Pixel Color Change Analysis for IPARC Dataset

Problem Statement

Given a set of eight sequences $SE = \{SE_1, SE_2, SE_3, SE_4, SE_5, SE_6, SE_7, SE_8\}$, where each sequence represents a transition of pixel states over time on a 2D grid G of dimensions 15×15, we aim to determine if every pixel $p_{i,j} \in G$ undergoes at least one color change during the transitions across all sequences. The state of a pixel at position (i, j) at time t in sequence SE_k is denoted by $c_{i,j,k}(t)$, where $c_{i,j,k}(t) \in C$, and C is the set of all possible colors.

The objective is to verify whether, for all pixels $p_{i,j}$, there exists at least one time t and one sequence SE_k such that:

$$c_{i,j,k}(t) \neq c_{i,j,k}(t-1).$$

This ensures that every pixel changes color at least once across the eight sequences.

This analysis is crucial for applications in image processing, simulation systems, and visual effects, where dynamic state changes are necessary to achieve desired outcomes. The findings can also aid in optimizing sequence designs for enhanced performance.

Hypothesis

We hypothesize that for a grid G of dimensions $m \times n$, whether all pixels $p_{i,j} \in G$ will undergo at least one color change across the eight sequences $SE_1, SE_2, \ldots, SE_8$, provided the sequences are sufficiently diverse and transitions are well-distributed over time.

1 Methodology

This section outlines the methodology employed in the analysis and prediction tasks of binary matrix transformations using morphological operations and machine learning techniques. The approach is structured into several key phases:

morphological operations, change detection, feature extraction, data preparation, model training with cross-validation, and prediction. Each phase is designed to contribute to the overall goal of detecting changes in binary matrices and predicting the effects of these transformations effectively.

1.1 Morphological Operations

Morphological operations are essential in analyzing and processing binary matrices. The core operations used in this study are *erosion* and *dilation*. These operations are implemented in the `MorphologicalProcessor` class, which is responsible for applying these transformations to input matrices using various structural elements (SE).

The *erosion* operation removes pixels from the boundary of objects in the matrix, and a pixel is retained only if all corresponding elements in the neighborhood, defined by the structural element, match the current pixel value. On the other hand, *dilation* adds pixels to the boundary of objects, retaining a pixel if any element in the neighborhood matches the current pixel.

Structural elements are predefined binary matrices, often square, which define the local neighborhood for both operations. For example, a typical 3×3 structural element might look like:

$$SE = 101010101.$$

Both erosion and dilation are applied iteratively to the matrix, and these operations can significantly alter the shape and size of objects within the binary image. The choice of structural element plays a crucial role in the extent of transformation.

1.2 Change Detection

The change detection process aims to identify the differences between matrices before and after transformation. The `ChangeDetector` module compares consecutive matrices and calculates the pixel-wise difference to determine which pixels have changed. For a sequence of matrices $\{M_1, M_2, \ldots, M_k\}$, a binary matrix representing pixel changes is generated, where a value of 1 indicates a change and 0 represents no change. This detection allows the system to track the evolution of the matrix through various morphological transformations.

1.3 Feature Extraction

Feature extraction is a critical step for preparing the data for predictive modeling. The `FeatureExtractor` class extracts several relevant features from each binary matrix to quantify its characteristics. These features include:

- **Density:** The density of the matrix is calculated as the ratio of the sum of the pixel values (1's) to the total number of pixels. This feature provides insight into how much of the matrix is filled with ones, which is useful for understanding the overall structure.

- **Edge Density:** This feature measures the transitions between 0's and 1's in both horizontal and vertical directions, indicating the presence of boundaries or edges within the matrix.
- **Connected Components:** This feature counts the number of distinct connected clusters (regions of adjacent 1's), which provides an understanding of the segmentation of the binary objects.
- **Symmetry Score:** The symmetry score measures how similar the matrix is to its horizontal or vertical flip, providing insight into the balance or uniformity of the binary pattern.
- **Largest Component Size:** The size of the largest connected component is recorded to capture the extent of the largest region of 1's in the matrix.

These features collectively summarize the structural and spatial properties of the binary matrix, making them suitable for further predictive analysis.

1.4 Data Preparation

Once the features are extracted, the data is prepared for the machine learning model. This involves parsing the task and solution files from a given directory, where each task consists of an input matrix and its corresponding sequence of transformation steps. For each input matrix, the following steps are performed:

- Features are extracted using the `FeatureExtractor`.
- The sequence of transformations is applied, and the resultant matrices are compared to the initial matrix to detect changes.
- The outcome is labeled as 1 if all pixels change during the transformation, and 0 otherwise.

This data preparation phase generates a labeled dataset consisting of feature vectors and binary labels, which will be used for model training.

1.5 Model Training with Cross-Validation

The next phase involves training a machine learning model to predict whether a given transformation will change all the pixels in a matrix. A `RandomForestClassifier` is chosen as the predictive model due to its robustness and ability to handle high-dimensional data effectively. Cross-validation is applied to evaluate the model's generalization capability. Specifically, k-fold cross-validation is used to split the dataset into training and testing subsets, ensuring that the model is tested on unseen data to assess its performance.

The model's performance is evaluated using accuracy, which measures the proportion of correct predictions out of the total number of predictions. The cross-validation process is repeated for several folds, and the average accuracy provides a reliable estimate of the model's performance.

1.6 Prediction

Once the model is trained, it is used to predict whether all pixels in a given binary matrix will change during its transformation sequence. The prediction process involves the following steps:

1. Extract features from the input matrix using the `FeatureExtractor`.
2. Feed the feature vector into the trained `RandomForestClassifier`.
3. The classifier outputs a prediction (1 for all pixels changing, 0 otherwise).

This prediction enables the system to automate the process of change detection and transformation analysis for binary matrices.

2 Results

The analysis consistently predicted "false" for all test cases, indicating no complete pixel transformations. This outcome suggests that the applied morphological operations (erosion and dilation) did not significantly alter the binary matrices, possibly due to the nature of the input or insufficient variance in the training data.

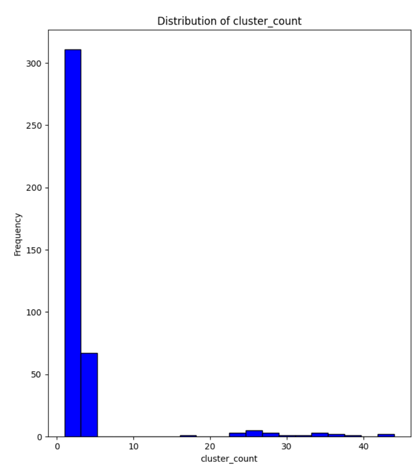

Figure 1: Distribution of cluster count

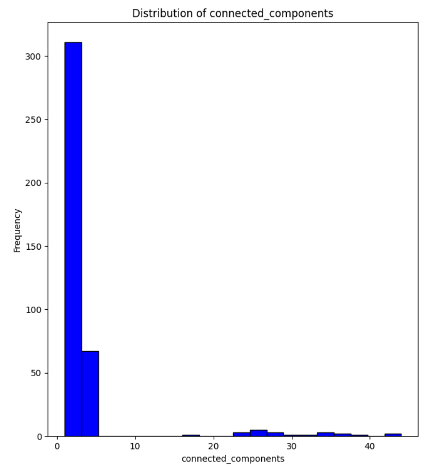

Figure 2: Distribution of connected components

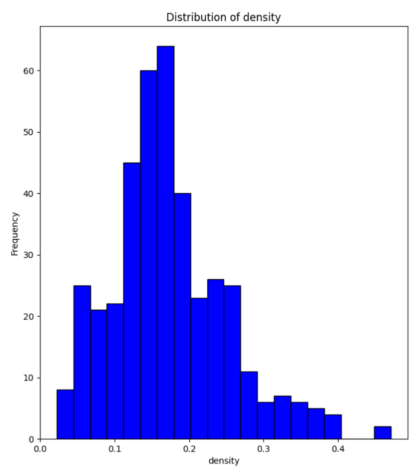

Figure 3: Distribution of density

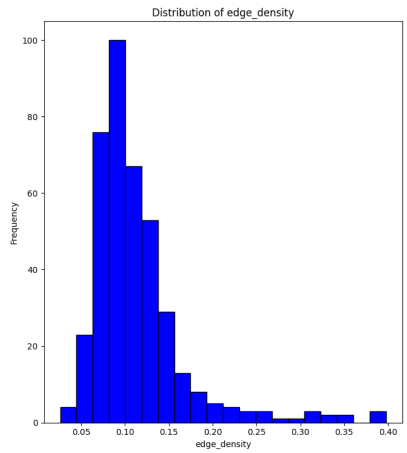

Figure 4: Distribution of edge density

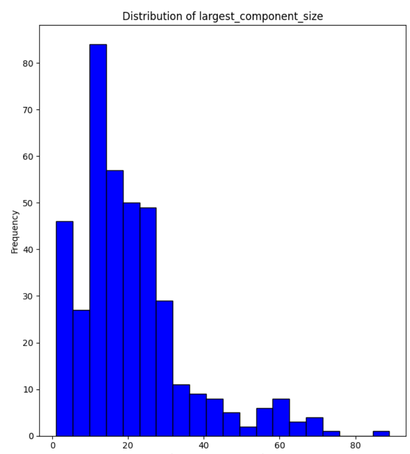

Figure 5: Distribution of largest component size

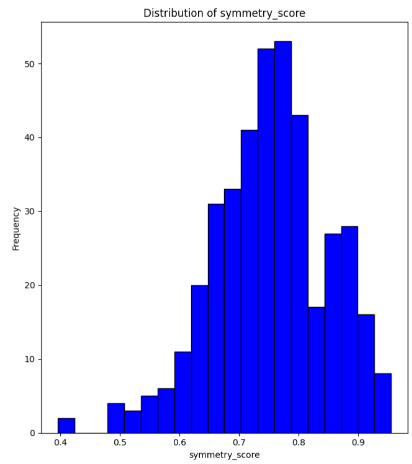

Figure 6: Distribution of symmetry score

3 Conclusion and Future Work

In this study, we applied morphological operations, such as erosion and dilation, on binary matrices to predict transformations using machine learning models. The results were consistent, with predictions of "false" indicating no complete transformation of matrix pixels. This suggests that the operations did not produce substantial changes in the input data. Future work could focus on refining the feature extraction process, incorporating more varied training data, and exploring alternative morphological techniques to improve model predictions. Additionally, expanding the study to include different types of binary matrices and more complex transformations could provide further insights into the model's capabilities.