CIS 6930 Sparse Coding – Project Report
Visual Object Detection using deformable sparse coding

Shang Wang
Ju yin Chung

1. Goals and problem statement

Main Goal :

Locate the target in the image.

Problem :

Use gabor wavelet to form dictionaries for training images and use Orthogonal matching pursuit to reconstruct our target images. After obtaining the coefficients of training images, we need to compute OMP for natural images so as to classify between target and non-target. the gabor wavelet is easy to be influenced by location, the main drawback of gabor wavelets is its high complexity both in memory and computational time.

2. Literature review-

Gabor Wavelet:

Wavelet transform could perform multi-resolution time-frequency analysis. The tunable kernel size results in different time-frequency resolution pair and the size is related to the analytical frequency. For example, smaller kernel size (in time domain) has higher resolution in time domain but lower resolution in frequency domain, and is used for higher frequency analysis; while bigger kernel size has higher resolution in frequency domain but lower resolution in time domain, and is used for lower frequency analysis. This great property makes wavelet transform suitable for applications such as image compression, edge detection, filter design, and some kinds of image object recognition, etc. Mathematical and empirical motivation: Gabor wavelet transform has both the multi-resolution and multi-orientation properties and are optimal for measuring local spatial frequencies. Besides, it has been found to yield distortion tolerance space for pattern recognition tasks..

Orthogonal Matching Pursuit:

OMP is an iterative greedy algorithm that selects at each step the column, which is most correlated with the current residuals. In this paper, we present a fully data driven OMP algorithm with explicit stopping rules. It is shown that under conditions on the mutual incoherence and the minimum magnitude of the nonzero components of the signal, the support of the signal can be recovered exactly by the OMP algorithm with high probability.

Deformable model:

The deformable part model supposes that an object is comprised by its parts. Therefore, the detector will find a match of the whole at first, and then use the part models to fine-tune the result. AND-OR Template (AOT) is a probabilistic image model for object detection. AND means composition of constituent parts into a larger part while OR means deformation, and different ways to compose smaller parts into a larger one. The AOT model is easy to be trained from images and no bounding boxes labeling is needed. In this paper, we use deformable model, based on sparse coding, to detect objects.

Support Vector Machine (SVM):

Support vector machine (SVM) is a principled technique to train classifiers in a supervised manner. SVM minimizes a bound on the empirical error and the complexity of the classifier simultaneously. Therefore, they can learn in sparse, high dimensional spaces with only a few training examples compared to neural network.

Competing methods-

Generative algorithms comprise of a probability model for the pose variability of the objects and an appearance model.

Parameters of the model can be estimated by training data and decisions are made of ratios of posterior probabilities.

Discriminative approaches usually build up classifiers to discriminate between images containing the object and those not containing the object.

The deformable part model supposes that an object is comprised by its parts. Therefore, the detector will find a match of the whole at first, and then use the part models to fine-tune the result.

As a result, due to the high variation of different individual, we use the last deformable model.

State of the art –

Although there are many existing classification methods in the current technology, it is still hard to find every kinds of classification model which is the most suitable to the specific case. The deep learning approach has gained many outstanding results recently, as long as the convolution network learns the data, it can do accurate classification as well. Therefore, in the future, I think combining those two advantages of the new and the old mothed will spark new creative method.

3. What worked, and what didn't. Discuss details along with experiments when applicable Some of the detection work, some do not.

Part1 : Classification

Experiment Design -

We use two kinds of SVM model to classify and generate different results in different condition. One is the SVM function in matlab that we can choose our kernel and adjust several parameters by ourselves and the other is LIBLinear which is an open source library for large-scale classification and experiments demonstrate that LibLinear is very efficient on large sparse data sets.

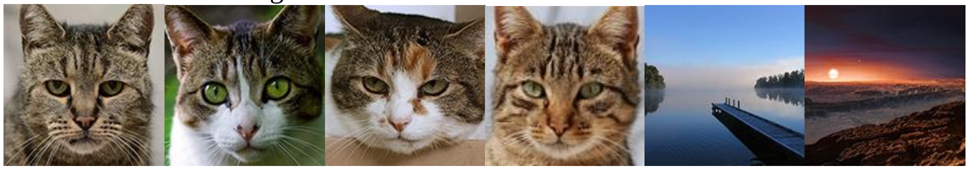
Analyzation -

First, SVM model :

```
itsvm(input, label, 'KernelScale', 'auto','Standardize',true,'OutlierFraction',0.01 )
```

We use svm score of svm function in matlab to judge a picture is a cat face or not. If svm score is greater than 0, the corresponding picture is a cat face otherwise not.

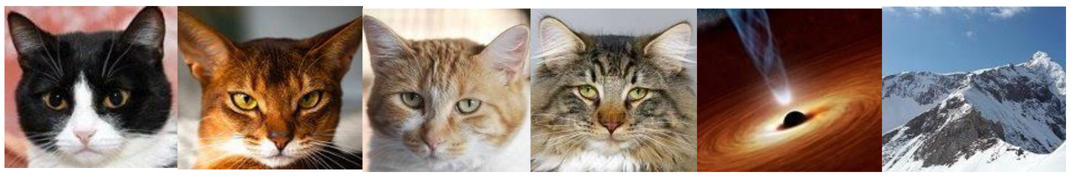
Cat face and nature image classification:



score =
0.3657 -0.3259 1.0230 0.0026 -1.2737 -1.0631

You will see some of the cat face can be recognized with a high score but some may not. The nature image that does not contain cat feature will present a good result.
We found that the training images has done a great influence on the result. We will talk about this at the end of this discussion.

Second, the LibLinear SVM model, we can see that it can detect most of the cats butsomecats with great variation will not be labeled as the same class. Even the nature image sometimes will be regarded as cat images. Because we input relatively less nature images, they can not be defined in a same class easily.



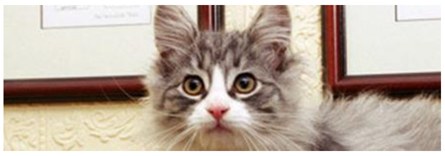
```
predict_label=
1          1          1          -1          1          -1
```

Part 2: Detection

SVM model :

In the last part, we know how to do classification and in this part, we are going to talk about detection. This is harder than classification. Classification is not good enough so detection cannot be better than that. Furthermore, some of the cat hair is likely to be regard as cat face. So the result should be worse.

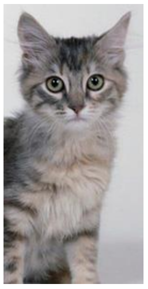
A fail detection Ex.



```
detectscore =
-1.1708          -0.6241          -0.4282
```

From left to right three 150 X 150 images are detected. And the result is not good because all three images score are less than 0. So it fails.

A better detection Ex.



```
detectscore2 =
0.3276          -0.0480          -0.4577
```

From top to bottom 3 images in the picture are detected. And the scores are shown from left to right. We can see that the top one is detected. This is a relatively success example.

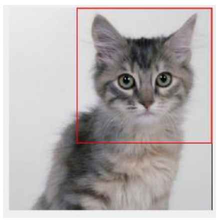
We rescale this picture to 150 X 150, 225 X 225, 300 X 300 and 375 X 375. And we do 150 X 150 image match for every rescale image. Say we do 1, 4, 9 and 16 times for the corresponding rescale images. (1 for 150 X 150, 4 for 225 X 225…. 75 pixels as an interval.

We use the max score as the matching result:
The results are as followed:

[1]150 X 150: Not detected.

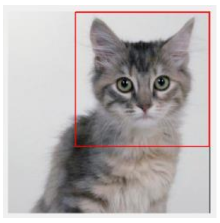
```
detectscoretemp =

  -0.0406
```

[2]225 X 225: detected:



```
detectscoretemp =

   -0.2670
    0.4614
   -0.5976
   -0.5012
```

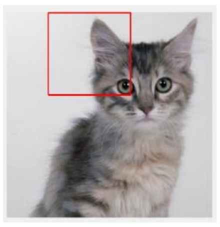
[3]300 X 300: detected.



```
detectscoretemp =

   -1.1537
   -0.4088
   -0.5386
    0.1217
   -0.5614
    0.0620
   -0.4908
   -0.0678
   -0.9221
```

And you see that the max score is **0.1217,** but it is still less than what we get in 225 X 225 image. So we still use the last one as the result.

[4]375 X 375: detected but was the wrong one.



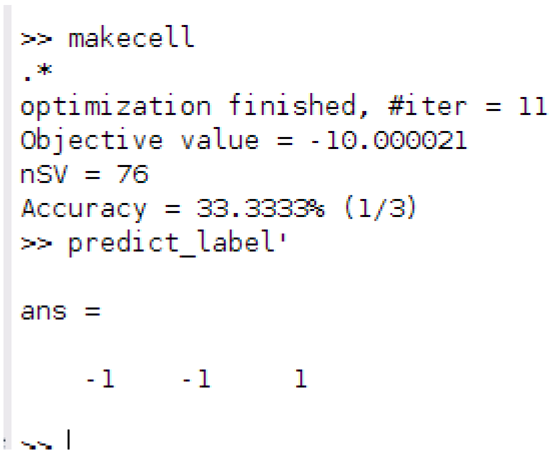
```
detectscoretemp =

  -1.2580
   3.8617
  -0.4346
  -0.7732
  -1.6502
   0.1444
   0.4735
  -0.9241
  -0.9005
  -0.9201
  -0.6010
  -0.6178
  -1.0429
  -0.3698
  -0.5049
  -0.7951
```

3.8617 is the biggest score but it is surely not the cat face. At this time, this algorithm fails. One reason that leads to this result is that the training image sets is not large enough. So it doesn't reflect every case that may happen.

Another reason is that the svm result is not good enough which somehow caused by a small training images set. You see 3.86 is much bigger than 1 which means it's much more inside the hyper plane than other images. But the images lies in that hyper plane should fail. So the result is not good enough.

LibLinear model:

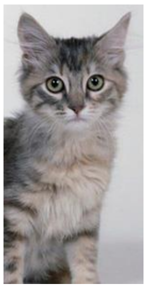
A fail detection Ex.

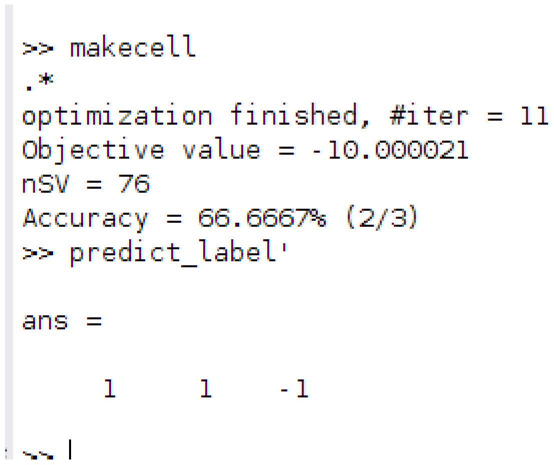


```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 33.3333% (1/3)
>> predict_label'

ans =

    -1    -1     1

>> |
```

From left to right three 150 X 150 images are detected. And the result is not good because the left is detected as a cat, but actually there is only body part. So it fails.

A better detection Ex.





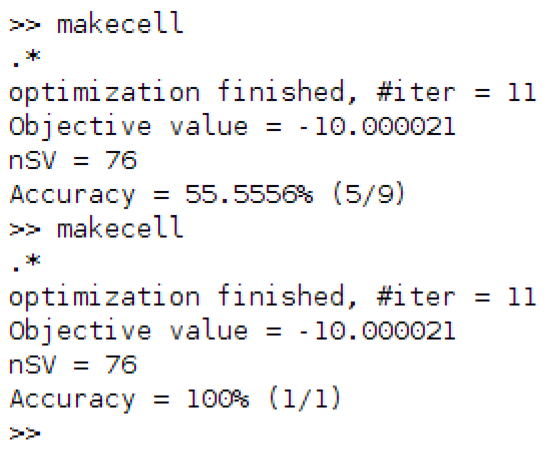
```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 66.6667% (2/3)
>> predict_label'

ans =

     1     1    -1

>> |
```

From top to bottom 3 images in the picture are detected. And the ans are shown from left to right. We can see that the top one is detected. This is a relatively success example.

We rescale this picture to 150 X 150, 225 X 225, 300 X 300 and 375 X 375. And we do 150 X150 image match for every rescale image. Say we do 1, 4, 9 and 16 times for the corresponding rescale images. (1 for 150 X 150, 4 for 225 X 225…. 75 pixels as an interval.

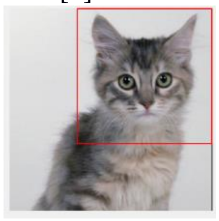
We use the max score as the matching result:
The results are as followed:

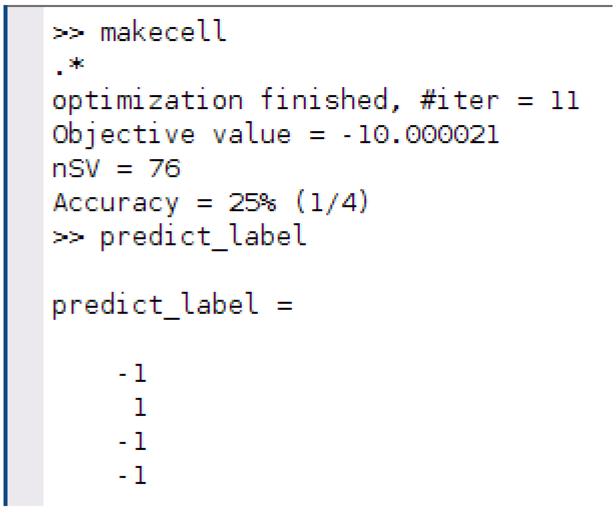
[1]150 X 150: Detected.



```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 55.5556% (5/9)
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 100% (1/1)
>>
```

[2] 225 X 225: Detected correctly:

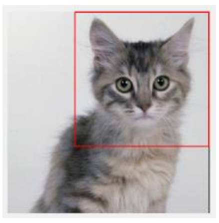




```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 25% (1/4)
>> predict_label

predict_label =

    -1
     1
    -1
    -1
```

The cat face is approximately the ¼ times the image size, so the size 225x225 and detect for 4 times is the most successful.

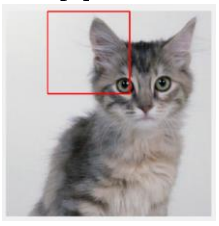
[3] 300 X 300: Detected not accurately.



```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 44.4444% (4/9)
>> predict_label'

ans =

    -1    -1    -1     1    -1     1     1     1    -1

>>
```

In this case, the detection window will not fit the cat face with sliding through the image every 75 pixels. Moreover, it will be misled by the body part of the cat.

[4] 375 X 375: Detected not accurately.



```
>> makecell
.*
optimization finished, #iter = 11
Objective value = -10.000021
nSV = 76
Accuracy = 50% (8/16)
>> predict_label

predict_label =

    -1
    -1
     1
    -1
    -1
     1
     1
    -1
     1
    -1
     1
    -1
    -1
     1
     1
     1
```

If we scale the image into 375 to 375, although we have more part to detect, the ratio between the cat face and the whole image is not correct. Everytime it scan the image it would be part of the body or part of the face. So, it hard to judge which is the real cat face.

I think we can improve this result by sliding the window every smaller amount of pixel. But we do not have enough time to complete this at this certain stage. Moreover, I think we should add a judgment condition after detection.

4. Learning outcomes – what you learned from this project.

Gabor Wavelets and deformable template is suitable for extraction biological features.
How can we use SVM to do image classification.
Different SVM model will cause different classification results.

5. For team projects: specific details of contributions of each team member. A single report for the team is sufficient, but you may choose to submit separate reports if you wish to concentrate on your own contribution to the project.

Shang Wang: Matlab svm, omp and detecting coding. Cat image training.
Ju yin Chung: Matlab libsvm coding and omp coding. Nature image training. Learn deformable template.

6. Thoughts on future work: discuss on ways to extend the project.

1. We will need to increase amount of the training sets and speed up calculating. If we increase the layers of training SVM, I believe the accuracy will increase. We can use cloud computing to make it better.
2. Scanning algorithm is not very good. It is brutal force in this point. DP is a better choice.
3. I don’t think matlab is a good software to do image processing. We should use other IDE rather than matlab.

7. References: bibliography and a concise acknowledgement of resources for software and data.

“LIBLINEAR: A Library for Large Linear Classification” for the us of LIBLINEAR” teaches us how to use LibLinear model, for example generating the data form the model accepts.

8. Expiations about the code

If you want to get the sparse code of the image, you will need to run gabor.m.
The result will store in .txt file. If you will need to remove comments in the code.
If you want to use svm to locate the detected image, you will need to run svm.m.
If you want to see the result of liblinear model, you should run makecell.
The first part of makecell is to make the txt data into the cell form that liblinear will accept, and the second part is to detect.

9. Training images

You will find 40 cat faces and 25 nature images in the folder.