Computer Vision task to detect cancer in skin lesion images

Anirudh Singh Sisodia

F215611

22COP507

Table of Contents

High Level Design

Block Diagram

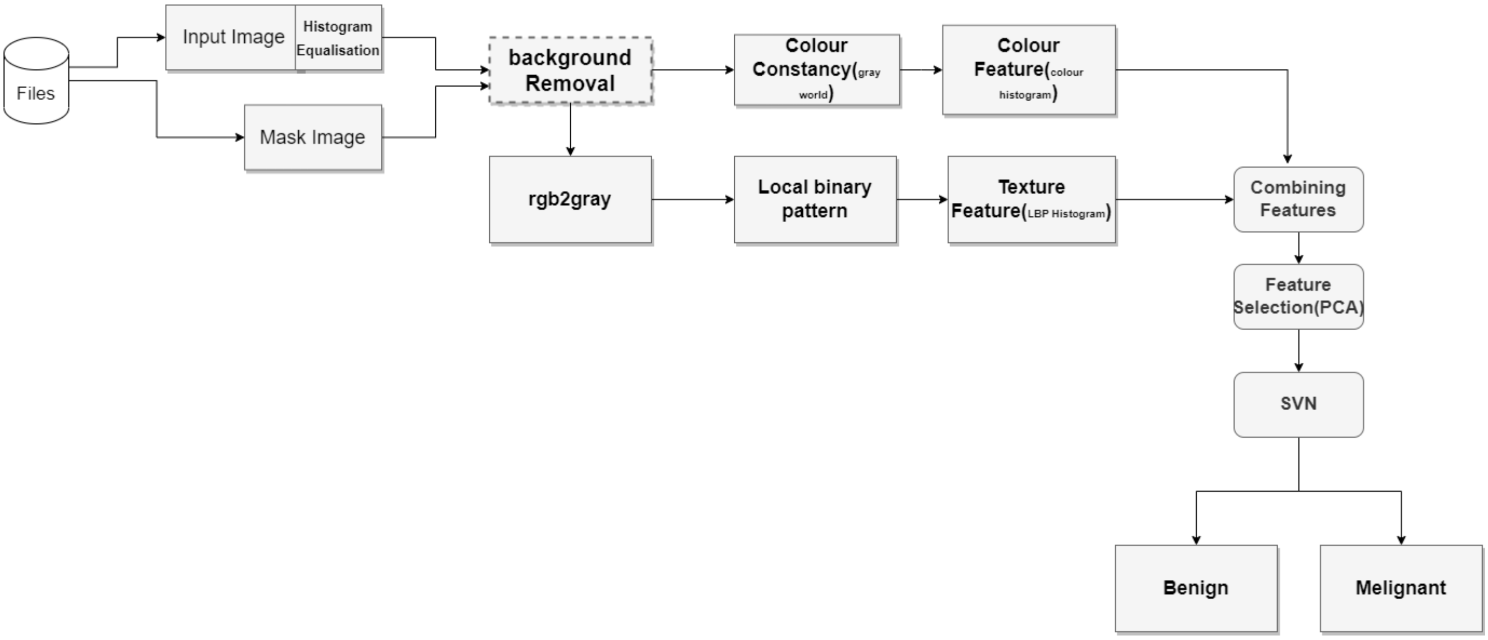


Figure 1: Classification Model

Description

The whole pipeline is divided into 3 sections:

1. Feature Extraction
2. Feature Selection
3. Classification

Feature Extraction

Input images are read into variables and there are 2 features on which the pipeline is focusing and those are colour and texture. Following are the main components of feature extraction phase.

- Background Removal: Here I used the mask images to remove the background of the colour images and only have the part of the image we are interested in to remove any unwanted noise
- Colour Constancy(**greyWorld.m**[3]): Then I applied a colour constancy algorithm developed in lab that implements Grey World.
- Colour Feature(**RGB_hist.m**[1]): Here I computed the colour histogram using the function I developed in lab and reshaped the output to give a row vector for each image.
- Rgb2gray: Here before extracted texture feature I converted to image from colour to grayscale as LBP does not explicitly use colour information.
- Local Binary Pattern: Here to find the texture of the neighbouring pixels I found the local binary pattern for each image.
- Texture Feature(**LBP.m**[2], **hist_lbp.m**[4]): To extract the texture feature I called another function that I developed in lab to get the histogram of LBP for each image and reshape it to a row vector.

Feature Selection

After the colour and texture features are selected these features are transposed and concatenated in one matrix and then fed to PCA algorithm to for dimensionality extract new feature vectors which contribute to the maximum variation in this data.

- PCA: Here the PCA gives the new adjusted features i.e. Principal Components to focus on for getting maximum variations of the fed data points. I see that the PC1 contributes to 99.8% and PC2 contributes to 0.2% of the variation in data so I Select PC1 as a single feature for this problem.

Classification

Provided SVM code is used for classification

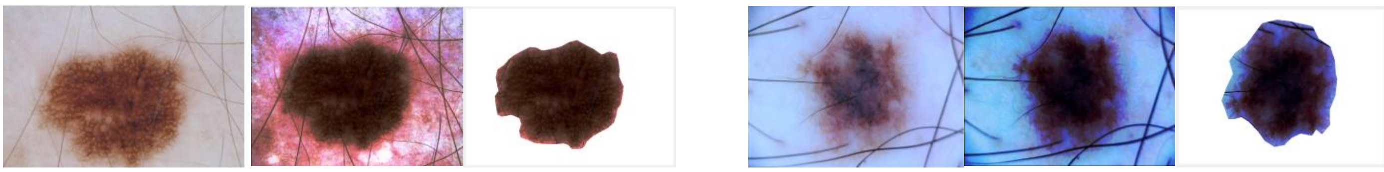
- SVN: The code is fed with **imagefeatures** extracted and selected in previous steps along with the provided **groundtruth** and it gives out the confusion matrix and the predictions for the input data. In the last step I use these values are used to calculate accuracy, precision, sensitivity, specificity and miss classification percentages.

System development and implementation

MATLAB Implementation details

- I start with the unziping the lesion images and mask images and store them in **images** and **mask_images** variables and apply histeq() to all the **images** while I am doing it. I also store the extract the ground truth from the txt file to **groundtruth** variable.
- Now we use the mask image(mask_images) to remove the background from the lesion images to extract the ROI of the images to roi_images variable and then extract features from them. We did This because there are a lot of images that have noise, unwanted hairs etc which makes the classification task difficult. We do this by setting the intensities to 255 of the colour images where the intensity is 0 in mask images to 255.

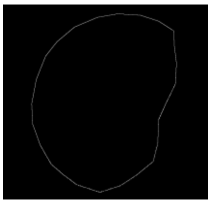
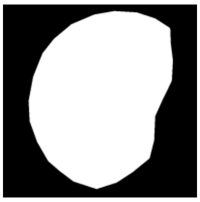
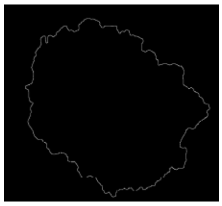
Images extracted from zip => **histeq(image)** => **background removed**



- Next step is extracting features from these images. I have extracted colour and texture features in the form of colour histograms and Local Binary Pattern histograms respectively. For Colour Histograms I used a functions I developed in lab **RGB_hist.m[1](explained below)**, to get 8 X 8 X 8 3-D histogram which I convert to 1 X 512 to get these as a feature vector. For Texture I use **LBP.m[2](explained below)** and **hist_lbp.m[4](explained below)** to get 1 X 256 feature vector for this. I also tried to extract shape feature in the form of boundary extraction from mask images and then calculating the boundary length(P) and bounded area(A) and used the formulae to get circularity index.

  $CI = 4A\pi/P^2$

  This was very computationally expensive and took a lot of time. The result was not that good so dropped this idea and continued with 2 features.

- Then I combine these features together in a 200 X 768 feature matrix. I fees this matrix to PCA to get one PC1 with 99 % contribution to variance.
- Then I feed these features to SVM classifier code where I put the PC1 to imagefeatures variable along with ground truth as groundtruth to get the confusion matrix and all the predictions.
- The confusion matrix which has 4 values. First row of the matrix have true positives and false positives and the second row has false negatives and true negatives. Data like this can tell us about how the model performed as we can calculate `accuracy, misclassification rate, precision, sensitivity and specificity.` Here I find all these and given below is the result I got for my CW.

Confusion matrix

TP	FP
FN	TN

=

71	29
45	55

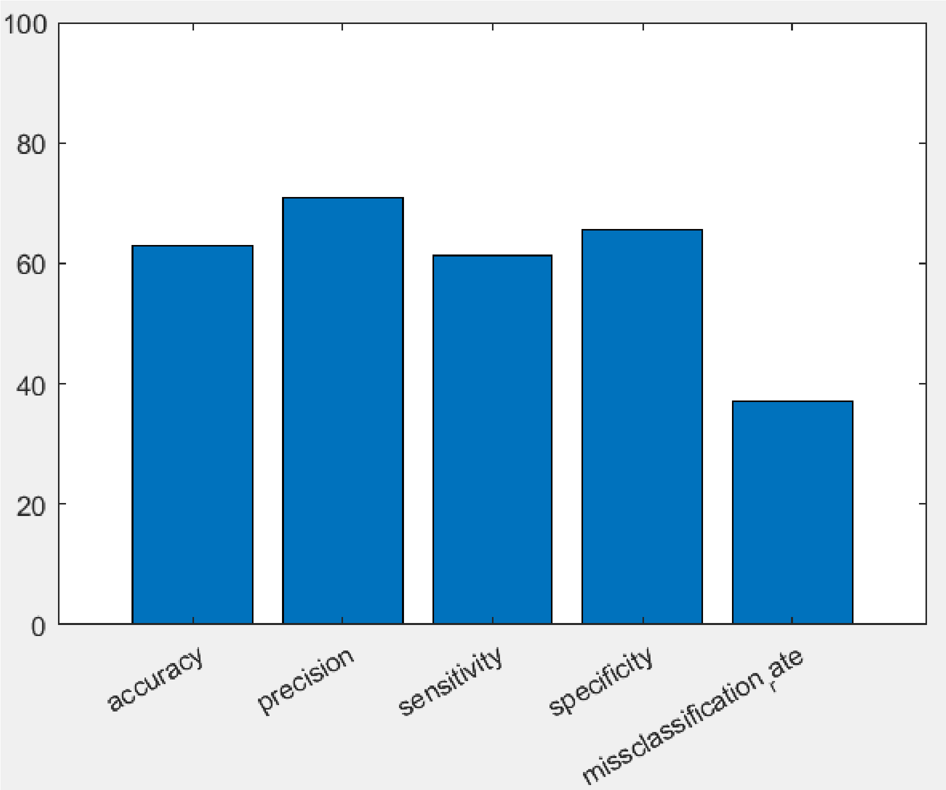
Metric	**Percentage**
Accuracy	63%
Precision	71%
Sensitivity	61.21%
Specificity	65.48%
Miss Classification	37%

Description employed algorithms

- **Histogram Equalisation**: For Pre-processing the images I applied **histeq()** function of MATLAB for histogram equalisation of all the images before feeding them in the feature extraction phase.
- **Gray World**: I developed a script to perform Gray World colour constancy algorithm where we first divide the image in its 3 channels and find the average pixel ($E_{R,W}$, $E_{G,W}$, $E_{B,W}$) intensity for each channel and then we find the illumination (E_{R}) from the image which is the mean of these 3 intensities. Then we find the $E_{R,W}/E_{R}$ and multiply each origical channel to get the colour constancy channels and join them. I developed this function in the script **greyWorld.m**[3]
- **Colour histogram:** I used the technique in which we select 8 as total bins in which each pixel from each channel will falls then we reshape everythin to 1 X 512 feature vector. I used the function present in the file **RGB_hist.m**[1] which I developed in the lab sessions.
- **Local Binary Pattern image:** We convert the RGB image to grayscale and Threshold each 3 X 3 matrix of the image data depending on the centre element of this matrix. Find the 8 bit binary code for each matrix and then convert it to decimal format and that's the new intensity value of the centre pixel. I developed this function in the script **LBP.m**[2]
- **hist_lbp.m**[4]: Accepts LBP image and return a histogram of that image using 256 bins for 256 intensities.

System performance and system evaluation

After I get the all the predictions and the confusion matrix I Plot the bar chart after calculating some metrics to analyse the performance. I got **63%** accuracy on this small dataset.

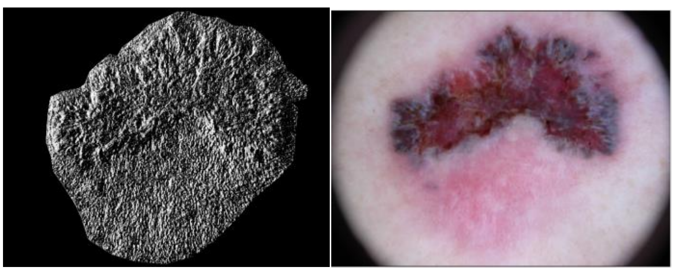


Now let's see at some of the classifications and how did the model performed on the images. Now let' s see at a couple of cases in which I made changes to the pipeline based on the results I got.

Observations

- Earlier when I first started solving this problem I used colour histogram without pre-processing(histeq) the image and the model was not able to classify at all then I used the segmented image to remove the background which improved the classification immediately by 5% and sensitivity by 8 % this shows that noise and hairs were occluding the lesion images and also when I was not using histogram equalisation there were many miss classifications and after using histeq the accuracy improved by 5.5 % and sensitivity by 5.5%.
- I also tried to use LBP feature without removing the background and found that it reduces the accuracy by 6% and sensitivity by 5.5%.
- The False negative value in confusion matrix is quite high which means that there are some image features that are not that complex to represent the image with cancer in the patient for example if we look at some LBP images of False Negative images:

Case: Incorrectly classified as benign but was melanoma.



We can See that the LBP image doesn't really differentiate with the texture of the affected region, This might be the reason for this wrong classification

Case: Correctly classified as malignant

Here we can see that the affected region was clearly identified by the LBP feature.

Appendix

1. **RGB_hist.m**

```
function hist = RGB_hist(image)
    totalbins = 8;
    binwidth = 256/totalbins;
    hist = zeros(totalbins,totalbins,totalbins);
    R = image(:,:,1);
    G = image(:,:,2);
    B = image(:,:,3);

    Rhist = floor(double(R)/binwidth)+1;
    Ghist = floor(double(G)/binwidth)+1;
    Bhist = floor(double(B)/binwidth)+1;

    [r1,c1] = size(Rhist);
    [r2,c2] = size(Ghist);
    [r3,c3] = size(Bhist);

    bin_image = [reshape(Rhist,r1*c1,1),reshape(Ghist,r2*c2,1),reshape(Bhist,r3*c3,1)];

    for i = 1:length(bin_image)
        a=bin_image(i,1);
        b=bin_image(i,2);
        c=bin_image(i,3);
        hist(a,b,c)=hist(a,b,c)+1;
    end
    hist=hist/sum(sum(sum(hist)));
end
```

2. **LBP.m**

```
function image = LBP(image)
    image2 = padarray( image , [1,1],'both');
    [r,c] = size(image);
    for i = 2:r-1
        for j = 2:c-1
            t = image(i,j);
            b8 = uint8(image2(i+1,j)>t);
            b7 = uint8(image2(i+1,j-1)>t);
            b6 = uint8(image2(i,j-1)>t);
            b5 = uint8(image2(i-1,j-1)>t);
            b4 = uint8(image2(i-1,j)>t);
            b3 = uint8(image2(i-1,j+1)>t);
            b2 = uint8(image2(i,j+1)>t);
            b1 = uint8(image2(i+1,j+1)>t);
            bit8 = uint8(b8*2^7+b7*2^6+b6*2^5+b5*2^4+b4*2^3+b3*2^2+b2*2+b1);
                 image(i,j)=bit8;
        end
```

```
    end

end
```

3. **greyWorld.m**

```
function gw = grayWorld(image)
    [r,c,ch] = size(image);
    R = image(:,:,1);
    G = image(:,:,2);
    B = image(:,:,3);

    avgR = sum(sum(R))/(r*c);
    avgG = sum(sum(G))/(r*c);
    avgB = sum(sum(B))/(r*c);

    avgimage = mean([avgR avgG avgB]);

    newR = (avgimage/avgR)*(R);
    newG = (avgimage/avgG)*(G);
    newB = (avgimage/avgB)*(B);

    gw(:,:,1) = newR;
    gw(:,:,2) = newG;
    gw(:,:,3) = newB;
end
```

4. **hist_lbp.m**

```
function hist = hist_lbp(image)
        lbp = LBP(image);
        hlist = unique(lbp);
        hist = zeros(256,1);
        for i = 1:256
            b = sum(lbp==i-1,'all');
            hist(i) = b;
        end
end
```

References

- Computer Vision Lab.
- Computer-aided Diagnosis of Melanoma Using Border and Wavelet-based Texture Analysis Rahil Garnavi, Member, IEEE, Mohammad Aldeen, Senior Member, IEEE and James Bailey.
- Color Indexing MICHAEL J. SWAIN Department of Computer Science, University of Chicago, Chicago, IL 60637 DANA H. BALLARD Department of Computer Science, University of Rochester, Rochester, NY 14627 Received January 22, 1991. Revised June 6, 1991.
- https://uk.mathworks.com/matlabcentral/answers/301666-how-to-find-length-or-perimeter-of-a-irregular-boundary#accepted_answer_233533
- https://uk.mathworks.com/matlabcentral/fileexchange?q=skin+lesions+classification
- MATLAB Documentations
- A comparative study for classification of skin cancer Tri Cong Pham1 School of Computer Science and Engineering, Thuyloi University 175 Tay Son, Dong Da, Hanoi, Vietnam phtcong@tlu.edu.vn Giang Son Tran 1 ICTLab, University of Science and Technology of Hanoi, VAST* tran-giang.son@usth.edu.vn Thi Phuong Nghiem 1 ICTLab, University of Science and Technology of Hanoi, VAST* nghiem-thi.phuong@usth.edu.vn Antoine Doucet L3i Lab, University of La Rochelle Av M. Crepeau, 17042 La Rochelle, France

antoine.doucet@univ-lr.fr Chi Mai Luong1 Institute of Information Technology VAST* lcmai@ioit.ac.vn Van-Dung Hoang** QuangBinh University Dong Hoi, Quang Binh, Vietnam dunghv@qbu.edu.vn

- https://blogs.mathworks.com/loren/2015/06/24/how-do-you-modify-the-background-of-an-image/?s_tid=answers_rc2-1_p4_MLT