UNIVERSITY OF KWAZULU-NATAL

SCHOOL OF MATHS, STATISTICS AND COMPUTER SCIENCE

COMP702: IMAGE PROCESSING AND COMPUTER VISION

MINI-PROJECT

SOUTH AFRICAN BANK NOTES RECOGNITION USING IMAGE PROCESSING TECHNIQUES

Authors:

Thariq Singh

219063421@stu.ukzn.ac.za

Pooja Ramahsarai

221003194@stu.ukzn.ac.za

Callyn Blayne Barath

220010761@stu.ukzn.ac.za

Lerisha Moodley

220036955@stu.ukzn.ac.za

Table of Contents

1 Introduction

The following link can be used to access the GitHub repository for the project files:

https://github.com/thariqsingh01/Comp702_Honours_Project

This project's goal is to create an image processing and computer vision system that can efficiently identify old and new South African bank notes (R10, R20, R50, R100, and R200). The system despite the differences in sides, scale, and rotations, needs to reliably classify the relevant bank notes. The project aims to enhance the automation of accurately classifying bank notes in financial applications through the use of proper image preprocessing and enhancement, segmentation, feature extraction and classification. This report will include a comprehensive comparison of a myriad of algorithms and techniques at each stage and an accessible GitHub repository containing all the projects code and a ReadMe file, entailing the method of use. An ATM (automated teller machine) is a smart robot that has many sensors such as cameras acting as eyes for identifying and detecting bank note images, customers, and so on[6]. With effective algorithms and techniques, a sizable number of features can be retrieved from common bank note images[7]. This endeavour offers more than just hands-on experience but also furthers the creation of dependable bank note recognition systems providing a more seamless experience using ATMs, vending machines, and other financial services[5].

2 Image Preprocessing and Enhancement

The dataset that was used in this project consisted of 63 images of South African bank notes. The images included both old and new notes of values R10, R20, R50, R100, and R200. Both the front and back sides of the bank note images were included. We also used images that were rotated, the dataset included notes rotated at 90 degrees clockwise, 90 degrees counterclockwise, and 180 degrees. In addition, the size of the images varies to represent real-world situations when notes might not be properly or precisely obtained. We used images of different formats, namely JPEG, JPG, and PNG.

The goal of preprocessing is to enhance features and reduce noise in images[8]. A range of approaches were employed, and their effectiveness were assessed:

Noise Reduction
- Gaussian Blur [12]
- Median Blur [9,10]
- Bilateral Filter [12]

Contrast Enhancement
- Histogram equalization [11]
- CLAHE [13]
- Gamma Correction [14]

Detail Enhancement
- Sharpening [12]
- Unsharp Masking [15]

*CLAHE - Contrast-Limited Adaptive Histogram Equalization

Three preprocessing configurations were assessed in order to identify the best collection of techniques to use:

Combination 1: Bilateral Filter, Gamma Correction, Unsharp Mask

- **Bilateral Filter**: effectively minimizes noise while maintaining edges, which is important for maintaining features.

- **Gamma Correction**: increases brightness, enhancing the visibility of features.
- **Unsharp Masking**: enhances feature clarity by sharpening edges.

Combination 2: Gaussian Blur, Histogram Equalization, Sharpening

- **Gaussian Blur**: blurs small details by smoothing the image.
- **Histogram Equalization**: increases contrast, but noise may be amplified.
- **Sharpening**: improves edges but may add noise.

Combination 3: Median Blur, CLAHE, Unsharp Mask

- **Median Blur**: maintains edges while minimizing on noise.
- **CLAHE**: increases contrast locally, making it easier to see in dimly lit environments.
- **Unsharp Masking**: sharpens edges even further, improving the clarity of details.

Comparative Analysis

Processed images were compared in order to evaluate each combination both quantitatively and visually:

Combination 1: Bilateral Filter, Gamma Correction, Unsharp Masking

Figure 2.1 Bilateral Filter, Gamma Correction, Unsharp Masking

- **Pros**: Sharp edges, improved brightness, and efficient noise reduction.
- **Cons**: There's a slight colour distortion from using gamma correction.

Combination 2: Gaussian Blur, Histogram Equalization, Sharpening

Figure 2.2 Gaussian Blur, Histogram Equalization, Sharpening

- **Pros**: Enhanced sharpness and contrast.

- **Cons**: Higher noise level and possible loss of delicate details.

Combination 3: Median Blur, CLAHE, Unsharp Masking

Figure 2.3 Median Blur, CLAHE, Unsharp Masking

- **Pros**: Localized contrast enhancement and efficient noise reduction.
- **Cons**: Can result in areas that are too amplified, giving an artificial appearance.

Justification for Chosen Combination

Combination 1 was selected for the following reasons:

- **Balanced Noise Reduction**: The bilateral filter preserves edges, which is crucial for feature extraction, and reduces noise more effectively.
- **Brightness Enhancement**: Gamma correction guarantees enhanced detail visibility in a range of lighting scenarios.
- **Edge Clarity**: Sharpening edges with unsharp masking makes important elements appear more pronounced.

3 Image Segmentation

Image segmentation involves partitioning our image into multiple segments or regions, typically to simplify the image for further analysis or to make it more meaningful and easier to analyse. In our project, it assisted in classifying our bank notes using various features. 5 segmentation techniques were used namely Canny edge detection, Otsu's method, Adaptive Thresholding, K-Means Clustering and Colour Segmentation.

Figure 3.1. Binary Otsu

Figure 3.2. Canny Edge Segmentation

Figure 3.3. Adaptive Threshold

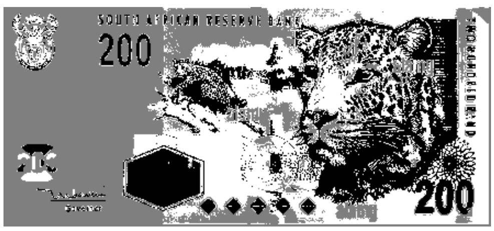

Figure 3.4. K-Means Clustering *Figure 3.5. Colour Segmentation*

- **Otsu's binary thresholding**: Offers automatic threshold determination, simplifying the binarization process and working well on high-contrast features [19]. However, it is primarily suited for grayscale images, resulting in the loss of critical colour information necessary for identifying South African banknotes making unsuitable.

- **Canny edge detection:** This method is less affected by lighting changes compared to colour-based techniques. However, it can be sensitive to noise, requiring robust pre-processing, and extracting meaningful features from edges alone can be complex [20]. Moreover, it does not utilize colour information, which is a significant drawback for banknote identification.

- **Adaptive thresholding:** Calculates thresholds for smaller image regions, effectively handles uneven lighting conditions and preserves fine details better than global thresholding methods. Despite its advantages, it is computationally more intensive and requires careful tuning of parameters like block size and constant value, which can vary between images [21].

- **K-means segmentation :** Clusters different parts of the banknote based on colour and texture, offers robust segmentation, and can handle various features, making it versatile for different denominations. However, it is computationally intensive and depends on the initial placement of cluster centres, making it unsuitable [22].

- **Colour segmentation:** particularly effective for South African banknotes due to their distinct colours for different denominations. This method is straightforward, easy to implement, and suitable for real-time applications. However, it is highly sensitive to varying lighting conditions, which can affect colour accuracy, and may also struggle with colour similarity between denominations in certain lighting scenarios [23].

To justify our choice of the Colour Segmentation Method going forward we consider the following advantages when compared to the other methods:

- **Distinct Colours:** South African banknotes utilize distinct colours for different denominations. This makes colour segmentation highly effective ([23]). Unlike grayscale methods (Otsu's thresholding), colour segmentation capitalizes on this key feature for identification.

- **Simplicity and Efficiency:** Colour segmentation is a straightforward and easy-to-implement technique compared to K-means clustering ([22]), making it suitable for real-time applications where processing speed is crucial.

- **Partial Lighting Tolerance:** While colour can be affected by lighting variations, proper calibration can mitigate this issue. Colour segmentation often performs better than methods like Canny edge detection, which are more sensitive to lighting changes and require complex pre-processing ([20]).

Addressing Limitations:

- **Advanced Calibration:** Colour segmentation can be improved by incorporating advanced calibration techniques to account for lighting variations.

- **Similar Colours in Certain Lighting:** For denominations with similar colours under specific lighting conditions, additional features like textures or patterns might be incorporated alongside colour segmentation for enhanced accuracy.

4 Feature Extraction

In the area of computer vision, features of an image are described as pieces of information. The information is normally about whether regions of the image has certain properties. To put it more concretely, features of an image are specific structures in the image such as points, edges or objects. As a result, feature extraction refers to the process of taking this raw data and converting it into numerical values that can be processed. This is done while preserving the information in the original data set.[1]

There are numerous algorithms of feature extraction that can be used. Some of the most popular methods been Convolutional Neural Networks (CNN), Gray Level Co-occurrence Matrix (GLCM) and Autoencoders.[2] For this project we used GLCM, Haralick and Linear Binary Pattern(LBP). In particular, GLCM has proven to be a popular feature extraction method. It is able to extract numerous features from images. Some of these features include Angular Second Moment(ASM), Inverse Different Moment, Entropy and Correlation.[3] For our project, we have decided to extract the ASM of the image along with the contrast, dissimilarity, homogeneity, energy, correlation. We stated the red, green, and blue features for each of the metrics state.[4] A small portion of the results from our Haralick feature extraction is shown in the table below:

Filename	Red_Contrast	Red_Dissimilarity	Red_Homogeneity	Red_ASM	Red_Energy	Red_Correlation
100back.PNG	196.1034	0.9055	0.9646	0.0.0068	0.0824	7.908
100back_rotated.PNG	438.5898	0.8473	0.9262	0.0059	0.0766	8.2687
100F.PNG	167.6054	0.9094	0.973	0.0086	0.0929	7.6832
100front_large.PNG	242.5199	0.9009	0.9452	0.0078	0.0887	7.7297
100observe.PNG	247.2783	0.8998	0.9441	0.0073	0.0856	7.8386

Table 4.1 : Haralick feature extraction Part 1(Red Features)

Filename	Green_Contrast	Green_Dissimilarity	Green_Homogeneity	Green_ASM	Green_Energy	Green_Correlation
100back.PNG	258.7796	0.9071	0.9331	0.0081	0.0903	7.6746
100back_rotated.PNG	471.0996	0.8495	0.8822	0.0071	0.0846	8.0465
100F.PNG	181.5425	0.9101	0.9742	0.0084	0.0918	7.7703
100front_large.PNG	297.3085	0.9022	0.9165	0.0086	0.0926	7.6293
100observe.PNG	319.3727	0.9012	0.9124	0.0085	0.0924	7.6735

Table 4.2 : Haralick feature extraction Part 2(Green Features)

Filename	Blue_Contrast	Blue_Dissimilarity	Blue_Homogeneity	Blue_ASM	Blue_Energy	Blue_Correlation
100back.PNG	213.4502	0.9075	0.9226	0.0092	0.096	7.4707
100back_rotated.PNG	373.1856,	0.8492	0.8665	0.008	0.0915	7.8499
100F.PNG	114.6659	0.9099	0.9739	0.009	0.0950	7.5899
100front_large.PNG	232.5791	0.9044	0.907	0.0124	0.1115	7.1863
100observe.PNG	246.2283	0.9034	0.9032	0.0122	0.1105	7.3099

Table 4.3 : Haralick feature extraction Part 3(Blue Features)

The results of all 3 algorithms were saved in 3 respective CSV files. A feed forward neural network was then created in order to compare the 3 feature extraction algorithms. The CSV files were fed into the neural network, and these were processed to determine which algorithm was the best. The metrics used to in the neural network are Texture Complexity, Overall Image Quality, Pattern Recognition and Anomaly Detection. The results are shown in the table below:

	GLCM	Haralick	LBP
Texture Complexity	70.7032	70.7032	19462947.4687
Overall Image Quality	93.0216	93.0216	3928.3329
Pattern Recognition	0.9043	0.9043	0.8233
Anomaly Detection	141.3585	141.3585	6392.1596

Table 4.4 : Feature Extraction Results of all 3 Algorithms

From these results, we can see that the results of the GLCM and Haralick algorithms are equal. The results for these 2 algorithms are very good and indicate good pattern recognition, texture complexity, image quality, and anomaly detection capabilities. On the other hand, LBP has mixed results. LBP has a very high texture complexity, overall image quality and anomaly detection result, however the pattern recognition is weaker compared to the other 2 algorithms. This is a big downside since for the detection of bank notes, this algorithm could not only miss certain features of the notes but also include added noise and irrelevant details. In conclusion, for feature extraction with regard to the detection of bank notes, we recommend the use of a GLCM or Haralick algorithm.

5 Notes Classification

Classification is a fundamental task in machine learning, involving the categorization of data into predefined classes or labels. Our objective was to create a reliable system capable of accurately classifying both old and new banknotes, considering the features from both sides of the notes. In our project we aimed to classify South African banknotes using three popular classifiers namely Random Forest, K-Nearest Neighbours (KNN), and Support Vector Machine (SVM).

Random Forest Classifier: This classifier is well known for its ensemble learning approach, which consists of constructing many decision trees during the training and combining their outputs to determine the most prevalent class [17]. In scenarios involving high-dimensional data such as banknote recognition, its resistance to overfitting made it a good choice for our project. It is capable of effectively manage large datasets, maintaining balanced weights and ensuring robust generalization to unseen variations in banknotes which is an essential requirement for accurate classification across many conditions.

K-Nearest Neighbours (KNN) Classifier: The KNN classifier demonstrates simplicity and effectiveness in handling irregular decision boundaries. It is an instance-based learning algorithm which categorizes samples based on the majority class among their nearest neighbours in the feature space. Its non-parametric nature allows it to get rid of underlying assumptions about data distribution, making it a good classifier of our project. KNN served as a basic standard for comparison against the more complex classifiers, allowing us to thoroughly evaluate how well different methods performed, especially when dealing with some decision boundaries that were not as clear.

Support Vector Machine (SVM) Classifier: The SVM classifier was a powerful classifier for handling the complexity and high dimensionality of the banknote features [16]. It worked by finding the best line to separate different types of banknotes in a space with many dimensions and proved effective. SVM's ability to handle these complex spaces, along with the clear separation it showed between different banknote types, made it a great choice for our project. By using a linear kernel and balanced weights, SVM makes it a great choice for classifying banknotes in real world scenarios.

Model Training and Evaluation

We split the dataset into training and testing sets with ratios of 80-20, 85-15, and 70-30 to evaluate the classifiers' performance on unseen data (Screenshots can be found within each split folder). Features from the Gray Level Co-occurrence Matrix (GLCM) were used as classification scores were higher than that of the Haralick and Linear Binary Pattern (LBP) features.

Comparative Analysis

SPLIT	CLASSIFIER	ACCURACY (%)	PRECISION	RECALL	F1-SCORE
70-30	**Random Forest**	68	0.74	0.69	0.70
70-30	**KNN**	47	0.42	0.58	0.39
70-30	**SVM**	74	0.67	0.69	0.67
80-20	**Random Forest**	46	0.55	0.50	0.30
80-20	**KNN**	38	0.42	0.57	0.33
80-20	**SVM**	85	0.88	0.94	0.88
85-15	**Random Forest**	80	0.82	0.79	0.76
85-15	**KNN**	50	0.47	0.53	0.38
85-15	**SVM**	80	0.82	0.79	0.76

**Macro Avg values are used for Precision, Recall and F1-Score*

Random Forest:

- Relatively stable performance across the different splits.
- Performs better with larger training sets (e.g.70-30 and 85-15 splits).
- Achieved a higher accuracy and F1-score compared to KNN but slightly lower than SVM in most cases.

KNN (K-Nearest Neighbours):

- Showed inconsistent performance across the different splits.
- Performed poorly with smaller training sets (e.g., 80-20 split).
- Had lower accuracy, precision, recall, and F1-score compared to Random Forest and SVM in most cases.

SVM (Support Vector Machine):

- Performed well in general across different splits.
- Achieved the highest accuracy, precision, recall, and F1-score in most cases, especially with the larger training sets.
- Showed a slight decline in performance with the smaller training sets but still outperforms Random Forest and KNN.

Overall, SVM appears to be the most robust classifier in this analysis, followed by Random Forest. KNN performs the poorest among the three classifiers, especially with smaller training sets. Additionally, we consider the trade-offs between accuracy, precision, recall, and F1-score based on specific application requirements which is essential when selecting the appropriate classifier.

6 Conclusion

In this project, we successfully developed a system for recognizing South African banknotes using advanced image processing techniques. Our approach encompassed several critical steps, including image preprocessing, segmentation, feature extraction, and classification. We evaluated multiple algorithms and techniques to enhance the accuracy and reliability of banknote recognition.

During image preprocessing, techniques such as bilateral filtering, gamma correction, and unsharp masking proved effective in minimizing noise and enhancing features. For image segmentation, colour segmentation proved the best as it has partial light tolerance and can handle South African bank notes' distinct colours better. Among the three feature extraction methods tested—GLCM, Haralick, and LBP—GLCM and Haralick showed superior performance, particularly in terms of texture complexity, image quality, pattern recognition, and anomaly detection.

For classification, we employed Random Forest, K-Nearest Neighbours (KNN), and Support Vector Machine (SVM). SVM emerged as the most robust classifier, consistently delivering high accuracy, precision, recall, and F1-scores across different training set splits. Random Forest also performed well, while KNN showed inconsistent results, especially with smaller training sets.

In conclusion, our findings demonstrate that a combination of GLCM or Haralick feature extraction with SVM classification offers a reliable solution for banknote recognition. This system can significantly enhance automation in financial applications, such as ATMs and vending machines, by accurately identifying different denominations of banknotes under various conditions.

7 References

[1] Kumar, G. and P.K. Bhatia, A Detailed Review of Feature Extraction in Image Processing Systems, in 2014 Fourth International Conference on Advanced Computing & Communication Technologies. 2014. p. 5-12.

[2] Manjunath Jogin, M., Madhulika M S, Divya G D, Meghana R K, Apoorva S, Feature_Extraction_using_Convolution_Neural_Networks_CNN_and_Deep_Learning. 2018. p. 5.

[3] P.Mohanaiah, P.S., L.GuruKumar, Image Texture Feature Extraction Using GLCM. 2013.

[4] A. O. Salau and S. Jain, "Feature Extraction: A Survey of the Types, Techniques, Applications," 2019 International Conference on Signal Processing and Communication (ICSC), NOIDA, India, 2019, pp. 158-164, doi: 10.1109/ICSC45622.2019.8938371. keywords: {Image color analysis;Feature extraction;Hidden Markov models;Iron;Data mining;Image retrieval;Feature extraction;image processing;application;mining;data},

[5] Lee, J.W., et al., *A Survey on Banknote Recognition Methods by Various Sensors.* Sensors (Basel), 2017. **17**(2).

[6] Sako, H., et al., *Self-Defense-Technologies for Automated Teller Machines*, in *International Machine Vision and Image Processing Conference (IMVIP 2007)*. 2007. p. 177-184.

[7] Lowe, D.G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60, 91–110… https://doi.org/10.1023/b:visi.0000029664.99615.94

[8] N.Panah, H.M., *Banknotes detected using Image Processing Techniques.* International Journal of Computer Science and Mobile Computing, 2017.

[9] Manikandan, K.; Sumithra, T. Currency recognition in mobile application for visually challenged. Discovery 2015, 30, 245–248.

[10] Hlaing, K.N.N. First order statistics and GLCM based feature extraction for recognition of Myanmar paper currency. In Proceedings of the 31st IIER International Conference, Bangkok, Thailand, 2 August 2015; pp. 1–6.

[11] Behjat, M.; Moallem, P. Fast and low-cost mechatronic recognition system for Persian banknotes. Int. J. Adv. Robot. Syst. 2014, 11, 44. [CrossRef]

[12] Daniel L´elis Baggio. *Mastering OpenCV with practical computer vision projects*. Packt Publishing Ltd, 2012.

[13] Musa, P., F.A. Rafi, and M. Lamsani, *A Review: Contrast-Limited Adaptive Histogram Equalization (CLAHE) methods to help the application of face recognition*, in *2018 Third International Conference on Informatics and Computing (ICIC)*. 2018. p. 1-6.

[14] Rahman, S., et al., *An adaptive gamma correction for image enhancement.* EURASIP Journal on Image and Video Processing, 2016. **2016**(1).

[15] J.N, A., *A Review on the Image Sharpening Algorithms Using Unsharp Masking.* IJESC, 2016. **6**: p. 8729-8733.

[16] D. M. Abdullah and A. M. Abdulazeez, "Machine learning applications based on SVM classification a review," Qubahan Academic Journal, vol. 1, no. 2, pp. 81-90, 2021.

[17] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," Journal of Applied Science and Technology Trends, vol. 2, no. 01, pp. 20-28, 2021.

[18] L. E. Peterson, "K-nearest neighbor," Scholarpedia, vol. 4, no. 2, p. 1883, 2009.

[19] J. Yousefi, "Image binarization using Otsu thresholding algorithm," Ontario, Canada: University of Guelph, vol. 10, 2011.

[20] L. Xuan and Z. Hong, "An improved canny edge detection algorithm," in 2017 8th IEEE international conference on software engineering and service science (ICSESS), 2017: IEEE, pp. 275-278.

[21] P. Roy, S. Dutta, N. Dey, G. Dey, S. Chakraborty, and R. Ray, "Adaptive thresholding: A comparative study," in 2014 International conference on control, Instrumentation, communication and Computational Technologies (ICCICCT), 2014: IEEE, pp. 1182-1186.

[22] P. Panwar, G. Gopal, and R. Kumar, "Image Segmentation using K-means clustering and Thresholding," Image, vol. 3, no. 05, pp. 1787-1793, 2016.

[23] D. E. Ilea and P. F. Whelan, "Image segmentation based on the integration of colour–texture descriptors—A review," Pattern Recognition, vol. 44, no. 10-11, pp. 2479-2501, 2011.

8 Appendix

Results obtained using GLCM and various split ratios:

70-30 Split:

```
Training Random Forest Classifier...
Random Forest Accuracy: 0.6842105263157895
Random Forest Classification Report:
              precision    recall  f1-score   support

          10       0.67      0.67      0.67         6
          20       0.50      0.75      0.60         4
         100       0.80      0.67      0.73         6
         200       1.00      0.67      0.80         3

    accuracy                           0.68        19
   macro avg       0.74      0.69      0.70        19
weighted avg       0.73      0.68      0.69        19

Random Forest Confusion Matrix:
 [[4 2 0 0]
 [0 3 1 0]
 [2 0 4 0]
 [0 1 0 2]]
```

```
Training K-Nearest Neighbors Classifier...
KNN Accuracy: 0.47368421052631576
KNN Classification Report:
              precision    recall  f1-score   support

          10       0.44      0.67      0.53         6
          20       0.50      0.25      0.33         4
          50       0.00      1.00      0.00         0
         100       0.50      0.33      0.40         6
         200       0.67      0.67      0.67         3

    accuracy                           0.47        19
   macro avg       0.42      0.58      0.39        19
weighted avg       0.51      0.47      0.47        19

KNN Confusion Matrix:
 [[4 0 0 2 0]
 [1 1 1 0 1]
 [0 0 0 0 0]
 [4 0 0 2 0]
 [0 1 0 0 2]]
```

```
Training Support Vector Machine Classifier...
SVM Accuracy: 0.7368421052631579
SVM Classification Report:
              precision    recall  f1-score   support

          10       0.83      0.83      0.83         6
          20       0.50      0.25      0.33         4
         100       0.86      1.00      0.92         6
         200       0.50      0.67      0.57         3

    accuracy                           0.74        19
   macro avg       0.67      0.69      0.67        19
weighted avg       0.72      0.74      0.72        19

SVM Confusion Matrix:
 [[5 0 1 0]
 [1 1 0 2]
 [0 0 6 0]
 [0 1 0 2]]
```

Random Forest *KNN* *SVM*

80-20 Split:

```
Training Random Forest Classifier...
Random Forest Accuracy: 0.46153846153846156
Random Forest Classification Report:
              precision    recall  f1-score   support

          10       0.00      1.00      0.00         0
          20       0.25      0.50      0.33         2
          50       1.00      0.50      0.67         8
         100       1.00      0.00      0.00         1
         200       0.50      0.50      0.50         2

    accuracy                           0.46        13
   macro avg       0.55      0.50      0.30        13
weighted avg       0.81      0.46      0.54        13

Random Forest Confusion Matrix:
 [[0 0 0 0 0]
 [1 1 0 0 0]
 [1 2 4 0 1]
 [1 0 0 0 0]
 [0 1 0 0 1]]
```

```
KNN Classification Report:
              precision    recall  f1-score   support

          10       0.00      1.00      0.00         0
          20       0.00      0.00      0.00         2
          50       0.75      0.38      0.50         8
         100       0.33      1.00      0.50         1
         200       1.00      0.50      0.67         2

    accuracy                           0.38        13
   macro avg       0.42      0.57      0.33        13
weighted avg       0.64      0.38      0.45        13

KNN Confusion Matrix:
 [[0 0 0 0 0]
 [1 0 1 0 0]
 [1 3 3 1 0]
 [0 0 0 1 0]
 [0 0 0 1 1]]
```

```
Training Support Vector Machine Classifier...
SVM Accuracy: 0.8461538461538461
SVM Classification Report:
              precision    recall  f1-score   support

          20       0.50      1.00      0.67         2
          50       1.00      0.75      0.86         8
         100       1.00      1.00      1.00         1
         200       1.00      1.00      1.00         2

    accuracy                           0.85        13
   macro avg       0.88      0.94      0.88        13
weighted avg       0.92      0.85      0.86        13

SVM Confusion Matrix:
 [[2 0 0 0]
 [2 6 0 0]
 [0 0 1 0]
 [0 0 0 2]]
```

Random Forest *KNN* *SVM*

85-15 Split:

```
Training Random Forest Classifier...
Random Forest Accuracy: 0.8
Random Forest Classification Report:
              precision    recall  f1-score   support

          10       1.00      0.67      0.80         3
          20       0.50      1.00      0.67         1
         100       0.80      1.00      0.89         4
         200       1.00      0.50      0.67         2

    accuracy                           0.80        10
   macro avg       0.82      0.79      0.76        10
weighted avg       0.87      0.80      0.80        10

Random Forest Confusion Matrix:
 [[2 0 1 0]
 [0 1 0 0]
 [0 0 4 0]
 [0 1 0 1]]
```

```
Training K-Nearest Neighbors Classifier...
KNN Accuracy: 0.5
KNN Classification Report:
              precision    recall  f1-score   support

          10       0.67      0.67      0.67         3
          20       0.00      0.00      0.00         1
          50       0.00      1.00      0.00         0
         100       0.67      0.50      0.57         4
         200       1.00      0.50      0.67         2

    accuracy                           0.50        10
   macro avg       0.47      0.53      0.38        10
weighted avg       0.67      0.50      0.56        10

KNN Confusion Matrix:
 [[2 0 0 1 0]
 [0 0 1 0 0]
 [0 0 0 0 0]
 [1 1 0 2 0]
 [0 1 0 0 1]]
```

```
Training Support Vector Machine Classifier...
SVM Accuracy: 0.8
SVM Classification Report:
              precision    recall  f1-score   support

          10       1.00      0.67      0.80         3
          20       0.50      1.00      0.67         1
         100       0.80      1.00      0.89         4
         200       1.00      0.50      0.67         2

    accuracy                           0.80        10
   macro avg       0.82      0.79      0.76        10
weighted avg       0.87      0.80      0.80        10

SVM Confusion Matrix:
 [[2 0 1 0]
 [0 1 0 0]
 [0 0 4 0]
 [0 1 0 1]]
```

Random Forest *KNN* *SVM*

Results from Feature Extraction:

```
LBP Feature Extraction
Image: 100back.PNG
Red_lbp_0: 0.0000
Red_lbp_1: 0.0024
Red_lbp_2: 0.0000
Red_lbp_3: 0.0288
Red_lbp_4: 0.0010
Red_lbp_5: 0.0913
Red_lbp_6: 0.0026
Red_lbp_7: 0.0270
Red_lbp_8: 0.8469
Green_lbp_0: 0.0000
Green_lbp_1: 0.0024
Green_lbp_2: 0.0000
Green_lbp_3: 0.0286
Green_lbp_4: 0.0010
Green_lbp_5: 0.0910
Green_lbp_6: 0.0024
Green_lbp_7: 0.0269
Green_lbp_8: 0.8476
Blue_lbp_0: 0.0000
Blue_lbp_1: 0.0024
Blue_lbp_2: 0.0000
Blue_lbp_3: 0.0287
Blue_lbp_4: 0.0009
Blue_lbp_5: 0.0927
Blue_lbp_6: 0.0026
Blue_lbp_7: 0.0269
Blue_lbp_8: 0.8458
```

Features using LBP Algorithm

```
Haralick Feature Extraction
Image: 100back.PNG
Red_contrast: 196.1035
Red_dissimilarity: 0.9056
Red_homogeneity: 0.9646
Red_asm: 0.0068
Red_energy: 0.0824
Red_correlation: 7.9079
Green_contrast: 258.7796
Green_dissimilarity: 0.9071
Green_homogeneity: 0.9331
Green_asm: 0.0081
Green_energy: 0.0903
Green_correlation: 7.6746
Blue_contrast: 213.4502
Blue_dissimilarity: 0.9075
Blue_homogeneity: 0.9226
Blue_asm: 0.0092
Blue_energy: 0.0960
Blue_correlation: 7.4707
```

Features using Haralick Algorithm

```
GLCM Feature Extraction
Image: 100back.PNG
Red_contrast: 196.1035
Red_dissimilarity: 3.1147
Red_homogeneity: 0.9056
Red_asm: 0.0068
Red_energy: 0.0824
Red_correlation: 0.9646
Green_contrast: 258.7796
Green_dissimilarity: 3.156
Green_homogeneity: 0.9071
Green_asm: 0.0081
Green_energy: 0.0902
Green_correlation: 0.9331
Blue_contrast: 213.4502
Blue_dissimilarity: 2.5656
Blue_homogeneity: 0.9075
Blue_asm: 0.0092
Blue_energy: 0.0960
Blue_correlation: 0.9226
```

Features using GLCM Algorithm

```
GLCM Texture Complexity: 70.70320392865253
GLCM Overall Image Quality: 93.0216293845286
GLCM Pattern Recognition Metric: 0.90429215989162
GLCM Anomaly Detection Metric: 141.35845427522293
Haralick Texture Complexity: 70.70320392865253
Haralick Overall Image Quality: 93.0216293845286
Haralick Pattern Recognition Metric: 0.90429215989162
Haralick Anomaly Detection Metric: 141.35845427522293
LBP Texture Complexity: 19462947.46868138
LBP Overall Image Quality: 3928.332958311008
LBP Pattern Recognition Metric: 0.8232725735918687
LBP Anomaly Detection Metric: 6392.159584981603
Mean LBP Homogeneity: 0.9530953999331355
Mean LBP Correlation: 0.7209347488061948
```

Results of the Comparison of the three Feature Extraction Algorithms