	.6579	.6579	**.7368**

Table : Best results from using second wavelet vertical detail.

Final Results

Data Set	σ	e	Accuracy
Training	6.05	0.75	0.89375 (143/160)
Validation	5.45	0.75	0.92105 (35/38)

Table : KLDA classification performance.

Final Results

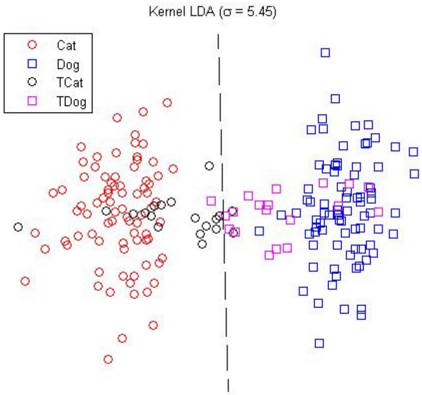

Figure : Separation using KLDA

Analysis of results

- Kohonen's Novelty Filter yielded the best overall performance with a peak of .9474 using 40 training images with the level 1 vertical detail wavelet.
- The number of images in the training set does affect the accuracy, but more importantly, which images are captured in the training set are more important.
- KLDA demonstrated best separation of classes over PCA and KPCA, but did not project well.
- MidRange (nonparametric) threshold classification boundary performed better than (parametric) Mahalanobis distance. This is an indicator that the data may not be Normal.

Analysis of results

- Kohonen's Novelty Filter yielded the best overall performance with a peak of .9474 using 40 training images with the level 1 vertical detail wavelet.
- The number of images in the training set does affect the accuracy, but more importantly, which images are captured in the training set are more important.
- KLDA demonstrated best separation of classes over PCA and KPCA, but did not project well.
- MidRange (nonparametric) threshold classification boundary performed better than (parametric) Mahalanobis distance. This is an indicator that the data may not be Normal.

Analysis of results

- Kohonen's Novelty Filter yielded the best overall performance with a peak of .9474 using 40 training images with the level 1 vertical detail wavelet.
- The number of images in the training set does affect the accuracy, but more importantly, which images are captured in the training set are more important.
- KLDA demonstrated best separation of classes over PCA and KPCA, but did not project well.
- MidRange (nonparametric) threshold classification boundary performed better than (parametric) Mahalanobis distance. This is an indicator that the data may not be Normal.

Analysis of results

- Kohonen's Novelty Filter yielded the best overall performance with a peak of .9474 using 40 training images with the level 1 vertical detail wavelet.
- The number of images in the training set does affect the accuracy, but more importantly, which images are captured in the training set are more important.
- KLDA demonstrated best separation of classes over PCA and KPCA, but did not project well.
- MidRange (nonparametric) threshold classification boundary performed better than (parametric) Mahalanobis distance. This is an indicator that the data may not be Normal.

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Future work

Although we only used 4 methods, there are a number of other methods that could be used for this problem. Some of the other methods we could try are:

- Radial Basis Functions
- Labeled Voronoi cell classification
- Set-to-set comparison with principal angles and Grassmannian distances
- Fourier Analysis
- Edge-based analysis
- Comparing different kernel function(s).

Thank you

References

G. Baudat and F. Anouar.
Generalized discriminant analysis using a kernel approach.
Neural Comput., October 2000.

Jen-Mei Chang.
Matrix methods for geometric data analysis and pattern recognition, 2011.

S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. Muller.
Fisher discriminant analysis with kernels, 1999.

B. Scholkopf, A. Smola, and K. Muller.
Kernel principal component analysis, 1999.

Any questions?