Sugar centrifuge image classification

Abdalah Namwenje, Anjoreoluwa Boluwajoko, Matthews Sejeso, Ian Campbell, Freeman Nyathi, Montaz Ali

Industry representative: Richard Loubser

Mathematics in Industry Study Group
University of the Witwatersrand, Johannesburg

January 26, 2025

Overview

1. Introduction
2. Data
3. Convolution Neural Network
4. Sugar centrifuge image segmentation

Introduction

Problem Statement

This presentation is to address the challenge of separating the massecute area from the sugar area in the processing image. The solution needs to work regardless of variation in lighting shading or camera setting. The system must operate unsupervised with high level of automation.

Control of Purity Process

- The System captures an image of sugar and massecute (A mixture of sugar crystals and syrup).
- The goal is to differentiate between the clean sugar area and massecute using image processing.
- Based on the area fraction of clean sugar, the value is adjusted to control water or other factors.
- This process is iterative and automated.

Images Production

- The dataset provided consisted of 28 AVI video files.
- The task required converting the video data into images for classification purposes.
- Using Python's `cv2` and `os` libraries:

Images Production

(a) 85.4% purity

(b) 87.7% purity

(c) 86.1% purity

Algorithm Exploration

Convolution Neural Network

A Convolution Neural Network (CNN) is a type of deep learning model specifically designed to process structured data such as images, videos, and spatial or temporal data. CNN are widely used in tasks such as image recognition, object detection, natural language processing, and more.

Why CNN Model?

- Convolution neural networks help reduce the number of inputs nodes
- Convolution neural network takes pixels correlation into account.
- Convolution neural networks can tolerate a small shift of the pixels in the image.

How CNN works

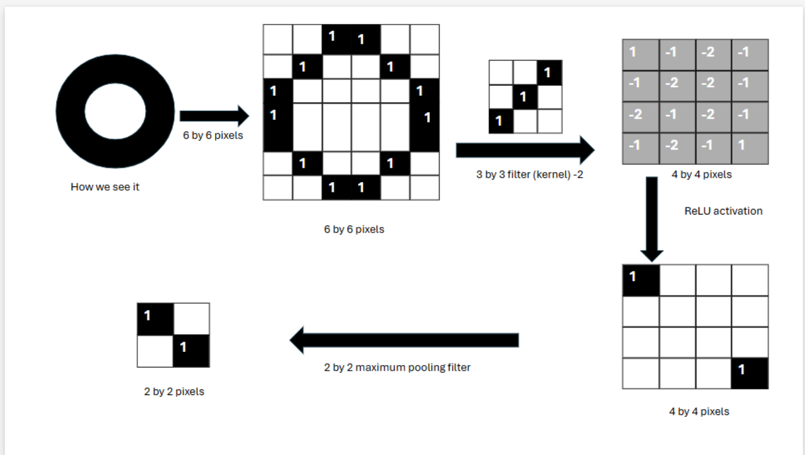


Figure: Input nodes reduction

How CNN works

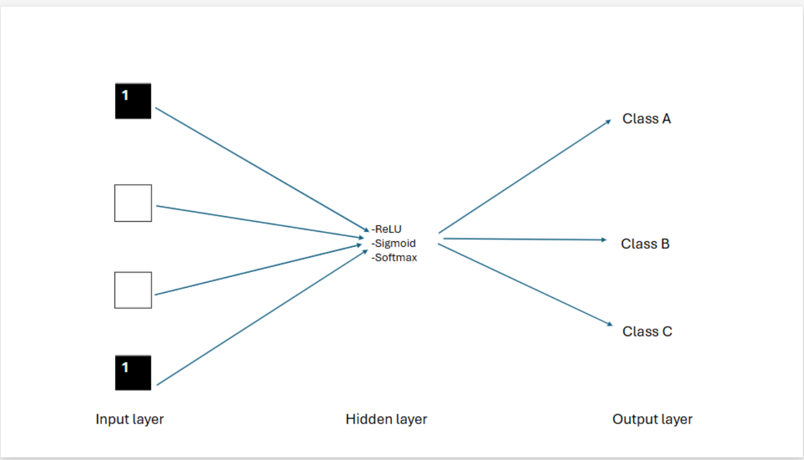


Figure: Input Classification

How CNN works

```python
# Define the model structure
def build_classification_model(input_shape, num_classes):
    model = Sequential([
        Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
        MaxPooling2D((2, 2)),
        Dropout(0.2),

        Conv2D(64, (3, 3), activation='relu'),
        MaxPooling2D((2, 2)),
        Dropout(0.2),

        Flatten(),
        Dense(128, activation='relu'),
        Dense(num_classes, activation='softmax')
    ])
    model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
    return model
```

Figure: Snippet of Code

Model Evaluation

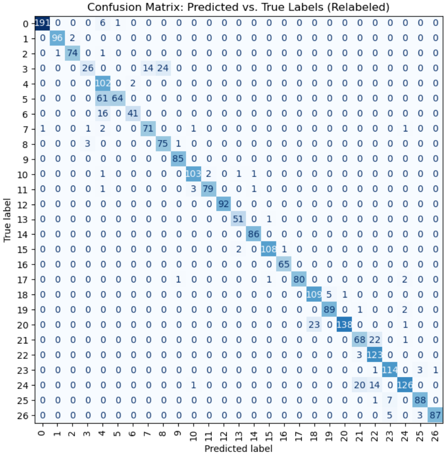


Figure: Outcome of the test

Relevance of the Model to the Problem

- The CNN classification model is suitable since the control process starts between 70% and 90% purity. Minor estimation errors within this range will not significantly affect results.
- Achieved **92% accuracy** in 10 iterations, classifying 2,485 out of 2,702 test images correctly. Used a total of 13,510 images (10,808 for training and 2,702 for testing). Robust against variations in lighting, shading, and camera settings.
- The CNN model inherently captures pixel correlations, effectively distinguishing massecute and sugar areas based on color differences.
- Inference times:
  - **CPU:** 5–20 ms per image (e.g., Intel i5, AMD Ryzen).
  - **GPU:** 1–5 ms per image (e.g., NVIDIA GPUs).
  - **Embedded devices:** Up to 50 ms on CPUs or 10 ms with accelerators like Google Coral Edge TPU.
- Operates efficiently within the required 2-second computation time and supports automation for unsupervised operations.

Sugar centrifuge image segmentation

Recap of the problem: Given the sugar centrifuge image, estimate the sugar purity.

The current implementation is the Otsu thresholding algorithm.

(a) Image

(b) Binary

Correlate the sugar purity with the sugar ratio, $s_r = \frac{N_i}{N}$, where N_i is the number of white pixels.

Sugar centrifuge image segmentation

The process of sugar centrifuge image segmentation involves the following steps:

- Preprocessing
- K-mean clustering algorithm
- Compute the Sugar ratio

Sugar image segmentation

Preprocessing the Image

- Crop the Image - Focus on the region of interest.
- Noise Reduction - Remove unwanted artifacts to improve segmentation accuracy.
- Contrast enhancement - Perform rescaling followed by histogram equalization to enhance the image contrast:

$$\bar{x} = \frac{P}{H - L}(x - L),$$

  where:

  - L: smallest pixel value in the image,
  - H: largest pixel value in the image,
  - P: maximum pixel value possible (commonly 255).

Sugar centrifuge image segmentation

K-Means clustering

K-Means clustering partitions n data points $\{x_i\}_1^n$ into k clusters $\{C_i\}_{i=1}^k$, minimizing the total variance within the cluster.

The goal is to minimize the following objective function,

$$F = \sum_{i}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2,$$

where:

- x is a data point.
- μ_i is the centroid of the cluster C_i, calculated as:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x.$$

Sugar centrifuge image segmentation

The k-means clustering algorithm:

1. Choose k initial centroids, $\mu_1, \mu_2, \ldots, \mu_k$, *randomly*.
2. Assign each data point x_j to the cluster with the nearest centroid:

$$C_i = \left\{ x_j : \|x_j - \mu_i\|_2^2 \leq \|x_j - \mu_l\|_2^2, \quad \forall l = 1, 2, \ldots, k \right\}.$$

3. Update the centroid of each cluster C_i to the mean of its points:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x.$$

4. Alternate between steps 2 and 3 until convergence, $\|\mu_i^{(t+1)} + \mu_i^{(t)}\|_2^2 \leq \epsilon$.

Sugar centrifuge image segmentation

We calculate the *sugar ratio*, s_r, using the formula:

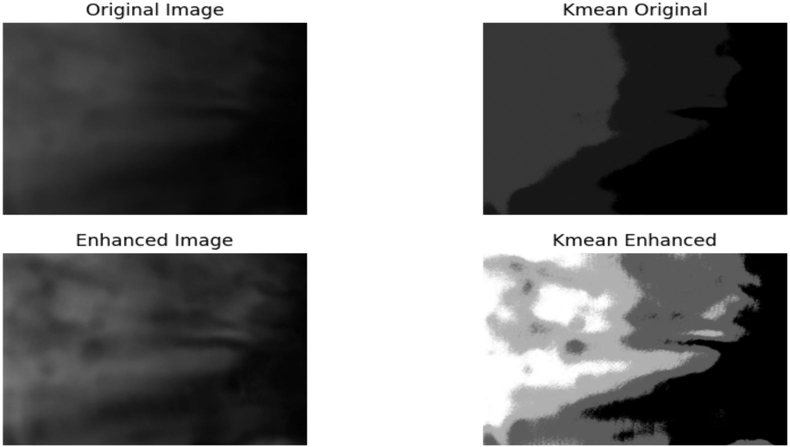
$$s_r = \frac{\sum_i^n \nu_i N_i}{P \times N},$$

where:

- ν_i: mean pixel value for cluster i,
- N_i: number of pixels in cluster i,
- $N = \sum_i N_i$: total number of pixels in the image.
- P: Maximum possible pixel value (commonly 255).

Sugar centrifuge image segmentation

The effect of contrast enhancement



Sugar image segmentation

Sugar ratio.

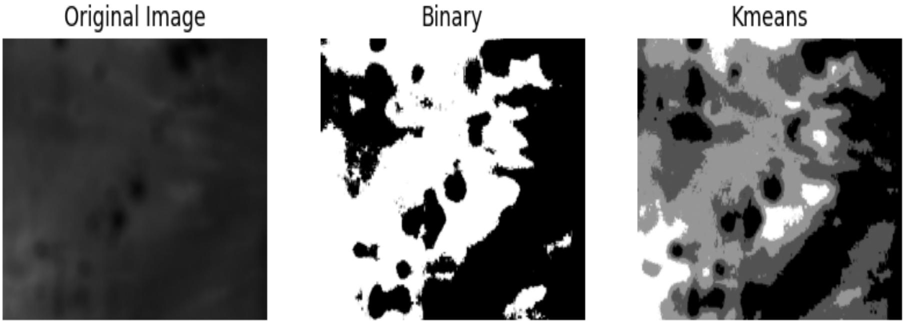


Figure: Sugar ratios: binary $= 0.6747$, k-means$=0.2964$

Sugar centrifuge image segmentation

Sugar ratio.

Figure: Sugar ratios: binary $= 0.7349$, k-means$=0.7096$

Conclusion

- The image segmentation approach shows promise; however, the algorithm requires significant improvement.
- Further steps include running the algorithm on the rest of the dataset and comparing the results with experimental data.

References

1. Loubser R. C., Barker B., Ndinisa N. V. and Mbuthuma Y.(2024). Using the C-centrifuge camera images to improve sucrose recovery. Proc South African Sugar Technologists Association, 96, 227-243.

2. Dhanachandra, N., Manglem, K., and Chanu, Y. J. (2015). Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. Procedia Computer Science, 54, 764-771.