Portfolio Task 1 - Anomaly Detection: Strategy and Decisions

1. Understanding the Task

Our task was to develop an automated method to detect cracks in infrastructure using image data. We were given only healthy (non-cracked) images for training. The goal was to learn the characteristics of healthy structures and detect anomalies - in this case, cracks - during testing. This situation matches a one-class learning problem.

2. Options Considered: Generative Models

We were instructed to select one of the following models:
- Autoencoder
- Variational Autoencoder (VAE)
- Diffusion Model

Each of these models can be trained on a single class (healthy images) and can recreate input images based on the patterns it has learned.

3. Why We Chose Variational Autoencoder (VAE)

We selected the Variational Autoencoder (VAE) over a basic Autoencoder or a Diffusion Model for the following reasons:

- Compared to a regular Autoencoder, a VAE introduces a probabilistic latent space. This makes the model more robust to small variations and improves generalization.
- Diffusion Models are very powerful but computationally expensive. Training and sampling are slow, which makes them less suitable for fast iteration and real-time inference.
- VAE strikes a balance: it's more expressive than a plain Autoencoder, but far less complex than Diffusion Models.

Therefore, VAE is a practical and efficient choice for learning healthy image patterns and detecting subtle deviations like cracks.

4. Training Philosophy

Since we only had non-cracked images, we trained the VAE to learn what a normal, undamaged surface looks like. The model learns to compress these images and then reconstruct them. After training, if a test image contains a crack, the VAE is unable to accurately recreate it, resulting in a higher reconstruction error.

Portfolio Task 1 - Anomaly Detection: Strategy and Decisions

5. Strategy to Detect Anomalies

Once the model is trained, we calculate the difference between the input image and the reconstructed image. This difference is the anomaly score. A small difference means the image looks like what the model has seen before. A large difference means the image is unfamiliar - potentially cracked.

To separate normal from cracked images, we used a threshold based on the reconstruction errors of healthy images (validation set).

6. Thresholding Method

We computed the mean and standard deviation of reconstruction errors on a validation set of healthy images. The threshold for detecting anomalies was set as:

Threshold = mean + 3 × standard deviation

This is a common heuristic in anomaly detection to flag outliers while minimizing false positives.

7. Evaluation Criteria

We used a separate test set that included both healthy and cracked images. Our model's predictions were compared to ground truth labels.

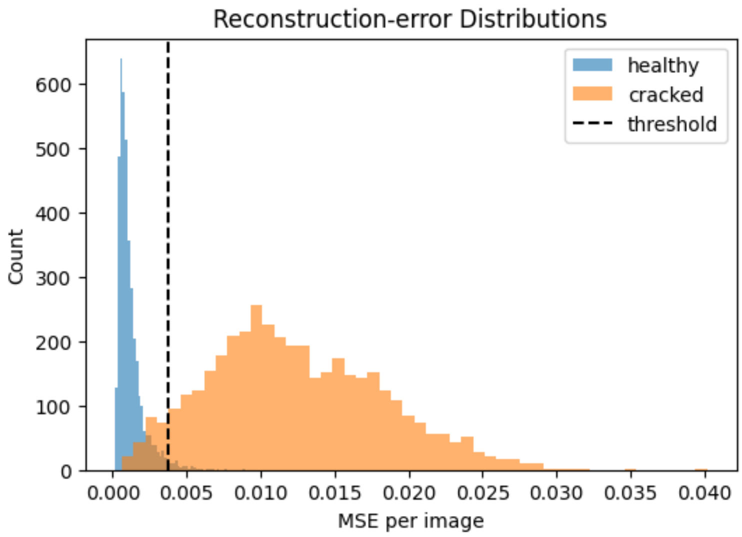
We used:
- ROC AUC Score: Measures how well the model separates the two classes
- Precision, Recall, and F1 Score: These metrics help evaluate detection quality

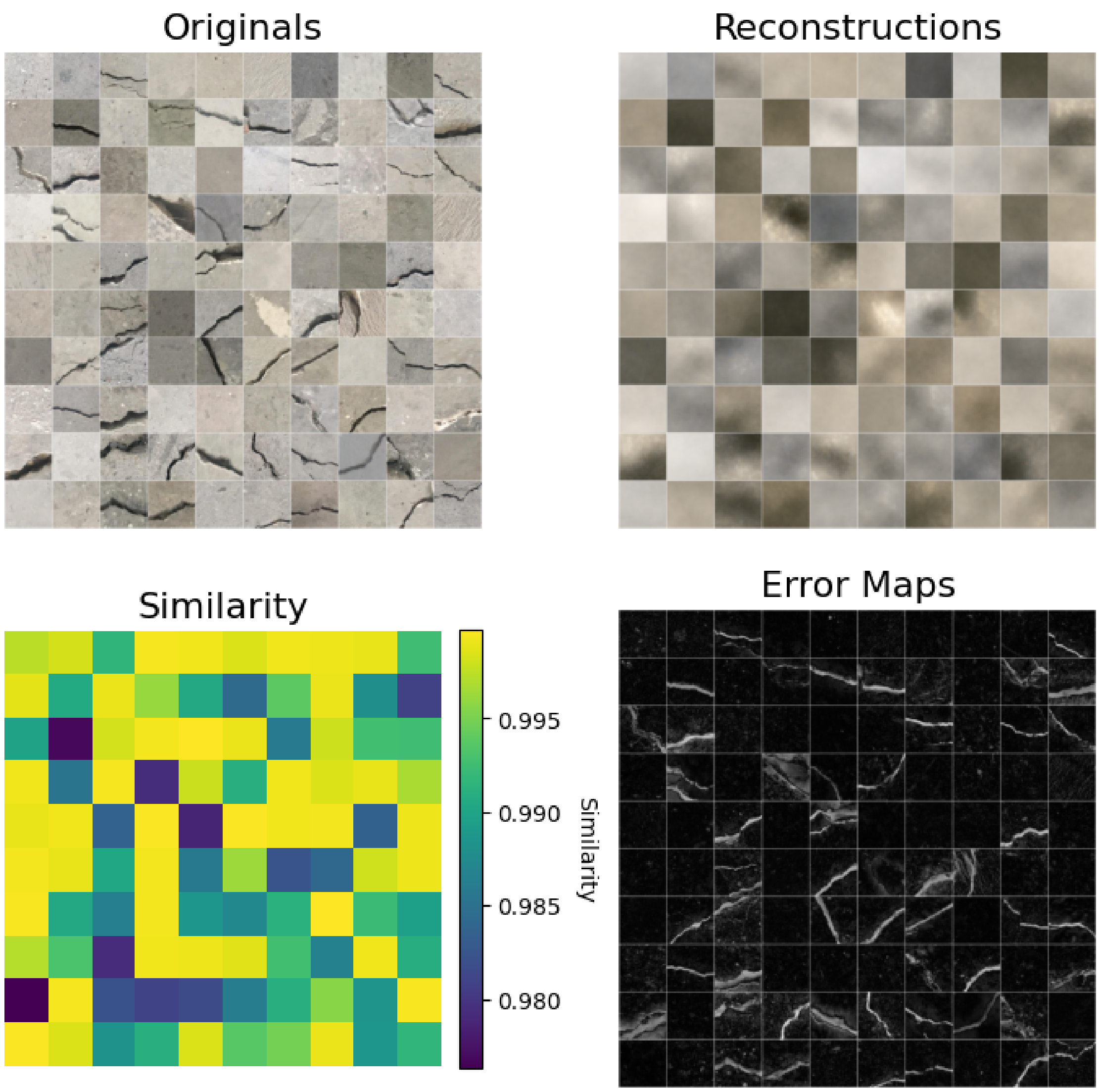
These metrics were chosen because they are standard for imbalanced classification and anomaly detection.

8. Visualizations Used

We created the following visual outputs to validate our method:
- Reconstruction-error distribution: shows how cracked and healthy images differ in reconstruction loss
- Grid of original vs reconstructed images: visual confirmation of model performance
- Similarity and Error Map visualizations: highlight where cracks are detected and how well each image was reconstructed

Portfolio Task 1 - Anomaly Detection: Strategy and Decisions





9. Similarity Calculation Choice

Although similarity can be measured in many ways, we deliberately chose Mean Squared Error (MSE) as the reconstruction loss. We did not use cosine similarity because it measures angular difference between vectors - useful in NLP or embeddings, but not appropriate for image reconstruction where spatial pixel-wise accuracy matters more.

10. Summary of Approach

Our solution uses a VAE to model healthy concrete structure images and detect deviations. We designed our detection pipeline to use reconstruction error as an anomaly signal. Thresholding was done statistically using validation data. Evaluation was based on common anomaly detection metrics.

Every decision - from model selection to metric choice - was based on balancing effectiveness, interpretability, and practical constraints.