CS520: AI Project 4-Image Processing

Leena Dighole

---

1. Introduction

Aim: Image colorization using supervised learning algorithms and performance analysis.

An image of an elephant with **300*533 pixel** size is selected as an input to out supervised learning algorithm.

We will use KNN and a soft classification Neural Network as our simple agent and Improved agent learning algorithms.

The original input image of an elephant is,

*Figure 1: The Original Elephant image of 300*533 pixels*

We will divide this image into 2 equal half parts and use the left half of the image as a training data and right half of the image as a test data.

Dim of Left half of original image = 150*533 pixels

Dim of Right half of original image = 150*533 pixels

A colored RGB image is a 3-D matrix with shape of width*height*3

3 represents the Red, Green and Blue color of the pixel whose value varies between an integer between 0-255. When we convert the colored image into greyscale image then we loose good amount of information. We will use KNN as our simple agent and shallow NN as our improved agent for coloring the greyscale right half part of the image.

The k-means clustering is used to recolor the image with optimal k value. We calculated the optimal k =7 using an elbow method. Our aim is to recolor the right half of the image using its left part (greyscale and recolored image) as a training dataset. The performances of simple agent and Improved agent are calculated using RMSE, MAD errors.

2. Image Pre-processing

The image is loaded using Python Image Library (PIL) into the python file and converts the image into RGB 3-D array. Now we want to convert this colored image into greyscale image. For this we used the ImageOps class from PIL. The function of the color to greyscale conversion of this inbuilt package is,

Greyscale function = 0.299r + 0.587g + 0.114b

Now we will have left and right half greyscale image ready with us to be processed further.

Figure2: The Greyscale version of the original image

The original image contains various shades of colors. For training our agents, we need some representative colors. We will use k-means clustering for getting the k centroids for recoloring the image. Lets calculate the optimal k value for using the k-means clustering. K- means is an unsupervised clustering method in order to classify the data into k clusters. For calculating optimal k value for clustering, we will use an elbow method. We will construct a graph of Sum of Squared Errors(SSE) Vs k values. The point at which we find the drastic change in the slope of he graph, the corresponding k value can be used as the optimal k value.

BONUS:

Lets plot the graph to find optimal k value.

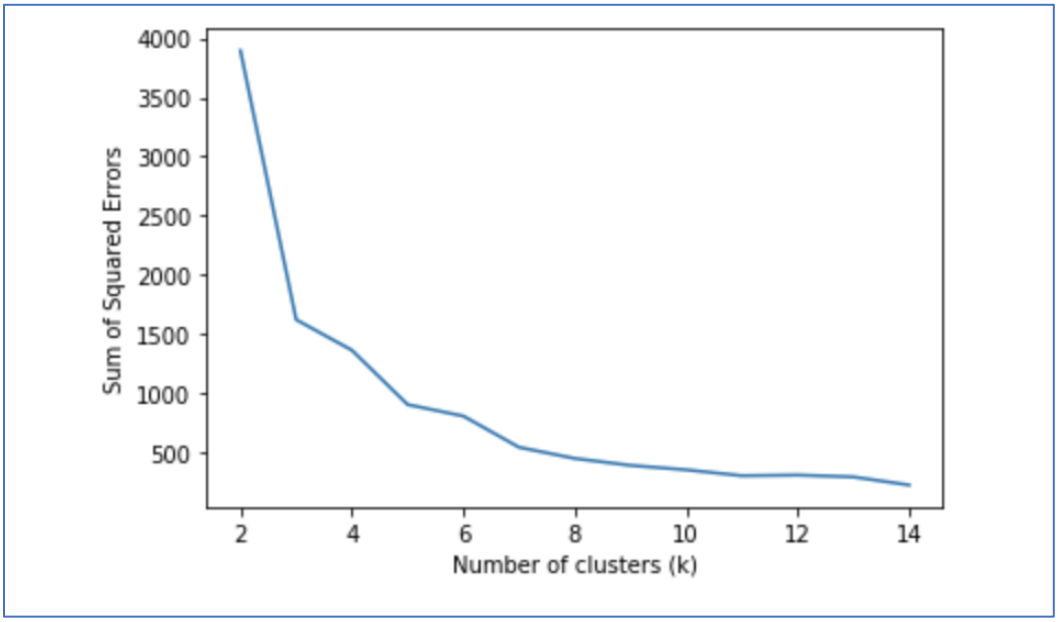

Figure 3: SSE Vs k graph for optimal k

From above graph, we can choose k =7 as the optimal k for clustering algorithm.

Using optimal k =7 value, lets perform k-means clustering algorithm in order to pick the 7 most representative colors of the image. Every pixel of the image gets cluster with 7 centroids and corresponding cluster. The Euclidean distance between each pixel and centroid is used to update the centroid location in order to classify all pixels into 7 clusters. The k-means returns the centroid locations and cluster assignments once it clusters all pixels into corresponding suitable clusters.

Once we have all pixels clustered, we will have 7 most representative colors used to color the left side greyscale image into left side colored image. We colored the left greyscale image using these 7 representative colors into colored left half of image. The colored image looks like as shown in below figure. We can notice the difference between the recolored left half of the image with the original left half of the image. Here recolored image is colored with only 7 colors.

Figure 4: The Recolored left half of the image

We will use the left half greyscale image, left half colored image and right half greyscale image as the input parameters for our basic and improved agents.

3. Basic Agent

The basic agent uses the KNN (K- Nearest Neighbors algorithm) to perform the hard classification. The input space of basic agent consists of left half greyscale image, left half colored image and right half greyscale image. Note that the left colored image has only limited k no of colors and thus we can use the classification method to recolor the right side of the greyscale image.

Basic agent uses the 3*3 patch of the greyscale image as the collection of the features for the pixels present in the middle of the patch. The Test data set is the right side greyscale image. The basic agent takes the 3*3 patch of the test data as the input and tries to find the similar 6 nearest neighbors in the testing data. The Euclidean distance between test and train patches is used in order to find the nearest neighbors. We will use the RGB color of the middle pixel of the patches from the recolored left half of the image. There will be different cases for coloring the test data patch and they are,

Case 1: Majority exists amongst nearest neighbors

If we get majority in 6 nearest neighbors then the majority RGB color gets assigned as the color of the middle pixel of the test data patch.

Case 2: Majority does not exist amongst nearest neighbors

If we don't get majority in 6 nearest neighbors then the RGB color of the nearest patch using Euclidean distance gets assigned as the color of the middle pixel of the test data patches.

Case 3: Boundary pixels

As there is absence of the 3*3 patch for boundary pixels, we will assign a black color to them with RGB values equals to 0.

We stored the information of the nearest neighbors using dictionary which our agent will use to color the test patch middle pixel.

Specifications:

- 3*3 test and training dataset patch
- Euclidean distance
- Clustering k= 7
- KNN neighbors = 6

The dictionary storage of 3*3 patches takes a long time to execute the command.

Lets see the output of the basic agent colorization.

Figure 5: The Recolored left half of the image with Basic agent colored right half

The left of the above image the recolored left half of the image and the right side is the output of the Simple agent. The simple agent doesn't recover all the color information from its greyscale version using KNN algorithm.

For simple agent,

Input space : left half greyscale image, left half colored image and right half greyscale image

Model space: K-Nearest Neighbors with 3*3 input patch

Output space: The colored right half of the image with the color of each pixel.

Performance Analysis of Basic Agent:

- The 7 centroid colors RGB value array is,

  [(212, 211, 214), (178, 155, 97), (86, 69, 50), (196, 181, 153), (154, 175, 207), (138, 119, 71),(40, 31, 24)]
- MSE : 231.75369606003753
- RMSE : 15.22345874169328
- MAD : 361.62296435272043
- Precision : [0.9633, 0.8835, 0.7442, 0.6854, 0.7082, 0.8081, 0.6598]
- Recall : [0.7779, 0.768, 0.8999, 0.7553, 0.6594, 0.9212, 0.7795]
- F1-score : [0.8607, 0.8217, 0.8147, 0.7187, 0.6829, 0.861, 0.7147]
- The Confusion matrix of 7 representative colors will be,

Confusion Matrix

11728	28	1935	278	126	897	85
2	15022	176	396	829	732	2404
208	24	9777	323	51	225	257
107	189	278	5035	740	27	290
49	279	577	998	4904	149	481
78	383	243	33	46	9381	19
3	1077	151	283	229	197	6859

Figure 6: The Confusion matrix of Basic Agent

Note that the above performance analysis is done using the left half colored image with the basic agent colored right side of the image as shown in figure 5.

The improved agent using NN will perform better than the Basic agent.

Lets surf the Improved agent logic and performances!

4. Improved Agent

The Improved agent uses Feed forward Neural Network to perform the soft classification. The input space of improved agent consists of left half greyscale image, left half colored image and right half greyscale image. Note that the left colored image has only limited k no of colors and thus we can use the classification method to recolor the right side of the greyscale image. The soft classification adds up the values of 7 output node value to 1. The output node represents the probability that the pixel belongs to which representative color. Using the elbow method, we choose our optimal k =7 for clustering. The matrix of the flattened feature patches serves as the input to the NN and training it.

We used one-hot encoding due to soft classification of multiple classes. The k-means clustering will have generalized k nodes at the output layer of the soft classification NN.

Data Preprocessing:

The input data must be normalized before passing through the NN. The normalization is done using mean normalization technique. After normalizing data, its scaled using minmax scaler between range [0,1]. The scaling of the data helps to eliminate the features with less significance. The sigmoid and Relu activation functions are used in the project. Training of the NN is done using error back propagation method.

Neural Network Specifications:

The no of input layers and hidden layers flexibility will gives us different performance results.

For Improved agent,

Input space: The NN with patch size of 5*5 pixels will be input space of each input node of NN.

Output space: The soft classification of total no of representative k colors will be the output nodes in output layer. Soft classification enables to get the probability that given patch belongs to particular representative color.

Model Space:

- No of hidden layers = 4 hidden layers and 1 input and 1 output layer
- Patches we are using 5*5 as the input to NN
- Total no of output nodes = 7 (K-means optimal K value)
- Learning rate = 0.000004
- No of iterations = 10000
- Optimization = Gradient descent method
- Batch size for mini batch gradient descent is 10000
- Activation Function = Sigmoid and Relu functions
- Soft Classification constraint for output layer
- Centroid array with 7 centroid representative color RGB values

Training Algorithm:

The initial weights initialization of NN are done randomly. The weights get updated in order to lower the cross entropy loss function. Back propagation algorithm is used to minimize the cross entropy loss.

Optimization:

The Gradient descent method , mini batch gradient descent method with appropriate batch size and Stochastic Gradient descent method optimizations can be used by the user for updating the weights using back propagation. We used simple Gradient descent method for our improved agent.

The weight updates and training algorithm takes a long time to run.

Lets see the output of the improved agent colorization.

The various performance parameters are calculated using left side colored image and right side image coloring done using an improved agent algorithm.

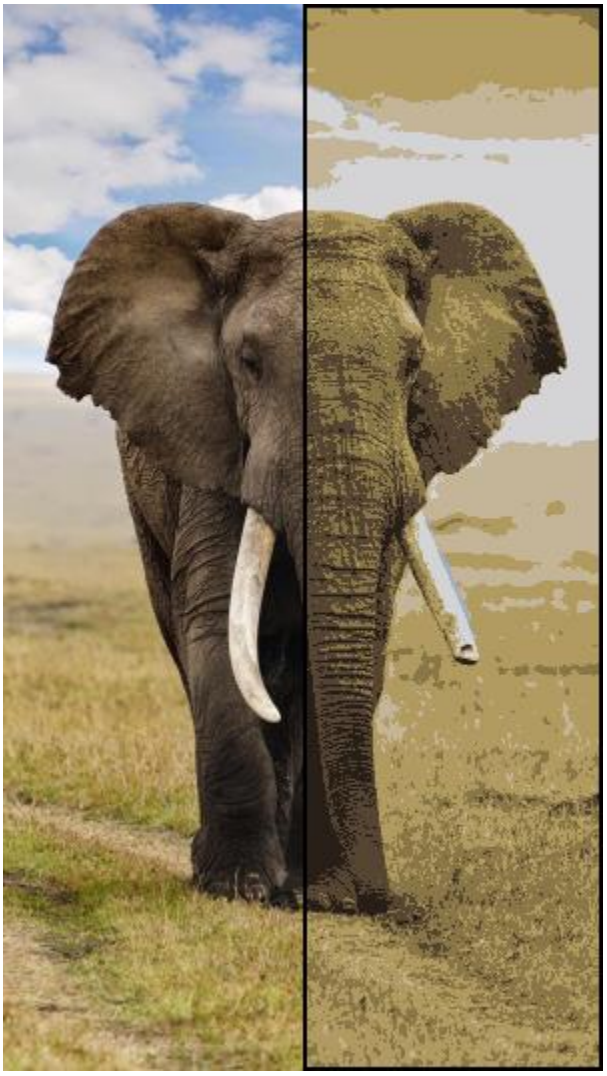

Figure 7: The Recolored left half of the image with Improved agent colored right half

The left of the above image the recolored left half of the image and the right side is the output of the Improved agent. The Improved agent using NN almost recovers all the color information from its greyscale version using feed forward NN algorithm.

Performance of Improved agent:

- The 7 centroid colors RGB value array is,

  [(212, 211, 214), (178, 155, 97), (86, 69, 50), (196, 181, 153), (154, 175, 207), (138, 119, 71),(40, 31, 24)]
- MSE : 54.17727329580988

- RMSE : 7.360521265223672
- MAD : 59.25834896810507
- Precision : [0.964, 0.9549, 0.9776, 0.6138, 0.4148, 0.9587, 0.9403]
- Recall : [0.9813, 0.8574, 0.9533, 0.6217, 0.5681, 0.9187, 1.0]
- F1-score : [0.9726, 0.9035, 0.9653, 0.6177, 0.4795, 0.9383, 0.9692]
- Support : [11712, 18591, 13332, 7124, 4913, 11937, 9625]
- The Confusion matrix of 7 representative colors will be,

```
Confusion Matrix :
 [[11493     0     0   190    29     0     0]
 [    0 15939     0   594  1597   461     0]
 [    0     0 12709     0     0    12   611]
 [  427   291     0  4429  1977     0     0]
 [    2   117     0  2003  2791     0     0]
 [    0   345   291     0   335 10966     0]
 [    0     0     0     0     0     0  9625]]
```

Figure 8: The Confusion matrix of Basic Agent

Lets analyze the performance differences of the simple and improved agent image coloring algorithm.

5. Simple Agent Vs Improved Agent

The Qualitative coloring performance of the simple agent is comparatively less accurate compared to the improved agent coloring performance.

Lets see the results and try to visualize the performances of the basic and improved agents.

The Left side colored image and the right side colored images of basic as well as improved agent can be viewed in below figure,

Figure 9: The Recolored left half of the image with simple agent and Improved agent colored right half

Clearly, we can notice the qualitative performances of Basic and improved agent. The Improved agent outperforms the simple agent in terms of image quality performance.

Lets compare the quantitative analysis of the agents.

Agent	RMSE	MSE	MAD
Simple	15.22	231.75	361.62
Improved	7.36	54.17	59.25

Table 1: Quantitative Performance analysis of Basic and Improved agent

Observations:

- The RMSE, MSE and MAD error values for improved agent are lower than simple agent.
- The qualitative analysis of the improved agent colored image is better than the simple agent.
- The precision and recall values of improved agent are more than the simple agent.
- With above qualitative and quantitative analysis, we observed that our improved agent outperforms simple agent.
- The improved agent tries to color the right half of grey scale image more efficiently than the simple agent.

Future Scope:

- The parameter tunning of the NN can give us different and improved results.
- Varying input space of NN with different patch sizes and increasing no of hidden NN layers can give us an improved result.
- We can use the CNN (Convolutional Neural Network) with sufficient filter size and try to color the image.
- For simple agent, we can vary k nearest neighbors and try to find the optimal k for which the agent gives good results.
- The LSTM RNN can be used as an improved agent and performance analysis can be done by tunning different parameters.