

ENGR 418 PROJECT REPORT

**School of Engineering
Faculty of Applied Sciences
University of British Columbia**

Project Title: Sorting using Raw Images

Group No.: 20

Members: Jesse Alele (82807728), Zach Kelly (41637836)

Date: December 18th, 2022

Introduction

This project takes a machine-learning approach to sort circular, rectangular and square-shaped lego blocks using raw image data. In the first stage of this project, the provided dataset included idealized raw image data of centred and oriented lego blocks. In this stage, the raw image dataset includes angled and disoriented lego blocks. To address non-ideal image data, feature engineering is integrated. The objective is to extract features that provide consistent and acceptable performance scores. In this report, the 3-class K-nearest neighbour classifier is trained using engineered features to predict the shape of the lego blocks. A sample of the dataset is shown below.



Figure 1: the three classes of lego blocks in various positions and orientations from the stage 2 dataset.

Theory

This stage of the project also dealt with classifying images according to the shapes of lego blocks within them. These shapes, however, were oriented and positioned in various ways. We found that the classification algorithm from stage one of the project yielded a poor performance when tested on 'Lego_data_set_2'. The logistic regression model used in stage one associates an arrangement of pixels from a single region within images to their corresponding label. When the orientation and position of these pixels change, the model struggles to maintain accuracy without human intervention. This is where feature engineering

becomes beneficial. Featurizing engineering simply means creating new features from a data set. Since we needed features that better distinguish the shapes at multiple locations within the images, features based on the dimensions of the shapes were considered. Additionally, to reduce the amount of data processing in gathering these features, we extracted shape edges from each image using a canny edge detector.

After obtaining new features, we considered using a different classifier other than logistic regression. We found that the K-nearest neighbour classifier(KNN) yielded better results when tested with the obtained features compared to logistic regression. KNN assumes that related data points are close to each other. In this case, features from the testing data set are cross-referenced with features from the training data set. Labels are then assigned depending on how close in distance both these features are to each other. KNN was considered because it is an effective method of supervised learning especially when the dimension of features is low.

Algorithm

Our algorithm consisted of four successive stages namely; feature extraction, label extraction, classifier training and definition then classifier testing.

Feature Extraction

Four features were extracted from each image to aid in distinguishing the three shapes we dealt with. The features extracted include; the sum of significant edges, shape area, minimum shape width and maximum change in shape width. To complete this stage, a 'feature_extractor' function was defined.

The sum of significant edges was obtained after rotating each image to multiple angles specified in the algorithm. For each angle, the shape edges were obtained using a canny filter. An array of column sums was then calculated from the image pixel array and the sums less than a specified value were changed to zero. This acted as a high-pass filter. Each array of column sums was then appended to a list of significant edges. When all rotation angles have been considered, all arrays of column sums within the significant edges list are added. All elements within the resulting list were then summed to obtain a single column sum from all rotation angles.

Similarly to the first feature, shape edges were obtained from each image to extract the remaining three features. Each shape area was estimated by finding how wide each row was from the image pixel array. Each width was then added to the next for a length equal to the image's height. The resulting sum gave the shape area. The next feature we obtained was the minimum shape width. This part was implemented by selecting the smallest width from each row of the already determined widths in the previous feature. The last feature we obtained was the maximum change in shape width. These values were obtained by taking the

maximum absolute difference in width from one row to the next in the image array for a length also equal to the image height.

Label Extraction

This stage of the algorithm was similar to how we assigned labels in the first part of the project. A 'label_extractor' function that returns an array of labels was defined. Filenames were retrieved from a defined folder. The filenames were then iterated through while assigning 0, 1 or 2 depending on whether the three letters 'circ', 'rec' or 'squ' existed in the filename. To further explain this, if 'circ' appeared in the filename, 0 was assigned as a label. If 'rec' appeared in the filename, 1 was assigned as a label and if 'squ' appeared in the filename, 2 was assigned as a label.

Classifier and Training

A KNN model was defined and trained using data sets from the training folder. Both 'x_train' and 'y_train' were obtained by calling the feature extraction and label extraction functions respectively.

Testing

Similarly, a testing function was defined to implement this stage of the algorithm. In the function's definition, the feature extractor and label extractor functions were called to obtain the respective data sets. The feature data set is then passed to the model to obtain predicted results. The predicted results are used to obtain the classifier's accuracy and confusion matrix. The function then returns the classifier's performance for a given folder.

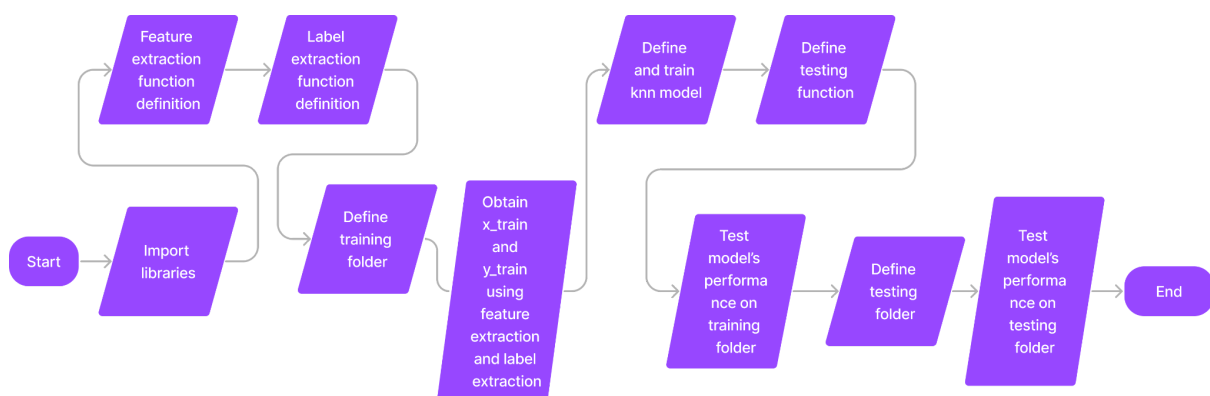


Figure 2: flowchart summarising algorithm

Results and Discussion

The K-nearest neighbours classifier is trained on the training set using feature data from the feature extractor function and label data from the label extractor function. The classification outcomes of the classifier on the training and testing sets are described below in the form of the confusion matrix.

col_0	0	1	2
row_0			
0	27	0	0
1	0	27	0
2	0	0	27

Figure 3: confusion matrix of the stage 2 classifier on the stage 2 training set

col_0	0	1	2
row_0			
0	26	0	1
1	0	27	0
2	1	0	26

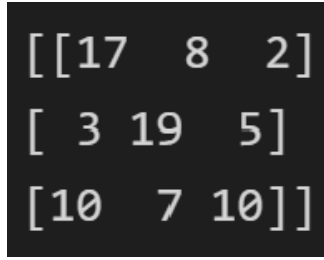
Figure 4: confusion matrix of the stage 2 classifier on the stage 2 testing set

The confusion matrix for the stage 2 training set has an overall accuracy of **100%** (all shapes were correctly classified). High accuracy on the training set is expected, hence the model learns the features using the training set.

The confusion matrix for the stage 2 testing set has an overall accuracy of **97.5%**. The classifier misclassified one circle and one square. Yielding individual accuracies of 96.3% for both circles and squares, and 100% accuracy for rectangles. Mis-classification of squares and circles is more expected, hence the extracted features depend heavily on classifying based on area measures (i.e. area and the sum of significant edges) and these shapes have similar areas. The misclassification may also be a result of the canny edge detection algorithm failing to classify the correct pixels as edges. The datasets contain shadowing which increases the grayscale value of the image background and yellow and white coloured lego blocks which have smaller grayscale values closer to the value of the white background. This in a sense blends the foreground (lego block) and background of the image making it more difficult to differentiate.

[[27	0	0]
[0	27	0]
[0	0	27]]

Figure 5: confusion matrix of the stage 1 classifier on the stage 2 training set



[17	8	2]
[3	19	5]
[10	7	10]]

Figure 6: confusion matrix of the stage 1 classifier on the stage 2 testing set

The confusion matrix for stage 1 has an overall accuracy of **100%** on the training set and **56.8%** on the testing set. The classifier misclassified ten circles, eight rectangles and seventeen squares. Yielding individual accuracies of 70% for circles, 70.4% for rectangles and 37% for squares. Worse accuracy is expected, hence the stage 1 classifier only recognizes shapes based on the grayscale value of the image pixels. The grayscale values will not be consistent for a particular across images of lego blocks, due to changes in position, orientation, shading in the image, etc. Thus the importance of feature engineering to extract more consistent features and enhance model robustness.

Conclusion

In this project, 3-class classification models were trained to classify images of circular, rectangular and square-shaped lego blocks. To successfully classify the stage 2 dataset containing angled and off-centred images a feature extraction algorithm was devised. The algorithm uses edge detection (canny filtering) to compute the feature parameters: sum of significant edges, area, minimum width and maximum change in width. Despite the increase in computational cost, the engineered features along with the K-nearest neighbours classification model achieve an overall accuracy of **100%** on the training set and **97.5%** on the testing set. While the logistics regression classification model used in stage 1 achieves an overall accuracy of **100%** on the training set and **56.8%** on the testing set. Suggesting that these engineered features are significantly more effective in classifying the lego blocks than using the grayscale pixel values as features.

To further improve the performance of the K-nearest neighbour classification model more engineered features may be added or existing ones modified. The sum of significant edges feature parameter may be modified to find the optimal rotation angle for each image. The optimal rotation angle may be defined as the angle that produces the largest sum of significant edges. The remaining feature parameters could then be computed at the optimal rotation angle for each image. Therefore, resolving the problem associated with the poor orientation of lego blocks.