Machine Learning

Assignment -1

Identifying Animals in Videos on the basis of Image Classification

Group 29:

- Aradhya Khandelwal – 2015A7PS0036P
- S. Pranav Siddharth – 2015A7PS0047P
- Vaibhav Varma – 2015A7PS0091P

▪ Problem Statement:

In the Pokemon universe, a Pokedex is a device which can be used to identify any Pokemon just by pointing the Pokedex at it, and then also get useful information about the same. Drawing inspiration from it, our idea was to build something to replicate the same functionality for animals, by creating a program to identify animals in a given video.

The way used to achieve this was to create an image classifier for a set of chosen animals, using an ensemble of different classifiers. By running this classifier on frames of a given video, and aggregating the scores, the animals in a video can be identified.

A major inspiration for the implementation of Support Vector Machines (SVM) classification used was a paper published by L. Sirovich and M. Kirby in 1987, "Low-dimensional procedure for the characterization of human faces". This paper introduced a computationally inexpensive method of facial recognition, using the Principal Component Analysis (PCA) method of dimensionality reduction. In it, the dimensionality of the training data is first reduced drastically using PCA, and then the testing data is plotted along the dimensions gathered from PCA. This allows only the relevant information to actually influence the classification. Each dimension was called an 'eigenface', and each 'eigenface' can be thought of as representing a particular feature. The value for a particular 'eigenface' indicates the importance of each feature.

Data Description:

The dataset that was originally intended for use was "Animals on the Web", a dataset created by Tamara Berg of UNC Chapel Hill, but when the classifiers were actually run on the dataset, the precision and recall values were too low to be of use. Upon further inspection, the dataset was not deemed to not have the required quality, along with the presence of a significant number of false positives. Hence, a decision was made to construct a database from scratch, with particular emphasis on the quality of images gathered. The animals chosen were bears, canaries, dolphins, tigers and frogs. 24 high quality images for each animal were gathered from Google Image Search results for the animals.

For preprocessing, all images in the dataset were first reduced to 300x300 dimensions. This was done because each pixel of an image was going to be used as a feature, and all images must have the same number of pixels to allow for classification. The central 300x300 area of the image was taken, since the animal will mostly reside in the centre of the image, and thus allows for least loss of information.

After this, a 120x300x300x3 matrix was created, containing RGB values of every pixel of every image. Then PCA was applied to this matrix, to determine axes of highest variance. Then the number of dimensions that capture at least 4% of the maximum variance were calculated, which came out to be 10. Then the original data was fitted to the PCA, to create our training data set.

- ## Modelling of the problem:

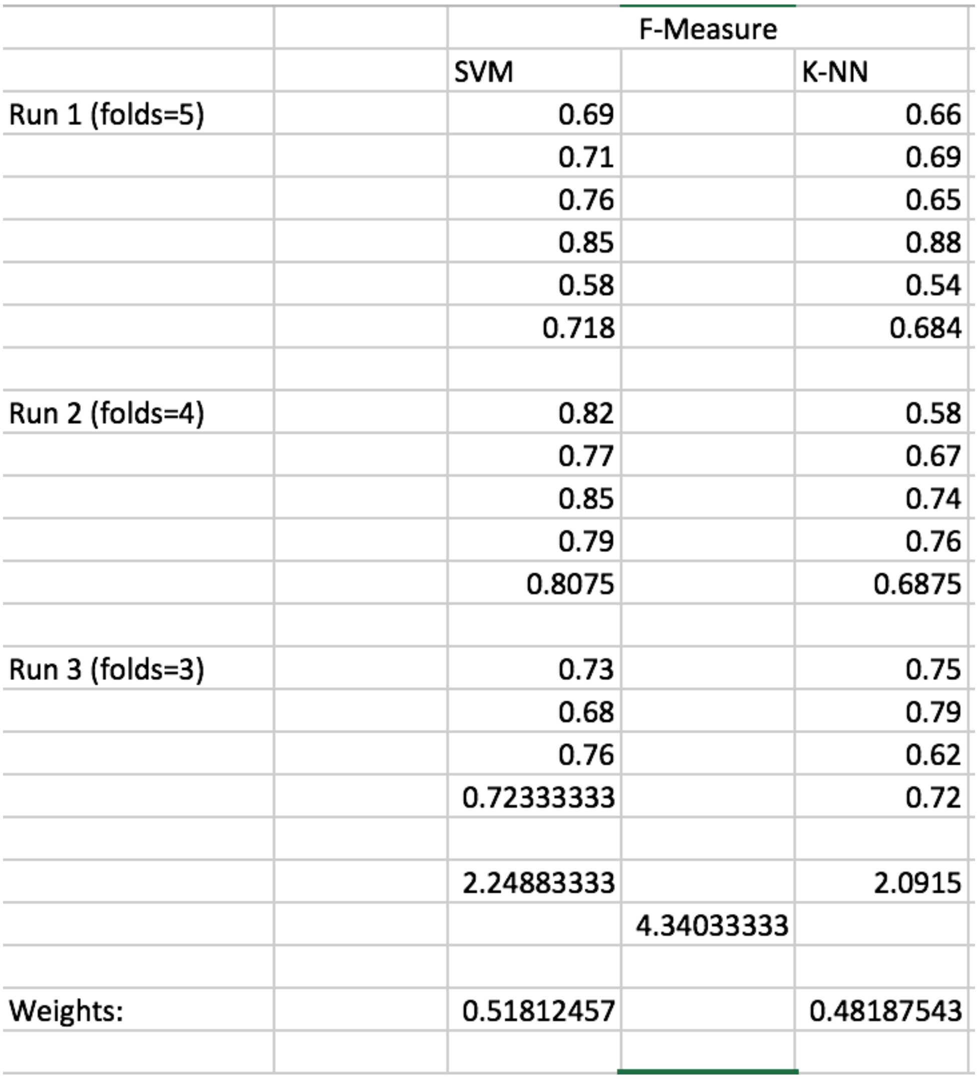
The classification model we decided to use was an ensemble of Support Vector Machines (SVM) and K-Nearest Neighbours (K-NN), because of their success in image recognition applications. The ensemble was chosen to be a weighted average of the scores obtained from SVM and K-NN classification.

To optimize the models, we ran Grid-Search on the SVM and K-NN models. The Grid-Search for SVM model adjusts parameters within specified values for kernels ('rbf' and 'linear'), C (1000, 5000, 10000, 50000, 100000) and gamma (0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1). The Grid-Search for K-NN model adjusts the number of neighbours from 1 to 15 trying to find the best fit for the model.

To determine the weights of the two models, we utilized K-Folds Cross Validation. K-Folds divides the original training data set into 'k' partitions, and does 'k' iterations over the dataset, taking a different partition as testing data every time, while the rest of the data acts as the training data. K-Folds helps determine the utility of a classification model, while also helping determine the weights to be given to each.

We ran K-Folds for values of 'k' ranging from 3 to 5, and then recorded Average F1-Score values (geometric mean of Average Precision and Average Recall) for the SVM and K-NN models, and then weighted it according to those findings. The recorded values were as given in Figure 1.

Post optimization, to achieve model persistence, we trained the model using the preprocessed data and stored the model as a '.pkl' file which could later be used for prediction.



			F-Measure	
		SVM		K-NN
Run 1 (folds=5)		0.69		0.66
		0.71		0.69
		0.76		0.65
		0.85		0.88
		0.58		0.54
		0.718		0.684
Run 2 (folds=4)		0.82		0.58
		0.77		0.67
		0.85		0.74
		0.79		0.76
		0.8075		0.6875
Run 3 (folds=3)		0.73		0.75
		0.68		0.79
		0.76		0.62
		0.72333333		0.72
		2.24883333		2.0915
			4.34033333	
Weights:		0.51812457		0.48187543

Figure 1

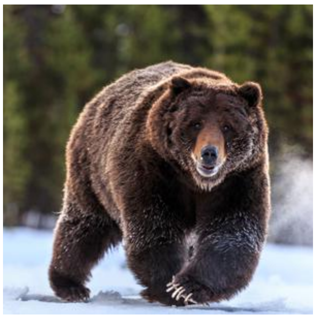
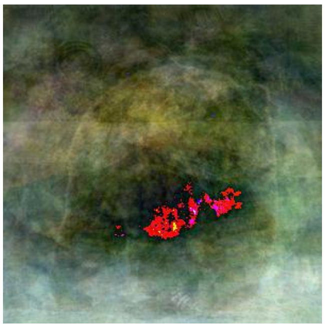
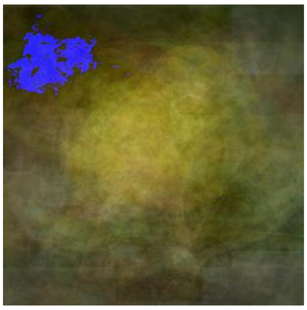
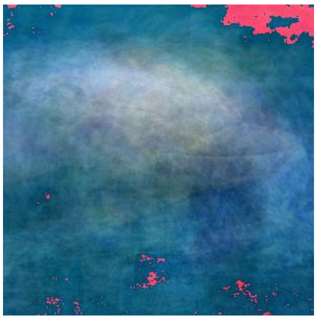
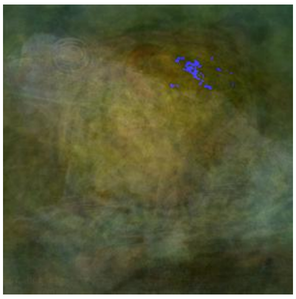
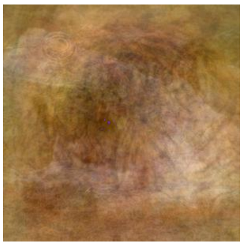
- # Methodology:

This section will describe how our program achieves animal identification from a given input video.

1) **Video Processing Module:** The objective of the video processing module is to extract frames from the given video. On average, videos contain 25 frames per second. We decided to extract every 10th frame for reducing computation cost, and also because there is no significant informational difference between adjacent frames. Another reason for doing so is that the number of testing data points ideally shouldn't be too close to the number of training data points.
2) **Preprocessing extracted frames:** The preprocessing module applied to the training data is called for every frame recorded from the video. This module first converts the images into 300x300 dimensions to achieve similarity to the training dataset. These processed images are then converted to an array of 300x300x3 dimensions, containing RGB values.
3) **Dimensionality Reduction:** The array obtained after above preprocessing step, is plotted along the principal components obtained from applying PCA on the training dataset. This transformed array will act as the input to our classification module.
4) **Classification:** Initially, we load our SVM and K-NN models from the '.pkl' files generated beforehand. A weighted prediction probability measure is obtained by weighing the prediction probability measures obtained by the SVM and K-NN models using the calculated weights. If the predicted class has a confidence measure of less than 0.75, that sample is discarded because of the possibility of noise, otherwise its identified as the animal for a particular frame. This classification process is repeated for all extracted frames, and the predicted classes are accumulated. The class with the maximum number of samples, as well as all classes with number of samples within 0.7 times the maximum are returned as the predicted animals in the video. These animals are then printed as output to the console.

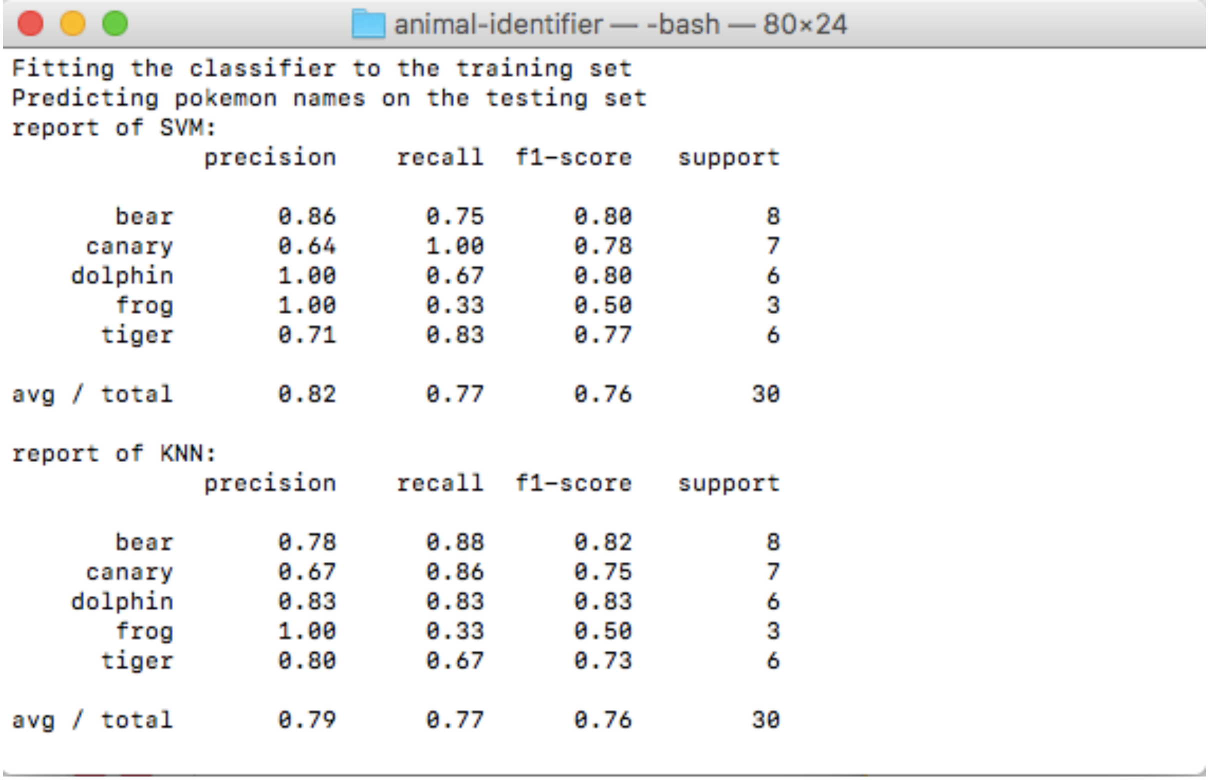
- ## Results and Discussion:

1) Reconstruction of images along principal components

Bear		
Canary		
Dolphin		
Frog		
Tiger		

The table given on the previous page gives the reconstruction of a sample image of each animal class along the principal components determined by PCA.

2) K-Folds Cross Validation on training dataset:



```
Fitting the classifier to the training set
Predicting pokemon names on the testing set
report of SVM:
             precision    recall  f1-score   support

       bear       0.86      0.75      0.80         8
     canary       0.64      1.00      0.78         7
    dolphin       1.00      0.67      0.80         6
       frog       1.00      0.33      0.50         3
      tiger       0.71      0.83      0.77         6

avg / total       0.82      0.77      0.76        30

report of KNN:
             precision    recall  f1-score   support

       bear       0.78      0.88      0.82         8
     canary       0.67      0.86      0.75         7
    dolphin       0.83      0.83      0.83         6
       frog       1.00      0.33      0.50         3
      tiger       0.80      0.67      0.73         6

avg / total       0.79      0.77      0.76        30
```

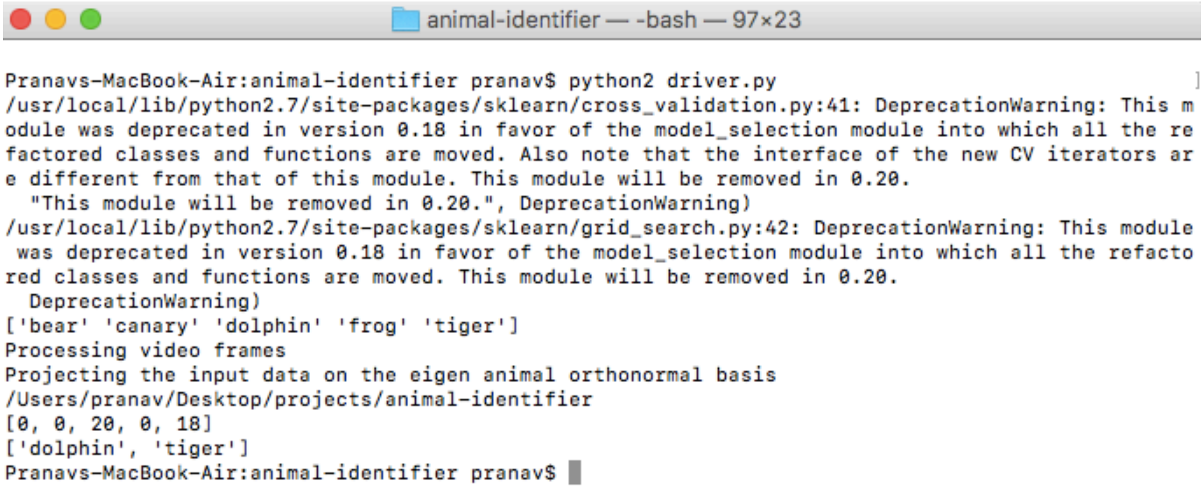
Figure 2

The above image is the classification report of one iteration of K-Folds Cross Validation (‘k’ = 4). The F1-Score obtained from this report, along with similar reports for other runs were used were to determining the weights of the SVM and K-NN models.

For this run, SVM had perfect performance for identification of frogs and dolphins, and less than satisfactory performance for canaries. K-NN had perfect performance for identification of frogs, and less than satisfactory performance for canaries.

3) Prediction of animals given video input:

- As shown in the figures (Figure 3, 4 and 5) below, the program first prints the animal classes.
- The second output line indicates start of video processing.
- The third line indicates start of dimensionality reduction module.
- The next line prints home directory of the project.
- The following line prints as output a list of the number of frames of the video containing the given animal (the animals are mapped alphabetically to the index of this list).

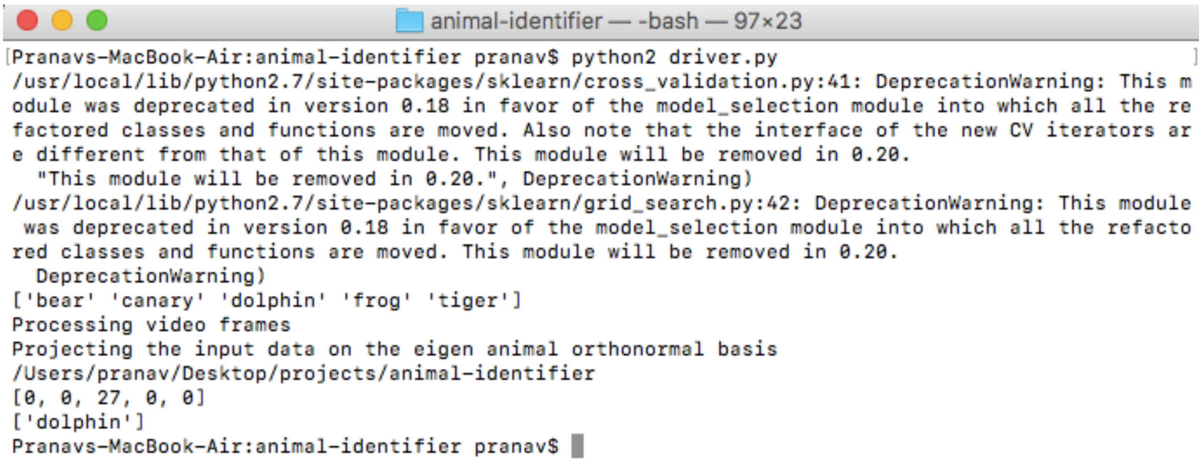


```
Pranavs-MacBook-Air:animal-identifier pranav$ python2 driver.py
/usr/local/lib/python2.7/site-packages/sklearn/cross_validation.py:41: DeprecationWarning: This m
odule was deprecated in version 0.18 in favor of the model_selection module into which all the re
factored classes and functions are moved. Also note that the interface of the new CV iterators ar
e different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/usr/local/lib/python2.7/site-packages/sklearn/grid_search.py:42: DeprecationWarning: This module
 was deprecated in version 0.18 in favor of the model_selection module into which all the refacto
red classes and functions are moved. This module will be removed in 0.20.
  DeprecationWarning)
['bear' 'canary' 'dolphin' 'frog' 'tiger']
Processing video frames
Projecting the input data on the eigen animal orthonormal basis
/Users/pranav/Desktop/projects/animal-identifier
[0, 0, 20, 0, 18]
['dolphin', 'tiger']
Pranavs-MacBook-Air:animal-identifier pranav$ 
```

Figure 3

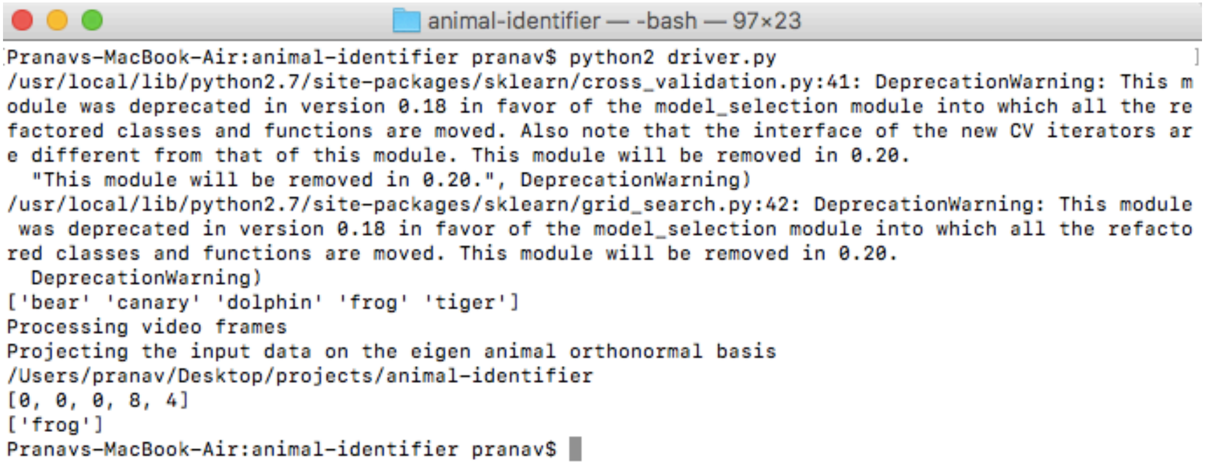
In Figure 3, the video used contains both dolphins and tigers. Of the 38 frames the classifier could predict with desired confidence, 20 frames were identified to be dolphins and 18 frames were identified to be tigers. Since these numbers are with 70% range of each other, the classifier predicts that the video contains both dolphins and tigers.

In Figure 4, the video used contains dolphins. Of the 27 frames the classifier could predict with desired confidence, all 27 were identified to be dolphins, which agrees with the expected output. The classifier behaves well for the given input video.



```
Pranavs-MacBook-Air:animal-identifier pranav$ python2 driver.py
/usr/local/lib/python2.7/site-packages/sklearn/cross_validation.py:41: DeprecationWarning: This m
odule was deprecated in version 0.18 in favor of the model_selection module into which all the re
factored classes and functions are moved. Also note that the interface of the new CV iterators ar
e different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/usr/local/lib/python2.7/site-packages/sklearn/grid_search.py:42: DeprecationWarning: This module
 was deprecated in version 0.18 in favor of the model_selection module into which all the refacto
red classes and functions are moved. This module will be removed in 0.20.
  DeprecationWarning)
['bear' 'canary' 'dolphin' 'frog' 'tiger']
Processing video frames
Projecting the input data on the eigen animal orthonormal basis
/Users/pranav/Desktop/projects/animal-identifier
[0, 0, 27, 0, 0]
['dolphin']
Pranavs-MacBook-Air:animal-identifier pranav$ 
```

Figure 4



```
Pranavs-MacBook-Air:animal-identifier pranav$ python2 driver.py
/usr/local/lib/python2.7/site-packages/sklearn/cross_validation.py:41: DeprecationWarning: This m
odule was deprecated in version 0.18 in favor of the model_selection module into which all the re
factored classes and functions are moved. Also note that the interface of the new CV iterators ar
e different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/usr/local/lib/python2.7/site-packages/sklearn/grid_search.py:42: DeprecationWarning: This module
 was deprecated in version 0.18 in favor of the model_selection module into which all the refacto
red classes and functions are moved. This module will be removed in 0.20.
  DeprecationWarning)
['bear' 'canary' 'dolphin' 'frog' 'tiger']
Processing video frames
Projecting the input data on the eigen animal orthonormal basis
/Users/pranav/Desktop/projects/animal-identifier
[0, 0, 0, 8, 4]
['frog']
Pranavs-MacBook-Air:animal-identifier pranav$ 
```

Figure 5

In Figure 5, the video used contains frogs. Of the 12 frames the classifier could predict with desired confidence, 8 frames were identified to contain frogs and 4 to contain tigers. The predicted class is still 'frog' because 4 is not within 70% range of 8. The performance is satisfactory, but not ideal, for this particular.

Applications:

An application that is immediately obvious is identification of animals in the wild in near real time. Zoologists and wildlife enthusiasts could point their mobile phone cameras at any animal, and use the live feed from there as input to the animal identifier to determine which animal it is. More information about the animal could be displayed once the animal is identified, as is done by the Pokedex. On more refinement, the identifier could also determine the exact species of the animal, which could help in discovery of new species in cases where the animal cannot be identified by the classifier.

Possible improvements:

- Increasing the number of animal classes to enhance utility
- Increasing the number of training data points, to improve classification accuracy, and to allow recording of more frames from input video
- Incorporating an additional classifier on the basis of the sound the animal makes. This classifier can be weighted with the video classifier to improve results.
- Using more principal components for better reconstruction of original images.