Information Technology Course

Module Software Engineering

by Damir Dobric/Andreas Pech

Schema Image Classification

Md Abu Syeem Dipu 1324442

Md Mahbub Ul Alam 1324358

Tanvir Hasan 1323951

Mir Mehedi Hasan Rayhan 1324345

Md Mijanur Rahaman 1324617

Md Zahidul Islam 1323948

Abstract - The image classification is a classical problem of image processing; machine learning fields and it is a complex process that may be affected by many factors. In this paper we study the image classification using deep learning and emphasis is placed on the summarization of major advanced classification approaches and the techniques used for improving classification accuracy. In this paper we are trying to categorize the images by the similarity of them Based on Hierarchical Temporal Memory.

a. Introduction

Image recognition is a term for computer technologies that can recognize certain people, animals, objects or other targeted subjects using algorithms and machine learning concepts. This is the process of identifying and detecting an object or a feature in an image. Human and animal brains make vision easily. Human can recognize an object very fast. But Recognizing an object is a critical task for a machine. Past few decades machine learning technology has a significant success on object recognition. A key component of an image classification system is a feature extractor, capable of detecting useful pieces of information that can matched with another picture and classified as same group.

The goal for the image classification is classified the same image using some methods or Algorithms and grouped the same image together. Here we took some different types image as input and binarize them. Use spatial pooler then calculate hamming distance. As output make group with similar images in a folder. We procedure of implementation is using dotnet Core 3.0 platform.

b. Methods

Image recognition is done in many different ways. The schema Image Classifier is a useful classifier for predicting the output from the input image and the similar image grouped in different directories. Figure 1 shows the process flow. First images send to binarizer that it can convert them into binary bits and hamming distance compute the binary image. Then we trained the data and grouped it by hamming distance.

Flow chart (Figure 1) shows the hole process step by step.

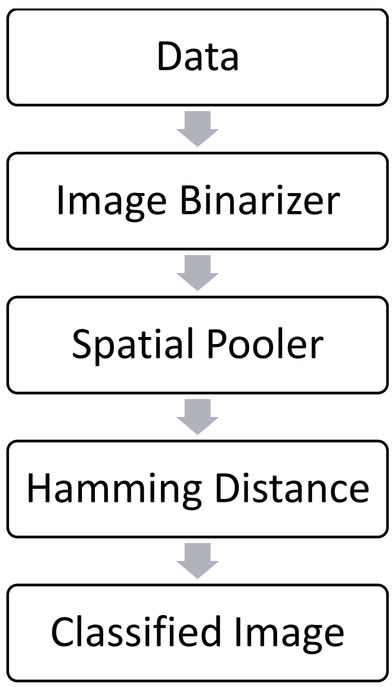


Figure 1: Image Classification Working Flow

i. Image Binarization

A binary image is a digital image that has only two used for a binary image are black and white, though any two colors can be used. The color used for the objects in the image is the foreground color while the rest of the image is the background color.

- Get an image from file.
- Create equivalent binary

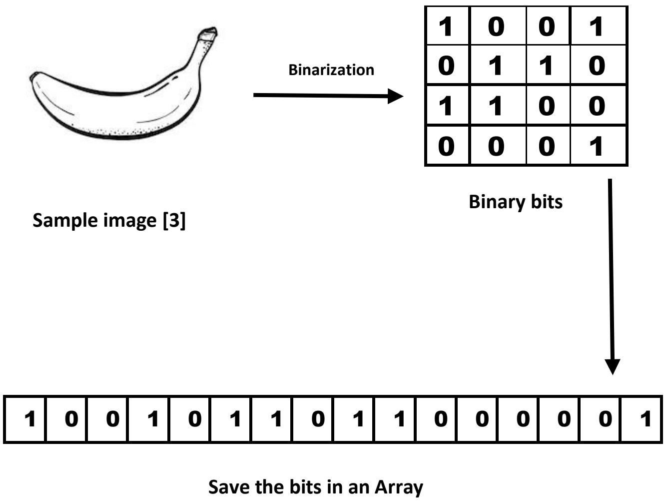


Figure 2: Image Binarization Process

The figure (Figure 2) demonstrates the process how an image converts into binary format. This is the process of taking a grayscale image and converting it to black-and-white, essentially reducing the information contained within the image from 256 shades of gray to 2: black and white, a binary image. This is sometimes known as image thresholding, although thresholding may produce images with more than 2 levels of gray. It is a form or segmentation, whereby an image is divided into constituent objects. This is a task commonly performed when trying to extract an object from an image. However, like many image processing operations, it is not trivial, and is solely dependent on the content within the image. [2]

ii. Spatial Pooler

The SP consists of three phases, namely overlap, inhibition, and learning. In this section, the three phases will be presented based off their original, algorithmic definition. This algorithm follows an iterative, online approach, where the learning updates occur after the presentation of each input. Before the execution of the algorithm, some initializations must take place.[1]

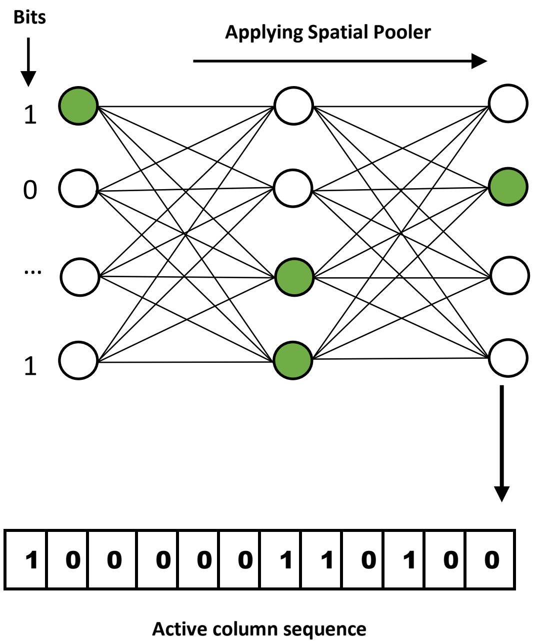


Figure 3: Spatial Pooler Applying Process

In the figure (Figure 3) we can see after applying Spatial Pooler we get active column sequence. In the figure (Figure 3) the Green nodes indicate the active bits. We use spatial pooler to train our binarize data. Then we get an active column sequence and store it for further use.

```
foreach input
    do{
        data->SP.compute(activeCols[inp], train=true)
        }while(hamming distance exists)
 save the activeCols[inp] for input
```

iii. Hamming Distance

Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. In other words, it measures the minimum number of substitutions required to change one string into the other, or the minimum number of errors that could have transformed one string into the other. In a more general context, the Hamming distance is one of several string metrics for measuring the edit distance between two sequences.

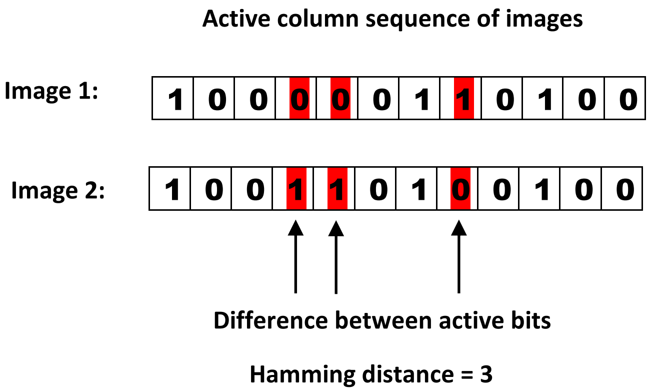


Figure 4: Hamming Distance Calculation

From the figure (Figure 4) we can understand how Hamming distance calculate. According the figure (Figure 4), if we consider to images (Image 1, Image 2), we will get active column sequence of each. The red block (Figure 4) marked the dissimilar active bits and here the value of hamming distance is 3.

We applied hamming distance after training our binary bits of the images with Spatial pooler. Hamming distance measure the active sequence similarity between the images. Images grouping was depended according the given Hamming distance.

```
hamming distance (array A, array B):
    numOfDifferentBits = 0
    for n in range(length(A)):
        if A[n] ! = B[n] and A[n] == 1:
            numOfDifferentBits += 1
    bits1 = Number of 1bit in A
    distance = ((bits1–numOfDifferentBits) *
    100.0 / bits1);
    return distance;
```

iv. Grouping

Grouping is the last stage of our working flow. There the similar images exits in a group directoris together. Process of the Grouping works with the output of the Hmmaing distance.

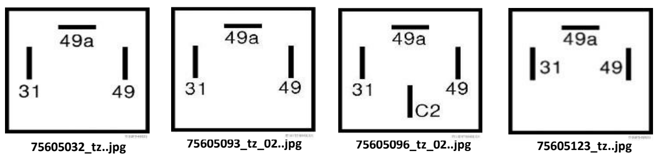
```
seqA,seqB = Active column sequence for image A, B
  grouping:
  for i 1 to N:
      for j i+1 to N:
      if(hammingDistance(seq[i],seq[j]) < theresold)
      write images in the same directory;
  Store Hamming distances to Excel file.
```

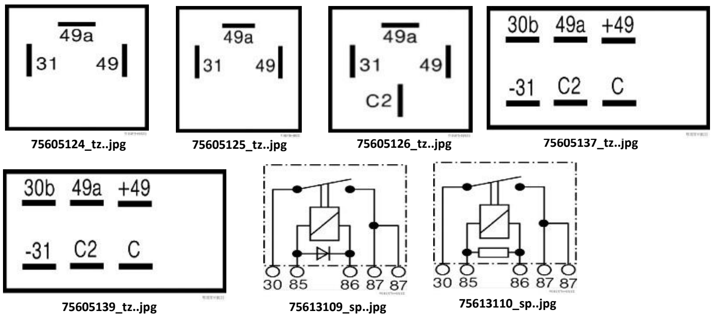
The strings received must be of equal length, hence it is reasonable for us to check that the strings passed in as arguments meet this requirement before any comparison occurs. Next, we would need to compare every character in the first string with the character holding their corresponding position in the second string. This helps us identify the points where differences exist as it is the total count of these instances that gives us the hamming distance. Bear in mind that lowercase a will differ from its uppercase character A, hence all our comparisons must be done in a common case.

c. Results

We used 342 images. We binarized all images then images are transformed and downscaled into input vectors. Then we train spatial pooler used these inputs. Then hamming distance used for calculating the distance between two vectors and grouped them in different directories according to the Hamming distance. Further, Hamming distance are stored in a Excel file called “distance.xlsx”.

Sample images as input:





We use above 11 images as a sample in put in our project. Then we get the Hamming distance between the images (Figure 5). The following table (Figure 5). shows the similarity between those images according the Hamming distance.

Output Table:

	75605032_tz.	75605093_tz_02.	75605096_tz_02.	75605123_tz.	75605124_tz.	75605125_tz.	75605126_tz.	75605137_tz.	75605139_tz.	75613109_sp.	75613110_sp.
75605032_tz.	100	96.25	91.463	89.535	89.535	80	80	80.46	80.46	71.591	71.264
75605093_tz_02.	96.25	100	95.122	93.023	93.023	83.529	83.529	83.908	83.908	75	74.713
75605096_tz_02.	91.463	95.122	100	95.349	95.349	88.235	88.235	87.356	87.356	78.409	78.161
75605123_tz.	89.535	93.023	95.349	100	100	89.535	89.535	90.805	90.805	81.818	81.609
75605124_tz.	89.535	93.023	95.349	100	100	89.535	89.535	90.805	90.805	81.818	81.609
75605125_tz.	80	83.529	88.235	89.535	89.535	100	100	91.954	91.954	85.227	83.908
75605126_tz.	80	83.529	88.235	89.535	89.535	100	100	91.954	91.954	85.227	83.908
75605137_tz.	80.46	83.908	87.356	90.805	90.805	91.954	91.954	100	100	86.364	86.207
75605139_tz.	80.46	83.908	87.356	90.805	90.805	91.954	91.954	100	100	86.364	86.207
75613109_sp.	71.591	75	78.409	81.818	81.818	85.227	85.227	86.364	86.364	100	97.727
75613110_sp.	71.264	74.713	78.161	81.609	81.609	83.908	83.908	86.207	86.207	97.727	100

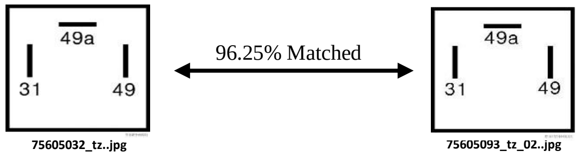
Figure 5: Hamming Distance Between Images

According to the Hamming distance we can see the image similarity. Fully similar are 100% matched (Deep Green color marked in fig:5). In our project we consider above 96% similarity (Light green in Figure 5) and in between 95% to 96% (Marked Yellow in the Figure 5). We grouped the image which are matched above 96%. In the output folder grouped the image by the similarity.

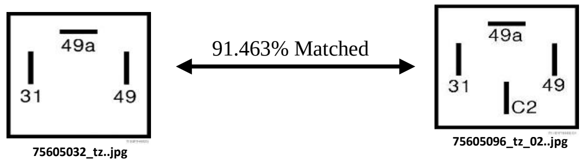
Hereafter, we use two images and compare them with other image to show the similarity. The images sources hyperlink under the name. the links are also provided in the Reference part.

Hamming distance between Image: **75605032_tz.** and others:

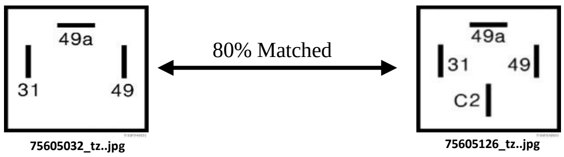
- Image: 75605032_tz. and Image: 75605093_tz_02. are 96.25% similar.



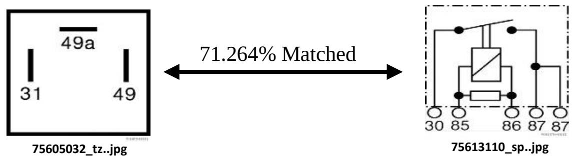
- Image: 75605032_tz. and Image: 75605096_tz_02. are 91.463% similar.



- Image: 75605032_tz. and Image: 75605126_tz. are 80% similar.

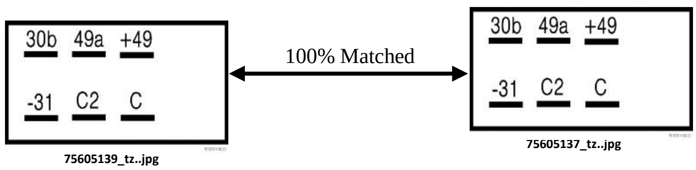


- Image: 75605032_tz. and Image: 75613110_sp. are 71.264% similar.

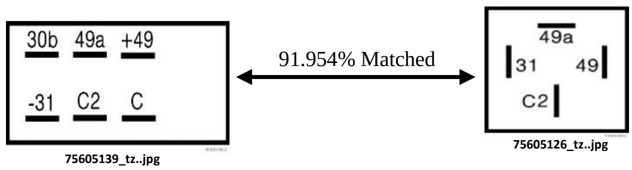


Hamming distance between Image: **75605139_tz.** and others:

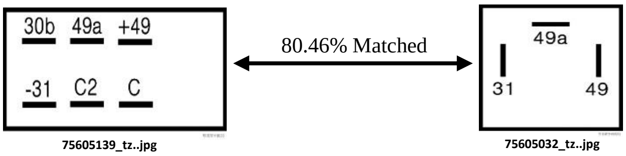
- Image: 75605139_tz. and Image: 75605137_tz. are 100% similar.



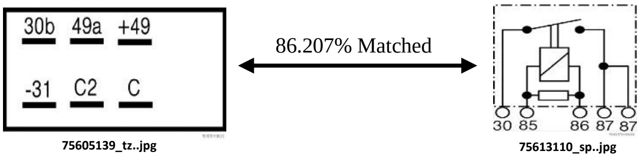
- Image: 75605139_tz. and Image: 75605126_tz. are 91.954% similar.



- ➢ Image: 75605139_tz. and Image: 75605032_tz. are 80.46% similar.



- ➢ Image: 75605139_tz. and Image: 75613110_sp. are 86.207% similar.



Limitation:

- If the dissimilarity less 96%, images put in same group. For this there may more than two image in one group.
- Having same active bit sequence consider as same image.

d. Conclusion

We have used the process and properties image classification based on Spatial pooler. The Spatial pooler has flexible coding schema that can use in real life machine learning application. We did not use any encoder. We just binarized image and give it to the spatial pooler. If we take look in our image and the output table, we can see exactly matched images are showing 100%. Further, some images are quite similar but there may some little dissimilarity.

Now in this era, there are many smart machines use image recognitions for identification process. All of this may not 100% accurate. Though our project has some limitation, but we can apply it various real-life work. We can improve result accuracy by using more images.

e. References

1. https://www.frontiersin.org/articles/10.3389/fnbot.2016.00081/full
2. https://craftofcoding.wordpress.com/2017/02/13/image-binarization-1-introduction/
3. https://www.shutterstock.com/image-vector/banana-detailed-hand-drawn-vector-illustrations-1093535486
4. Images URL:
   - Image 75605032_tz: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605032_tz..jpg
   - Image 75605093_tz_02: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605093_tz_02..jpg
   - Image 75605096_tz_02: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605096_tz_02..jpg
   - Image 75605126_tz: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605126_tz..jpg
   - Image 75613110_sp: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75613110_sp..jpg
   - Image 75605139_tz: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605139_tz..jpg
   - Image 75605137_tz: https://github.com/UniversityOfAppliedSciencesFrankfurt/se-cloud-2019-2020/blob/trozanHorse/Schema_Image_Classification/Images/75605137_tz..jpg