جامعة جدة
University of Jeddah

Report for Thermal Images Data Classification

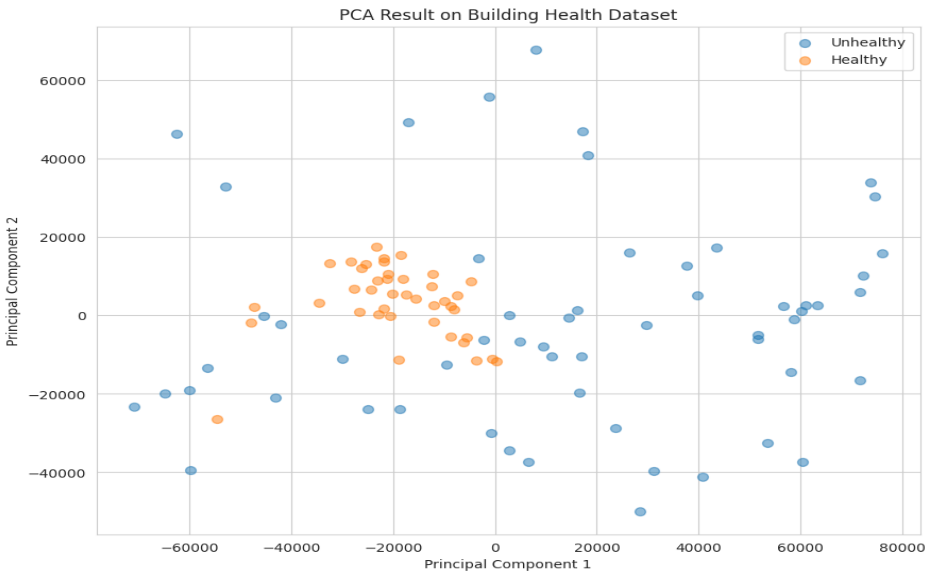
Report content:

- Principal component analysis
- Images Data Augmentation Techniques Used
- Model Architecture CNN
- Model Configuration and Training
- Training and Validation Report
- Classification and Confusion Matrix Report
- Finally, Observations and recommendations about the data

Prepared by \\ Machine Learning Dev. Muhannad Mansour

- Principal component analysis:

Principal Component Analysis (PCA): Reduce the dimensionality of your images and visualize them in a 2D or 3D space. Helps in visualizing clusters or separations between classes.



Spread of Unhealthy Images: The "Unhealthy" images are more spread out across the PCA plot. This could indicate a wider variety in the visual features of "Unhealthy" images compared to "Healthy" ones.- Overlap Between Classes: There's a clear overlap between the "Healthy" and "Unhealthy" classes in the reduced feature space. This overlap suggests that, based on the raw pixel values of the images, there's similarity between some of the "Healthy" and "Unhealthy" images. It's possible that a linear classifier (like Logistic Regression) might struggle to perfectly separate these two classes based on these principal components.

Density of Healthy Images: There's a dense cluster of "Healthy" images around the center of the plot. This suggests that a significant portion of the "Healthy" images are similar to each other in this reduced feature space.

Spread of Unhealthy Images: The "Unhealthy" images are more spread out across the PCA plot. This could indicate a wider variety in the visual features of "Unhealthy" images compared to "Healthy" ones.

Separation along Principal Component 1: While there's overlap, some of the "Unhealthy" images tend to have higher values on the Principal Component 1 axis, suggesting that this component does capture some variance that can be useful for differentiating the two classes.

- Images Data Augmentation Techniques Used:

1. Rescale:

This normalizes the pixel values to the range [0,1]. For an image, it means that originally, the pixels range from 0-255, but post rescaling, they'll range between 0 and 1.

2. Rotation Range:

The image will be rotated by a random angle between -15 to +15 degrees.

3. Zoom Range:

The image will be zoomed in or out by a factor randomly picked within the range [1-0.15, 1+0.15].

4. Width Shift and Height Shift:

The image will be shifted horizontally or vertically by a fraction of its width or height, respectively.

6. Shear Range:

The image will undergo shear transformation in the range [-0.15, 0.15].

7. Horizontal and Vertical Flip:

Randomly decide to flip images. It's useful when there's no assumption of horizontal or vertical symmetry in the dataset.

8. Channel Shift Range:

Randomly perturbs the channels (like R, G, B for color images) by a factor within the range [-0.2, 0.2].

9. Fill Mode:

Uses the nearest neighboring pixel values to fill the new pixels. Other methods like "constant", "reflect", etc., have different strategies for filling.

10. Cval:

When pixels are created after a transformation and if fill_mode is "constant", these pixels will be assigned a value of 0.

11. Data Format:

Indicates the ordering of the dimensions in the data. In this case, the channels (like R, G, B) come last. The alternative is "channels_first".

12. Validation Split:

Reserves 20% of the data for validation purposes.

When the training process begins, train_generator will provide augmented images (as specified in datagen) from the directory in batches of 32 for training. Similarly, the validation_generator will supply augmented images from the reserved validation subset for validating the model's performance after each epoch (or as specified in the training loop).

- Model Architecture CNN:

1. Model Type: (Sequentia) The model is built using Keras' Sequential API, which means that it's a linear stack of layers where you can just add one layer at a time.

2. Convolutional Neural Network (CNN): The model is designed for image classification, utilizing multiple layers of 2D convolutions followed by max-pooling to extract features from images.

3. Architecture Depth: The network contains three sets of Conv2D and MaxPooling2D layers, increasing filter complexity as the network goes deeper. This is followed by dense layers for classification.

4. Regularization: A Dropout layer is incorporated with a rate of 50% to prevent overfitting, ensuring that the model generalizes well to new data.

5. Optimizer & Learning Rate: The Adam optimizer is used with a learning rate of 0.0001 for efficient and adaptive weight updates.

6. Classification & Loss: The final output uses a softmax activation for binary classification, and the model is trained using binary cross-entropy as the loss function.

- ## Model Configuration and Training:

1.K-Fold Cross-Validation:

- The StratifiedKFold is employed to conduct a 5-fold cross-validation (as n_splits=5).
- Stratification ensures that each fold retains the same percentage of samples for each class, crucial for imbalanced datasets.

2.Data Generators with DataFrames:

- Image paths and labels are sourced from the image_df DataFrame.
- The datagen.flow_from_dataframe method loads and augments images on-the-fly based on the DataFrame.
- Images are resized to 256x256 pixels and converted to grayscale (color_mode='grayscale').
- The training generator shuffles data, whereas the validation generator does not.

3.Class Weights for Imbalanced Data:

- Computed using the compute_class_weight method, ensuring that the model pays more attention to under-represented classes.
- The weights are used during training to give more importance to under-represented classes.

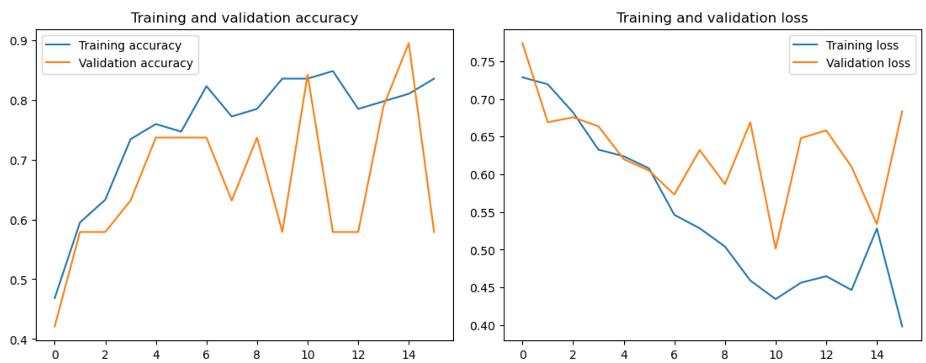
4.Early Stopping:

- If the validation loss doesn't improve for 5 epochs (patience=5), the training is halted.
- The model weights from the best epoch (in terms of validation loss) are restored.

5.Training Procedure:

- The model is trained for up to 25 epochs (epochs=25).
- The training and validation generators are used to supply data.
- The class_weights_dict is applied during training to handle class imbalance.
- Training could stop early based on the early_stopping callback.

- <u>Training and Validation Report:</u>



1.Consistent Growth in Training Accuracy:

The training accuracy (blue curve) steadily increases across the epochs, demonstrating that the model is effectively learning patterns from the training data.

2.Validation Accuracy Variability:

The validation accuracy (orange curve) exhibits variability and does not show a consistent increase. It experiences noticeable fluctuations, particularly around epochs 6-10 and post epoch 12.

This variability suggests that the model might not be generalizing perfectly to unseen data. It could also indicate the effects of different subsets of data presented in each fold of cross-validation.

3.Divergence between Training and Validation Metrics:

Around epoch 6 and beyond, there is a noticeable divergence between training accuracy and validation accuracy, as well as between training loss and validation loss. The divergence implies potential overfitting, where the model might be memorizing the training set instead of learning generalized features.

4.Training Loss Decrease:

The training loss (blue curve in the second graph) consistently decreases, which is an expected behavior as the model optimizes its weights over epochs.

5.Erratic Validation Loss:

The validation loss (orange curve in the second graph) is more erratic. It decreases initially until around epoch 6 but then starts showing significant oscillations.

The abrupt increase in validation loss post epoch 12 suggests that the model's predictions on the validation set are deviating from the actual labels, strengthening the suspicion of overfitting.

Impact of Limited and Imbalanced Data on Model Results:

1.Overfitting on Small Datasets:

With only 93 images, the dataset is considerably small for training deep neural networks. This limitation often leads to overfitting, where the model starts to memorize specific details of the training images rather than learning generalized features. The divergence between training and validation metrics, as observed in your results, is a strong indicator of this overfitting phenomenon.

2.Limited Representativeness:

Small datasets might not comprehensively capture the variability and diversity present in the broader category of images the model is supposed to recognize.

This lack of representativeness can lead to poor generalization on new, unseen data.

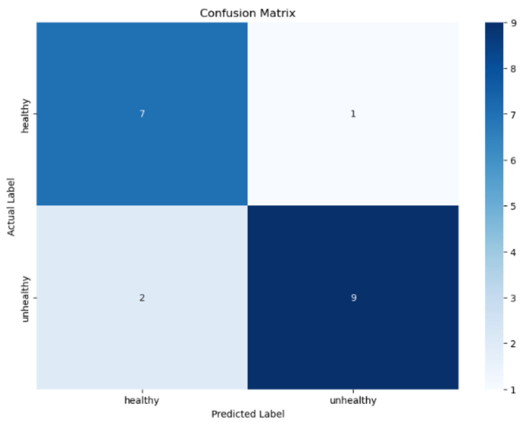
3.Imbalance Affects Model Biases:

An imbalanced dataset can make the model biased towards the class with more samples. As a result, the model might achieve a deceptively high accuracy by predominantly predicting the majority class. The erratic validation accuracy in the results could be influenced by this imbalance.

4.Difficulty in Validating the Model:

When dealing with a small number of images, the validation set itself becomes quite limited. This can make the validation metrics more sensitive to the particular images chosen for validation in each fold. The oscillations observed in the validation accuracy and loss could be exacerbated by this small validation set size.

- Classification and Confusion Matrix Report:



1.Confusion Matrix Breakdown:

True Positive (TP): 7

The model correctly predicted 7 instances as "healthy" which were indeed "healthy".

True Negative (TN): 9

The model accurately identified 9 instances as "unhealthy" which were truly "unhealthy".

False Positive (FP): 1

The model incorrectly predicted 1 instance as "unhealthy" when it was actually "healthy".

False Negative (FN): 2

The model wrongly classified 2 instances as "healthy" when they were "unhealthy".

2.Performance Metrics:

Precision: 0.78

Of all instances predicted as "healthy", 78% were truly "healthy".

Recall: 0.88

Of all actual "healthy" instances, the model correctly identified 88% of them.

F1-Score: 0.82

The harmonic mean of precision and recall for "healthy" predictions is 82%.

Unhealthy Class:

Precision: 0.90

Of all instances predicted as "unhealthy", 90% were genuinely "unhealthy".

Recall: 0.82

Of all actual "unhealthy" instances, the model correctly identified 82% of them.

F1-Score: 0.86

The harmonic mean of precision and recall for "unhealthy" predictions is 86%.

Overall Metrics:

Accuracy: 0.84

The model's overall correctness across both classes is 84%.

Macro Average F1-Score: 0.84

The average F1-Score for both classes is 84%.

Weighted Average F1-Score: 0.84

The F1-Score, weighted by the number of true instances for each label, is 84%.

3.Key Observations:

Balanced Precision and Recall:

Both classes exhibit a good balance between precision and recall. While there are minor discrepancies between these metrics for each class, the model seems to perform well in not only predicting a class but also in capturing the majority of true instances of that class.

Higher Precision for 'Unhealthy':

The model exhibits a slightly higher precision for "unhealthy" predictions, meaning when it predicts "unhealthy", it's more likely to be correct compared to its "healthy" predictions.

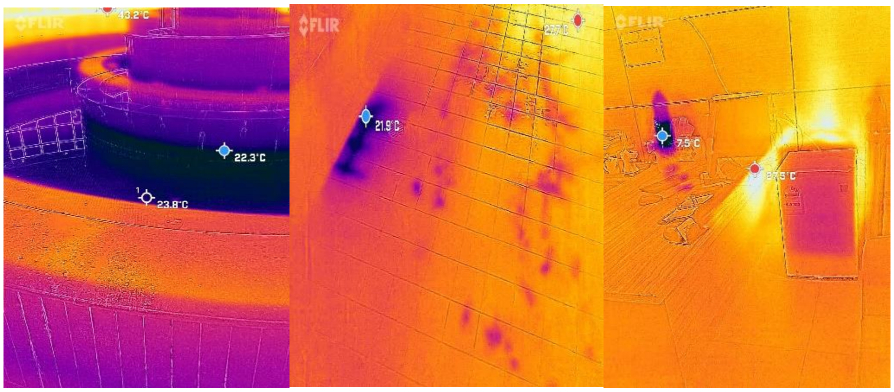
Misclassifications Exist:

The model misclassified 3 out of 19 instances. Specifically, it tends to mislabel "unhealthy" instances as "healthy" more often, as indicated by the 2 false negatives.

- **<u>Finally, Observations and recommendations about the data</u>**

I first wanted to review here some of the images that were taken, which have some problems and are not compatible with the purpose of the project, which is to identify the internal brickwork of the construction elements in the buildings.

Let's look at a sample of unhealthy images:



The initial evaluation of the provided imagery does not show conclusive evidence of unsanitary conditions. Specific areas of concern include:

a. There is no discernible indication of internal leaks within walls, columns, beams, ceilings, or floors.

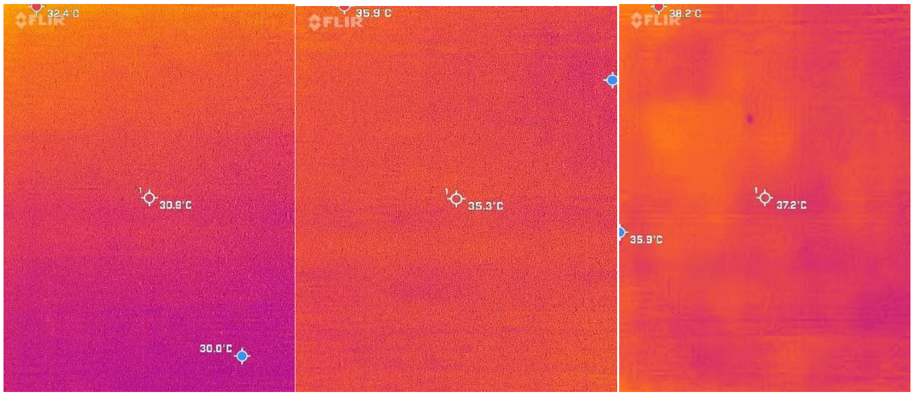
b. The first image showcases a fountain pool filled with water, which does not inherently indicate an unsanitary condition.

c. The second image depicts surface water on bathroom floors. While this could be a potential slip hazard, it does not necessarily denote internal humidity or unsanitary conditions.

d. The third image shows a desk and there is a refrigerator next to it. It is logical that it generates a lower temperature. There is also a bottle of cold water on the desk and there is no at any unhealthy structural element.

Some of the images appear to be repeated, which might have limited the range and depth of data provided.

Let's look at a sample of healthy images:



As I mentioned previously in the PCA section, health images are very similar and repetitive.

It would have been better to take pictures on a large scale in the place with temperature differences, and not in this way, which may be somewhat misleading.

For example, there are many unhealthy images that do not have any internal humidity problems, and this is what I mean by data imbalance. Also, the repetition of images in the two categories is also considered an imbalance, and this makes the model not generalize to all Patterns of data

It is better for the images that are collected to be as shown below for unhealthy images with temperature differences.