Magnetic Tile Defect Detection: A Machine Learning Approach

Objective: Develop a machine learning model to classify images of magnetic tiles as 'defective' or 'non-defective,' further categorizing defective tiles by defect type.

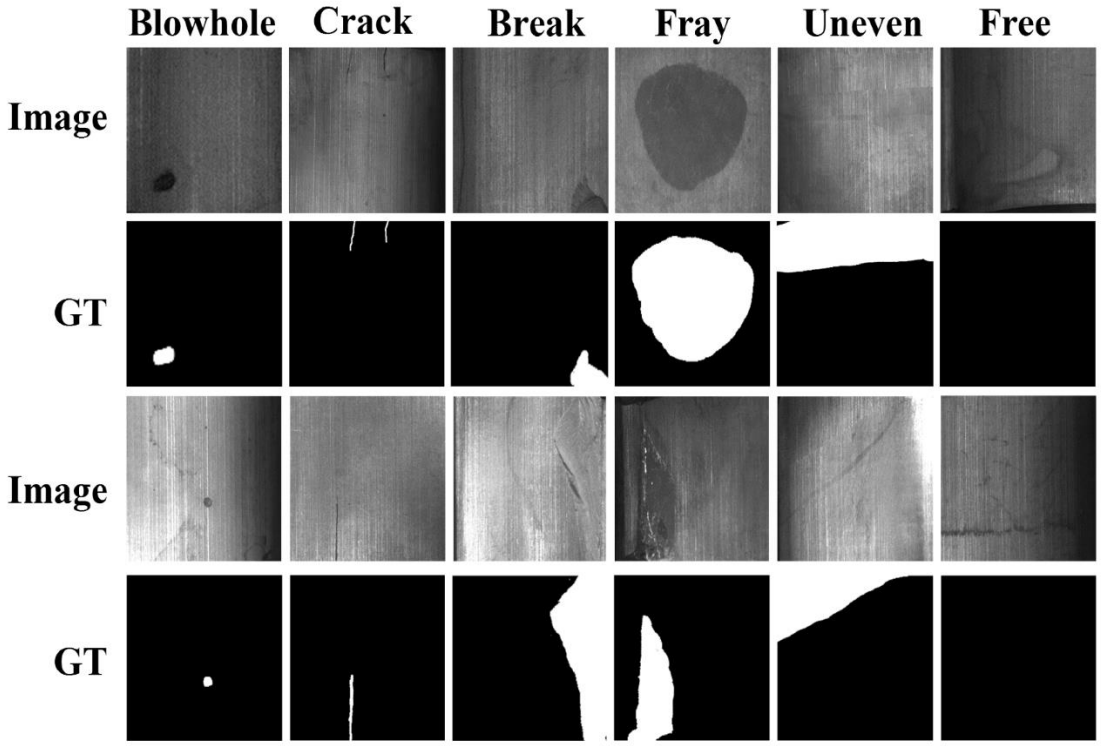
Deliverables: This report summarizes the methodology, model performance, and insights gained, accompanied by Jupyter Notebook files (image_preprocessing_and_data_visualization.ipynb and magnetic_tile_surface_defect_detection.ipynb) containing the code for data preprocessing, model training, and evaluation.

Dataset Description: Magnetic Tile Surface Defects

Surface defect detection is a critical process for quality control in manufacturing, particularly in industries like magnetic tile production. Manual inspection is often labour-intensive and prone to errors. Automating this process using image processing and machine learning techniques offers a significant opportunity to improve efficiency and product quality. However, automatic defect detection in magnetic tiles faces several challenges, including complex textures, diverse defect shapes (blowhole, crack, break, fray, unevenness), and varying illumination conditions.

This project utilizes a dataset specifically designed for magnetic tile surface defect detection. The dataset comprises images of magnetic tiles, categorized into six classes:

- **Defective-Blowhole:** Images of tiles exhibiting blowhole defects.
- **Defective-Crack:** Images of tiles with crack defects.
- **Defective-Break:** Images of tiles showing break defects.
- **Defective-Fray:** Images of tiles with fray defects.
- **Defective-Uneven:** Images of tiles with uneven surfaces.
- **Non-defective:** Images of tiles without any visible defects.



The dataset presents a significant class imbalance, with a substantially larger proportion of "Non-defective" (i.e. MT_Free) images compared to defect categories. This imbalance necessitates careful consideration during model training and evaluation to avoid bias towards the majority class.

The dataset also includes corresponding "ground truth" (GT) images for each original image. These GT images are binary masks highlighting the defective regions. While these masks could be valuable for tasks like segmentation or precise defect localization, they were not utilized in this project's classification task due to limitations in project scope and a focus on developing a classification model using the original images alone. Future work could explore incorporating the GT images for more advanced defect analysis.

Furthermore, the original dataset contained images of varying sizes. To ensure compatibility with the CNN model and maintain uniformity during training, all images were resized to a standard dimension of 256x256 pixels as a crucial preprocessing step.

1. **Dataset and Preprocessing** (image_preprocessing_and_data_visualization.ipynb)

This notebook focuses on preparing the image data for model training. It performs the following preprocessing steps:

1. **Data Cleaning**: Removes extraneous file formats (e.g., PNG) from the source directories, ensuring uniformity.
2. **Format Verification**: Checks and reports any non-JPG files, maintaining data integrity.
3. **Image Loading and Exploration**: Loads images and prints filenames, with optional display capabilities for visualization. This allows for qualitative assessment of the dataset
4. **Dataset Statistics**: Computes and displays the total image count and per-category counts, providing insight into class distribution.
5. **Image Preprocessing**: Standardizes the images using the following techniques:
    - Normalization: Scales pixel values to the [0, 1] range for improved model stability.
    - Resizing: Resizes all images to 256x256 pixels for consistent input dimensions.
    - Grayscale Conversion: Converts images to grayscale.
    - Histogram Equalization : Enhances contrast in images, though not ultimately utilized in the final model.
6. **Saving Pre-processed Data**: Stores the pre-processed images in organized subfolders, creating a structured dataset for efficient model training.
7. **Data Visualization:** Displays pre-processed and resized images for visual inspection of the transformations. (Another point for adding specific image displays.)
8. **Dataset Reorganization**: Moves images into category-specific subfolders within a unified directory structure ("Organized_Magnetic_Tile_Images").
9. **Ground Truth (GT) Data Creation**: Creates a standardized dataset for training within "GT" subfolders.
10. **GT Resizing**: Ensures all GT images have consistent dimensions (256x256).
11. **Size Verification:** Calculates and reports the average image size for each category, confirming consistent resizing.

2. **Model Training and Evaluation** (magnetic_tile_surface_defect_detection.ipynb)

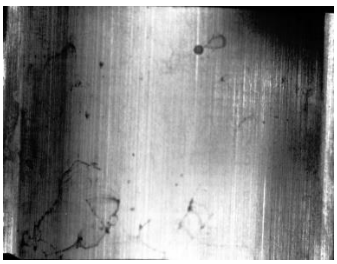
This notebook handles model training and evaluation:

1. **Environment Setup**: Installs and imports necessary libraries (TensorFlow, Keras, scikit-learn, etc.).
2. **Hardware Optimization**: Checks for and utilizes GPU availability for accelerated computation.
3. **Data Loading**: Loads the pre-processed image data and corresponding labels from the organized dataset. Categories are mapped to numerical indices.
4. **Data Splitting**: Splits the data into training (80%) and testing (20%) sets using stratified sampling.
5. **Model Architecture:** Defines the CNN model architecture.
6. **Model Compilation**: Compiles the model using Adam optimizer, categorical cross-entropy loss, and accuracy metric. Sets learning rate and batch size.
7. **Model Training**: Trains the model and prints training loss/accuracy per epoch.
8. **Model Evaluation**: Evaluates the model on the test set and reports the results.
9. **Model Persistence**: Saves the trained model ("resnet30_model.h5").
10. **Prediction Demonstration**: Provides code to load, preprocess, and classify new images.
11. **Performance Visualization**: Generates graphs of training/testing accuracy and loss, and a confusion matrix.

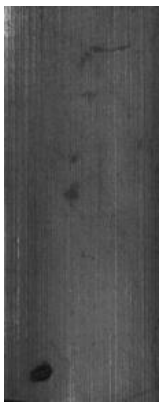
Image pre-processing sample images

Original image

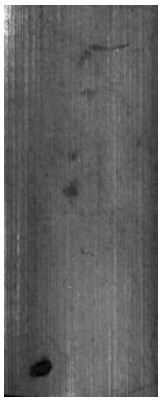
Equalized image



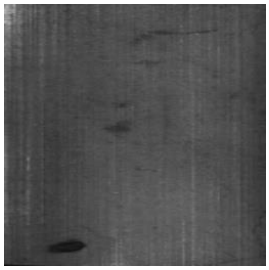
Gray scaled image



Normalized image



Resized image (256*256)



Model architecture:

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d_5 (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_6 (Conv2D)	(None, 125, 125, 64)	18,496
max_pooling2d_6 (MaxPooling2D)	(None, 62, 62, 64)	0
conv2d_7 (Conv2D)	(None, 60, 60, 128)	73,856
max_pooling2d_7 (MaxPooling2D)	(None, 30, 30, 128)	0
conv2d_8 (Conv2D)	(None, 28, 28, 256)	295,168
max_pooling2d_8 (MaxPooling2D)	(None, 14, 14, 256)	0
conv2d_9 (Conv2D)	(None, 12, 12, 512)	1,180,160
max_pooling2d_9 (MaxPooling2D)	(None, 6, 6, 512)	0
flatten_1 (Flatten)	(None, 18432)	0
dense_2 (Dense)	(None, 1024)	18,875,392
dense_3 (Dense)	(None, 6)	6,150

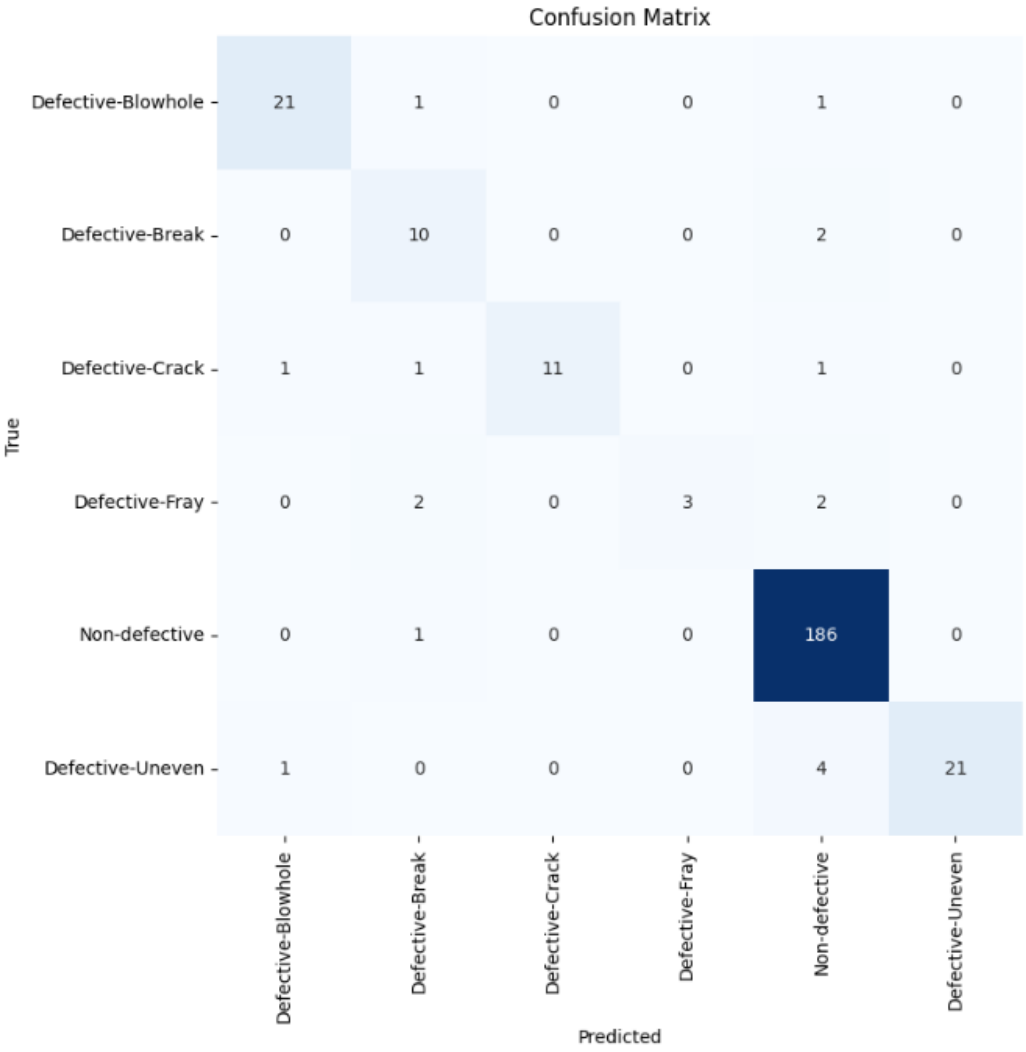
Total params: 20,450,120 (78.01 MB)

Trainable params: 20,450,118 (78.01 MB)

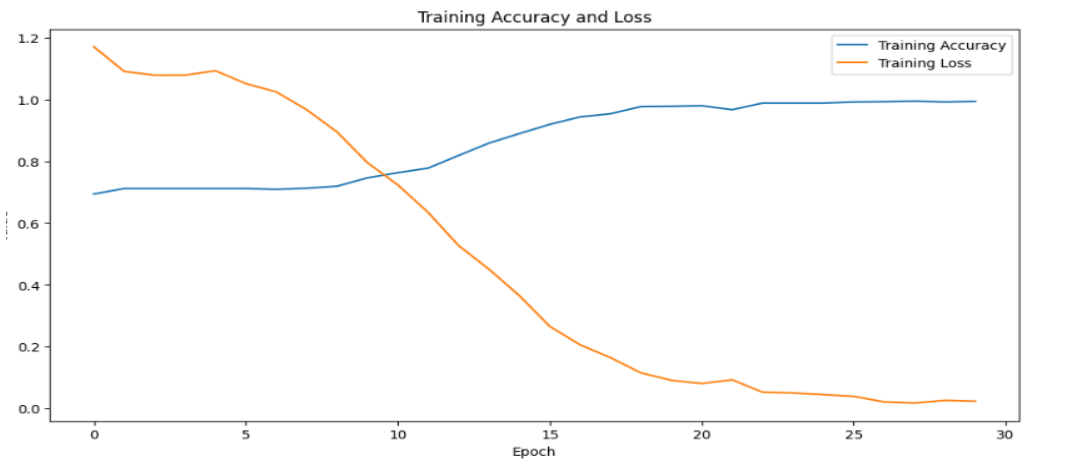
Non-trainable params: 0 (0.00 B)

Optimizer params: 2 (12.00 B)

Confusion matrix:



Graph:



Results and Discussion

The trained CNN model achieved an overall accuracy of 94% on the test set. This indicates that the model correctly classified 94% of the magnetic tile images. However, given the class imbalance in our dataset (a significantly larger number of "Non-defective" samples), it's essential to examine per-class performance metrics for a more comprehensive evaluation.

The following table presents the precision, recall, F1-score, and support for each category:

Class	Precision	Recall	F1-Score	Support
Defective-Blowhole	0.91	0.91	0.91	23
Defective-Break	0.67	0.83	0.74	12
Defective-Crack	1	0.79	0.88	14
Defective-Fray	1	0.43	0.6	7
Non-defective	0.95	0.99	0.97	187
Defective-Uneven	1	0.81	0.89	26
accuracy		0.94		269
macro avg	0.92	0.79	0.83	269
weighted avg	0.94	0.94	0.93	269

Key Observations from the Metrics and Confusion Matrix:

- High Overall Accuracy but Class-Specific Variations: While the overall accuracy is high (94%), the performance varies significantly across different defect categories. This highlights the impact of the class imbalance.
- Excellent Performance on Non-defective: The model excels at identifying "non-Defective" tiles, achieving high precision (0.95), recall (0.99), and F1-score (0.97). This is evident in the confusion matrix, with a large number of true positives for this class.
- Challenges with Defective-Fray: The model struggles the most with "Defective-Fray," as indicated by the low recall (0.43) and F1-score (0.60). This suggests difficulty in correctly identifying these defects. The confusion matrix likely shows a significant number of false negatives for this class.
- Good Performance on Blowhole, Crack, and Uneven: The model demonstrates good performance on "Defective-Blowhole," "Defective-Crack," and "Defective-Uneven" categories with high precision and generally good recall. However, the relatively lower recall for "Defective-Uneven" (0.81) suggests some misclassifications, possibly as "Non-defective," as seen in the confusion matrix.
- Defective-Break Performance: "Defective-Break" shows moderate performance with a lower precision (0.67) but a better recall (0.83). This indicates that the model might be over-classifying some other categories as "Defective-Break," leading to false positives.

Future Improvements

Based on the analysis of the model's performance, several potential improvements can be explored in future work:

1. Addressing "Defective-Fray" Challenges: The low recall for "Defective-Fray" indicates a significant area for improvement. Potential solutions include:
    - Data Augmentation: Applying aggressive data augmentation techniques (rotation, shearing, zooming, color jittering) specifically to the "Defective-Fray" images might improve the model's ability to learn the relevant features and generalize better.
    - More Training Data: Gathering more images of "Defective-Fray" tiles would likely be the most effective way to improve performance on this category.
    - Fine-Grained Classification: If "Defective-Fray" has visual variations within the category itself, creating subcategories or using a more specialized model for this specific defect type could be beneficial.
2. Mitigating Class Imbalance Effects: While stratified sampling helps, further techniques can be employed:
    - Cost-Sensitive Learning: Assigning higher weights to the minority classes during training can penalize misclassifications more heavily and improve their recall.
    - Oversampling/Under sampling: Techniques like SMOTE (Synthetic Minority Over-sampling Technique) can generate synthetic samples for minority classes, while under sampling can reduce the number of majority class samples, creating a more balanced training set.
3. Improving "Defective-Break" Precision: The relatively lower precision for "Defective-Break" suggests the model is misclassifying other categories as "Defective-Break."
    - Error Analysis: Carefully examine the images misclassified as "Defective-Break" to identify common patterns or features that might be causing the confusion. This analysis can guide further improvements to the model or preprocessing steps.
    - Feature Engineering: Explore additional features that might help distinguish "Defective-Break" from other categories more effectively.

4. Hyperparameter Tuning & Model Architecture:
    - Systematic Hyperparameter Search: Utilize techniques like grid search or Bayesian optimization to systematically explore different hyperparameter combinations and identify optimal settings. Focus on parameters like learning rate, batch size, dropout rates, and the number of layers/filters in the CNN.
    - Alternative Architectures: Experiment with different CNN architectures (e.g., deeper ResNets, EfficientNets, or custom architectures) to potentially improve feature extraction and classification.
    - Transfer Learning: Consider using pre-trained models as a starting point and fine-tuning them on your dataset. This can leverage the knowledge learned from large datasets and often leads to faster convergence and better performance.

5. Use of GT images from the original dataset to train the model can improve accuracy and precision, as demonstrated in the research "Surface Defect Saliency of Magnetic Tile" by Huang, Y., Qiu, C., Guo, Y., Wang, X., & Yuan, K. (2018), presented at the IEEE 14th International Conference on Automation Science and Engineering (CASE). In their study, they used a U-Net model in deep learning.

Conclusion

This project successfully developed a machine learning model for multi-class magnetic tile defect detection, achieving an overall accuracy of 94% on the test set. The model, a Convolutional Neural Network (CNN), was trained on a pre-processed dataset of magnetic tile images, demonstrating its ability to distinguish between various defect types ("Blowhole," "Break," "Crack," "Fray," "Uneven") and "Non-defective" tiles. A crucial aspect of this project involved addressing the inherent class imbalance in the dataset, primarily achieved through stratified sampling during the training and testing phases. Data augmentation and preprocessing techniques, including normalization and resizing, played a critical role in enhancing model performance and robustness.

While the overall accuracy is encouraging, a detailed analysis using a confusion matrix and per-class metrics (precision, recall, and F1-score) provided a more nuanced understanding of the model's performance. The analysis revealed significant variations in performance across different defect categories. The model excelled at classifying "Non-defective" tiles but encountered challenges with "Defective-Fray," highlighting an area for targeted improvement. The evaluation also revealed performance characteristics for other defect types, such as the relatively lower precision for "Defective-Break," suggesting potential areas for further investigation.

Despite these identified areas for improvement, the developed multi-class model demonstrates the feasibility and effectiveness of CNNs for automated magnetic tile defect detection in an industrial setting. Future work will focus on enhancing the model's performance by addressing the observed limitations. Potential strategies include targeted data augmentation for underperforming categories, exploring cost-sensitive learning to mitigate class imbalance effects, and experimenting with alternative CNN architectures or hyperparameter tuning techniques for improved feature extraction and classification accuracy. By addressing these challenges, the model can be refined to provide even more reliable and efficient defect detection, ultimately improving the quality control processes in magnetic tile manufacturing.