EDA

March 23, 2025

1 Exploratory Data Analysis of Crack Morphology

Shantanu Motiani (13757830) MSc IS:DS

GitHub

1.1 Introduction

This notebook presents an in-depth exploration of a crack segmentation dataset with a focus on unsupervised learning techniques for feature extraction and pattern discovery in crack morphology. This analysis aims to discover natural patterns and groupings within crack morphologies that could inform more effective pavement performance estimations.

1.2 Dataset Overview

The dataset consists of images of various structural surfaces along with corresponding binary masks indicating the presence of cracks. These masks serve as ground truth for segmentation tasks and provide valuable information about crack morphology and distribution patterns. By analyzing these masks in depth, we can gain insights into the diverse characteristics of cracks that appear in real-world scenarios.

This crack segmentation dataset contains around 11.200 images which are merged from 12 available crack segmentation dataset.

There're also images which contain no crack, which could be filtered out by the pattern "noncrack*" All the images in the dataset are resized to the size of (448, 448).

References for original datasets:

CRACK500: >@inproceedings{zhang2016road, title={Road crack detection using deep convolutional neural network}, author={Zhang, Lei and Yang, Fan and Zhang, Yimin Daniel and Zhu, Ying Julie}, booktitle={Image Processing (ICIP), 2016 IEEE International Conference on}, pages={3708–3712}, year={2016}, organization={IEEE} }' .

> @article{yang2019feature, title={Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection}, author={Yang, Fan and Zhang, Lei and Yu, Sijia and Prokhorov, Danil and Mei, Xue and Ling, Haibin}, journal={arXiv preprint arXiv:1901.06340}, year={2019} }

GAPs384: >@inproceedings{eisenbach2017how, title={How to Get Pavement Distress Detection Ready for Deep Learning? A Systematic Approach.}, author={Eisenbach, Markus and Stricker, Ronny and Seichter, Daniel and Amende, Karl and Debes, Klaus and Sesselmann, Maximilian and

Ebersbach, Dirk and Stoeckert, Ulrike and Gross, Horst-Michael}, booktitle={International Joint Conference on Neural Networks (IJCNN)}, pages={2039–2047}, year={2017} }

CFD: >@article{shi2016automatic, title={Automatic road crack detection using random structured forests}, author={Shi, Yong and Cui, Limeng and Qi, Zhiquan and Meng, Fan and Chen, Zhensong}, journal={IEEE Transactions on Intelligent Transportation Systems}, volume={17}, number={12}, pages={3434–3445}, year={2016}, publisher={IEEE} }

AEL: >@article{amhaz2016automatic, title={Automatic Crack Detection on Two-Dimensional Pavement Images: An Algorithm Based on Minimal Path Selection.}, author={Amhaz, Rabih and Chambon, Sylvie and Idier, J{'e}r{^o}me and Baltazart, Vincent} }

cracktree200: >@article{zou2012cracktree, title={CrackTree: Automatic crack detection from pavement images}, author={Zou, Qin and Cao, Yu and Li, Qingquan and Mao, Qingzhou and Wang, Song}, journal={Pattern Recognition Letters}, volume={33}, number={3}, pages={227–238}, year={2012}, publisher={Elsevier} }

1.3 Methodology

This EDA follows three main phases:

1.3.1 1. Mask Analysis and Feature Extraction

We begin by extracting meaningful features from the binary crack masks. Rather than relying solely on raw pixel data, we compute a rich set of morphological features that characterize crack patterns:

- **Density metrics**: Percentage of crack pixels relative to the total image area
- **Connectivity measures**: Number of connected components representing individual cracks
- **Geometrical properties**: Crack area, length, width, and orientation distributions
- **Complexity indices**: Tortuosity (ratio of actual path length to Euclidean distance), fractal dimension (measuring how crack patterns fill space across scales), and branching.

These handcrafted features provide interpretable measures that relate directly to physical crack properties, enabling both quantitative analysis and domain-informed interpretation. This will also add an extra dimension of interpretability when we use more 'black-box' approaches to learn visual features without labels.

1.3.2 2. Unsupervised Learning of Visual Features

While handcrafted features offer interpretability, they may miss subtle visual patterns. To complement our feature set, we leverage deep learning techniques to extract rich visual representations:

- We employ a pre-trained ResNet model as a feature extractor
- The deep layers of the network capture high-level visual patterns and contextual information
- Principal Component Analysis (PCA) is applied to reduce dimensionality while preserving the most informative variations
- t-SNE and UMAP projections help visualize the high-dimensional feature space in two dimensions

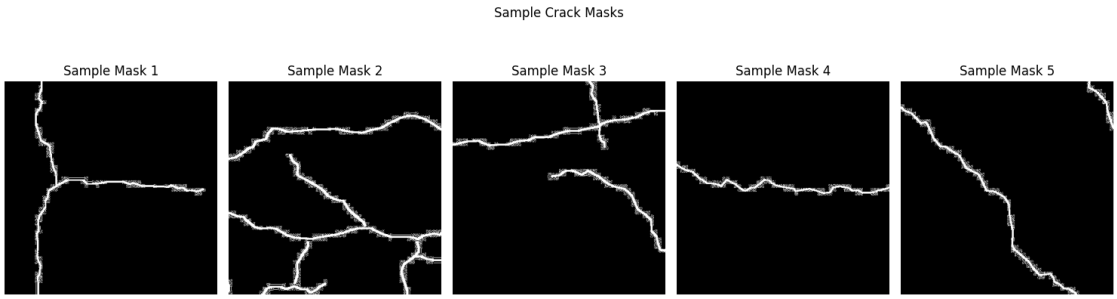
This combination of handcrafted and learned features creates a comprehensive representation of crack characteristics across multiple levels of abstraction.

1.3.3 3. Cluster Analysis and Visualization

To discover natural groupings within the dataset, we apply K-means clustering to the combined feature space:

- Optimal cluster count determined through silhouette analysis, Davies-Bouldin index, and Calinski-Harabasz score
- Cluster centroids represent "prototypical" crack patterns
- Various visualization techniques (scatter plots, radar charts, feature importance analysis) help interpret cluster characteristics
- Representative samples from each cluster illustrate the visual patterns captured

```
WARNING: CPU random generator seem to be failing, disabling hardware random
number generation
WARNING: RDRND generated: 0xffffffff 0xffffffff 0xffffffff 0xffffffff

Starting analysis of crack segmentation masks…
Found 2000 mask files
```




```
Extracting features from masks…

100%|     | 2000/2000 [13:57<00:00,  2.39it/s]

Successfully extracted features from 2000 masks

Feature statistics:
           density   num_cracks  avg_crack_area  max_crack_area  \
count  2000.000000  2000.000000     2000.000000     2000.000000
mean      0.034567    56.390500     2961.070768     5941.014000
std       0.036475    73.172639     6204.225282     6934.387102
min       0.000000     0.000000        0.000000        0.000000
25%       0.013053     1.000000       40.999936     1867.000000
50%       0.024870     5.000000       65.618992     3983.500000
75%       0.043568   100.250000     2534.950000     7296.000000
max       0.401716   435.000000    44848.000000    80458.000000

       total_crack_area  avg_crack_length  max_crack_length  \
count       2000.000000       2000.000000       2000.000000
mean        6937.700500        326.366684       5941.014000
```

std	7320.646626	489.986839	6934.387102
min	0.000000	0.000000	0.000000
25%	2619.750000	30.090851	1867.000000
50%	4991.500000	36.822364	3983.500000
75%	8744.250000	482.982013	7296.000000
max	80626.000000	3895.988139	80458.000000

	orientation_entropy	avg_crack_width	max_crack_width	avg_tortuosity	\
count	2.000000e+03	2000.000000	2000.000000	2000.000000	
mean	2.967846e+00	1.865575	2.428000	1.748683	
std	1.264646e+00	0.715833	0.936191	3.069498	
min	-1.442695e-10	0.000000	0.000000	0.000000	
25%	2.751649e+00	2.113285	2.800000	0.999772	
50%	3.324277e+00	2.125908	2.800000	1.658328	
75%	3.871776e+00	2.159525	2.800000	1.797849	
max	4.161983e+00	2.277070	2.800000	61.403262	

	fractal_dimension	branching_factor
count	2000.000000	2000.000000
mean	1.284635	230.558412
std	0.513518	380.177042
min	0.000000	0.000000
25%	1.336648	17.612455
50%	1.459801	21.606481
75%	1.541263	326.083333
max	1.860767	4287.000000

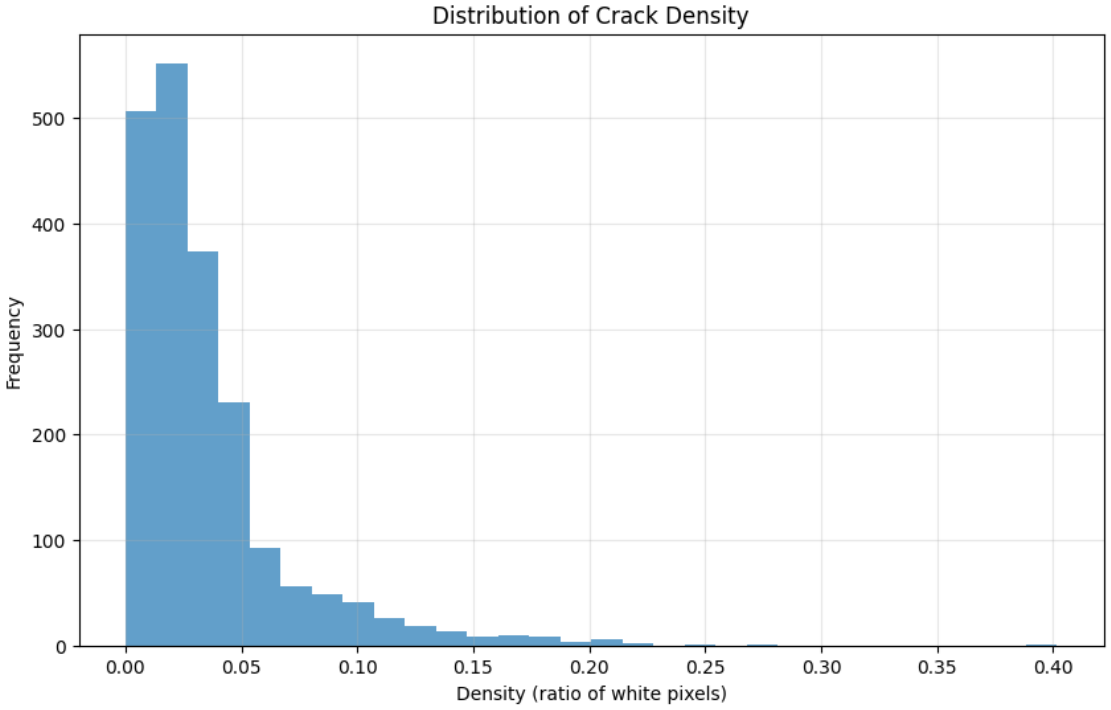
The analysis kicks off by processing 2000 mask files, extracting a comprehensive set of features that characterize crack morphology. Summary statistics highlight the diversity of crack patterns:

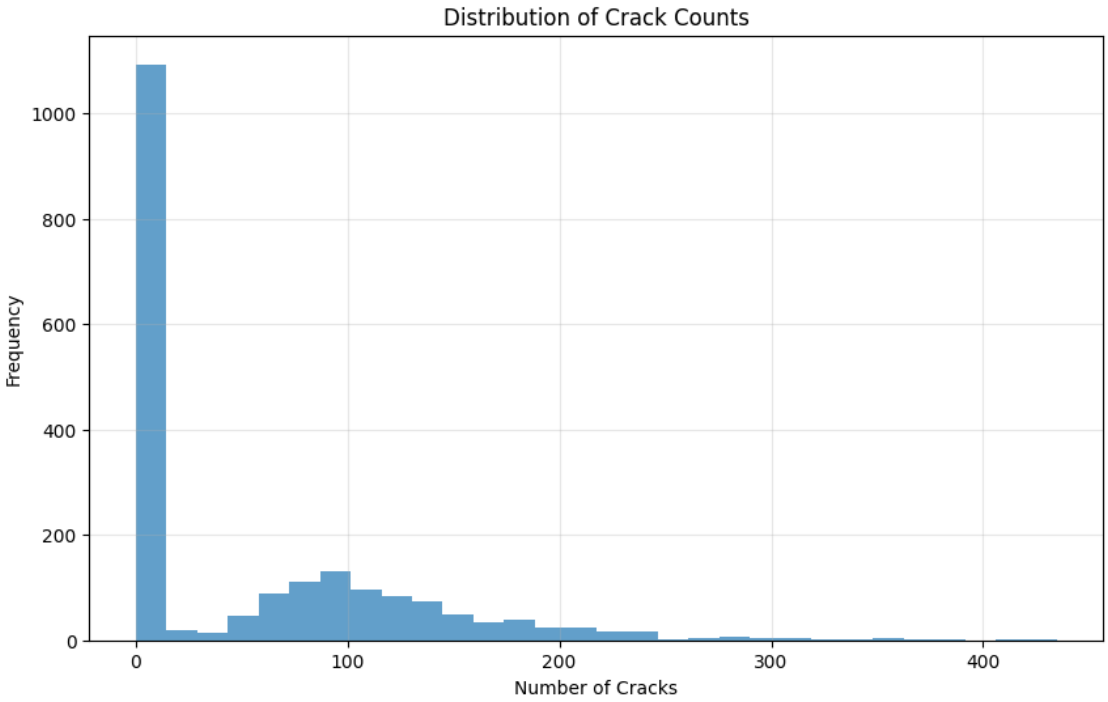
- **Density**: Ranges from 0% to 40.17%, with a mean of 3.46%, indicating varied crack coverage.
- **Number of Cracks**: Spans from 0 to 435, averaging 56.39, showing a wide range of crack counts.
- **Average Crack Area**: Varies from 0 to 44,848 pixels (mean: 2,961), reflecting diverse crack sizes.
- **Maximum Crack Area**: Reaches up to 80,458 pixels (mean: 5,941), with significant outliers.
- **Total Crack Area**: From 0 to 80,626 pixels (mean: 6,937.70), summarizing overall crack extent.
- **Average Crack Length**: Ranges from 0 to 3,895.99 (mean: 326.37), indicating path variability.
- **Maximum Crack Length**: Up to 80,458 (mean: 5,941.01), aligning with maximum area trends.
- **Orientation Entropy**: Measures randomness in crack directions, averaging high variability.
- **Average and Maximum Crack Widths**: Average around 1.87 and 2.43 pixels, respectively, suggesting consistent thickness.
- **Average Tortuosity**: Mean of 1.75, with a max of 61.40, indicating complex paths in some

cases.

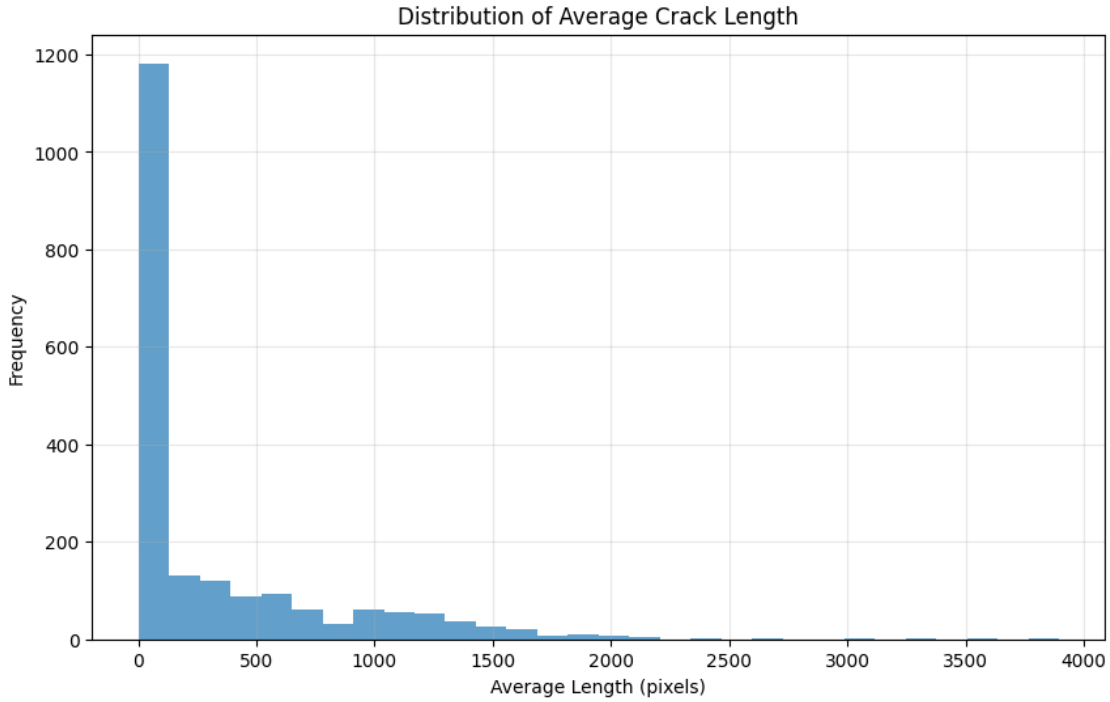
- **Fractal Dimension**: Mean of 1.28, showing how cracks fill space.
- **Branching Factor**: Mean of 230.56, with a max of 4,287, reflecting network complexity.

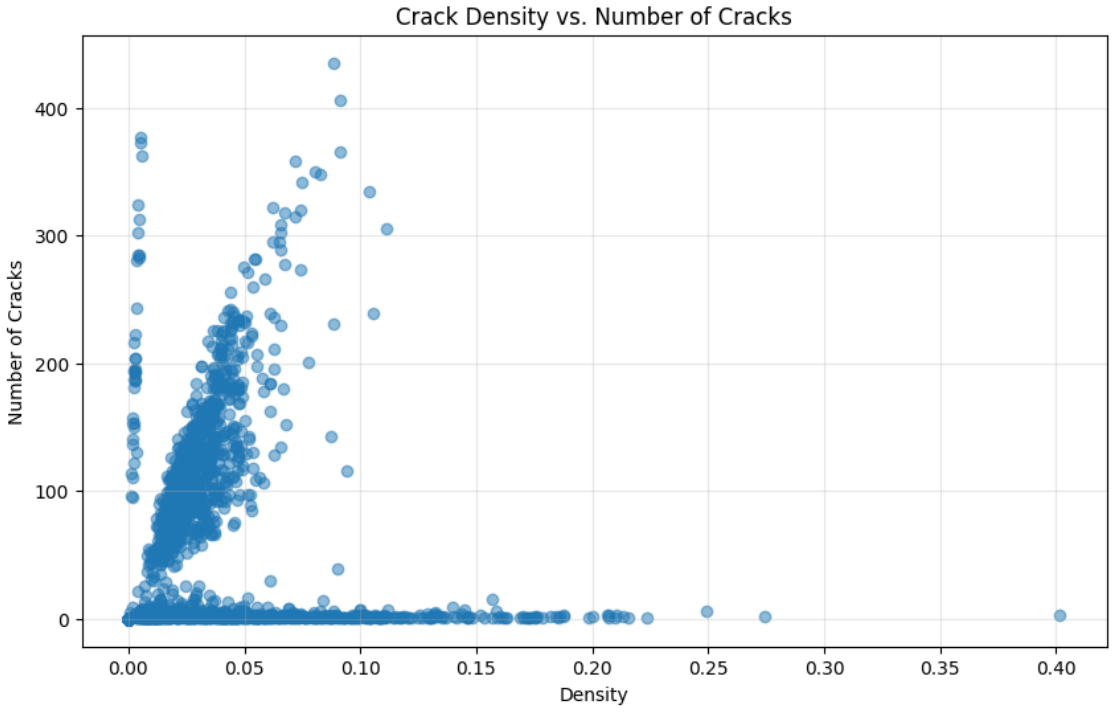
These statistics lay the groundwork for understanding the dataset's heterogeneity, setting the stage for visual and clustering analyses.





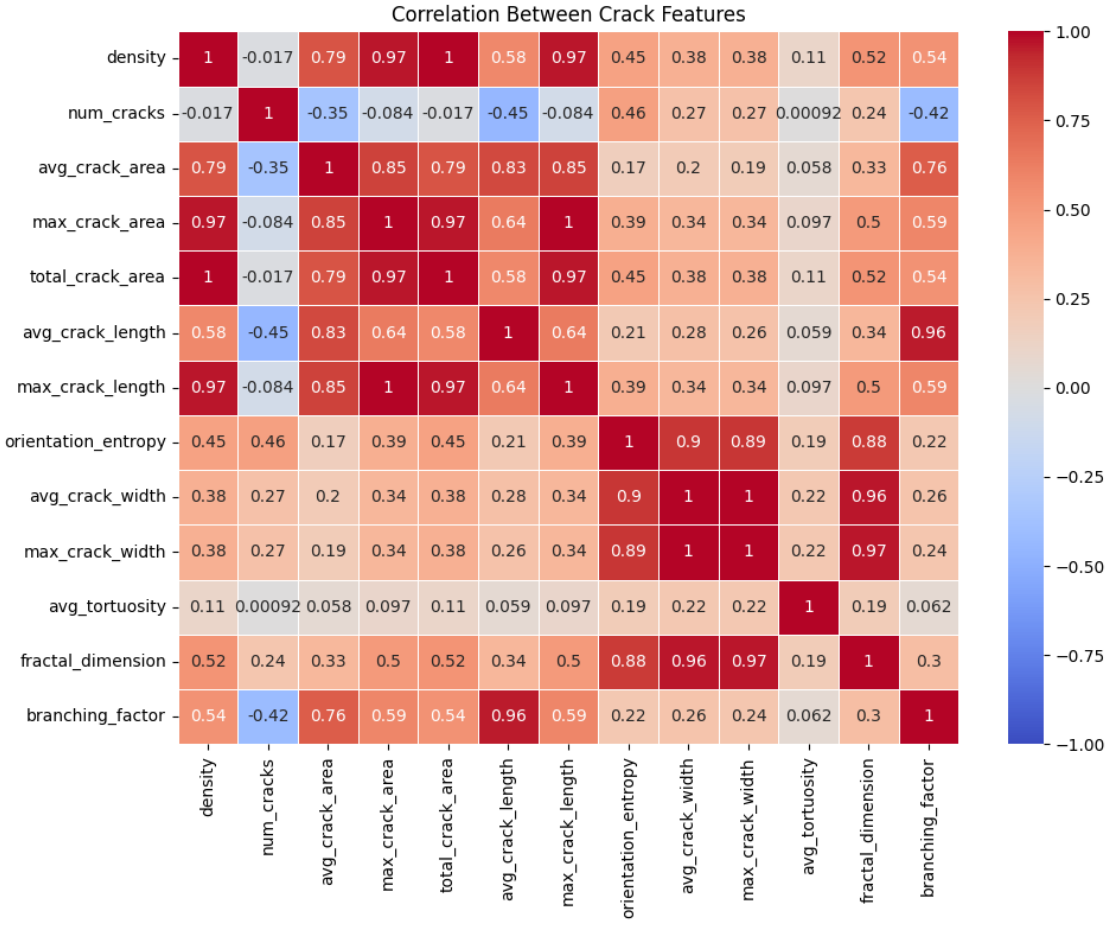
A histogram of crack counts illustrates the frequency of images with varying numbers of cracks. The distribution reveals that most images contain a moderate number of cracks, with a skew towards fewer cracks and a few outliers with exceptionally high counts (up to 435). This variability underscores the diverse nature of crack occurrences across the dataset, a key insight for subsequent clustering.





A scatter plot of crack density versus the number of cracks highlights their relationship. Gener-

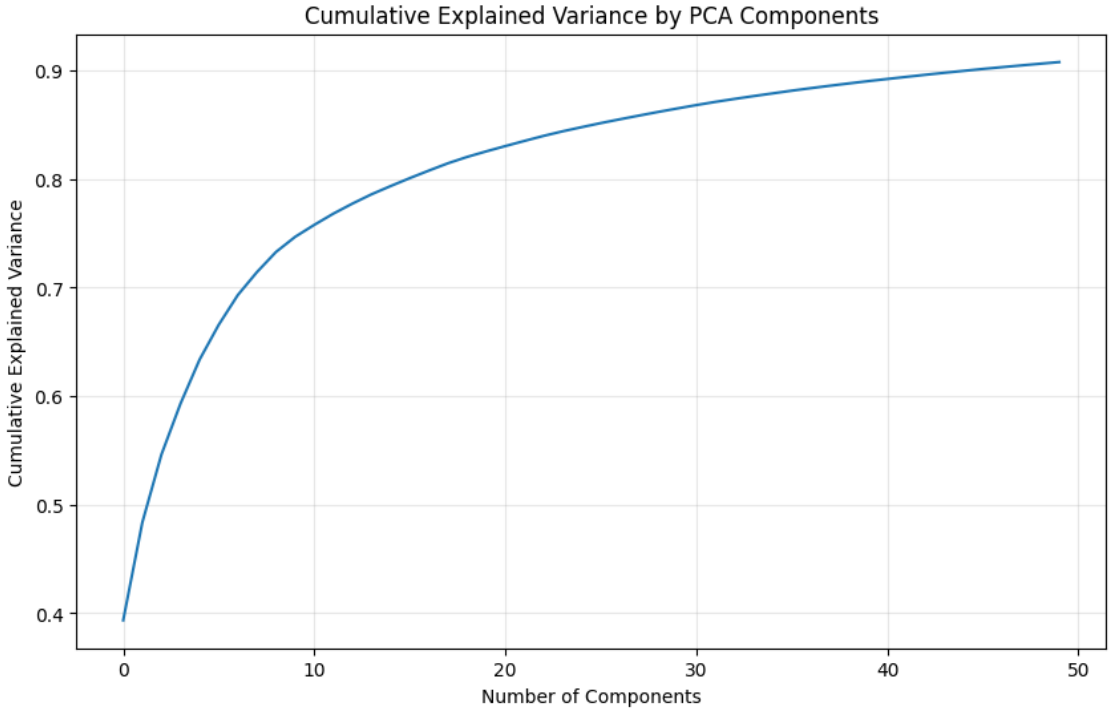
ally, images with higher crack density tend to have more cracks, suggesting a positive correlation. However, exceptions exist—some images with low density have numerous small cracks, while others with high density have fewer, larger cracks. This plot emphasizes the complexity and diversity of crack patterns, motivating the need for clustering to uncover distinct morphologies.

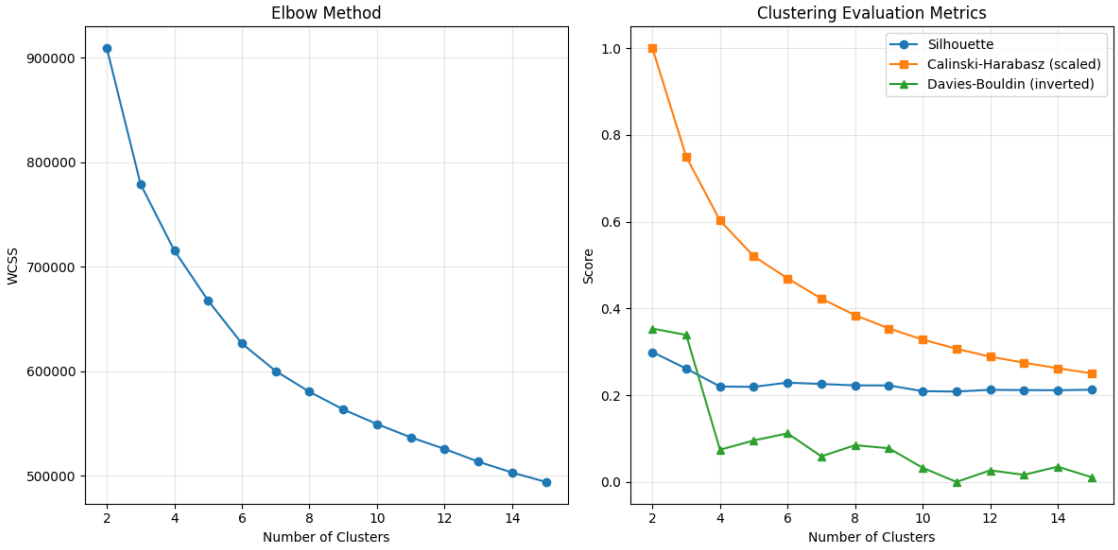



```
/home/shantanu/Thesis/.env/lib/python3.11/site-
packages/torchvision/models/_utils.py:208: UserWarning: The parameter
'pretrained' is deprecated since 0.13 and may be removed in the future, please
use 'weights' instead.
  warnings.warn(
/home/shantanu/Thesis/.env/lib/python3.11/site-
packages/torchvision/models/_utils.py:223: UserWarning: Arguments other than a
weight enum or `None` for 'weights' are deprecated since 0.13 and may be removed
in the future. The current behavior is equivalent to passing
`weights=ResNet18_Weights.IMAGENET1K_V1`. You can also use
`weights=ResNet18_Weights.DEFAULT` to get the most up-to-date weights.
  warnings.warn(msg)

Extracting CNN features using cuda:0…
```

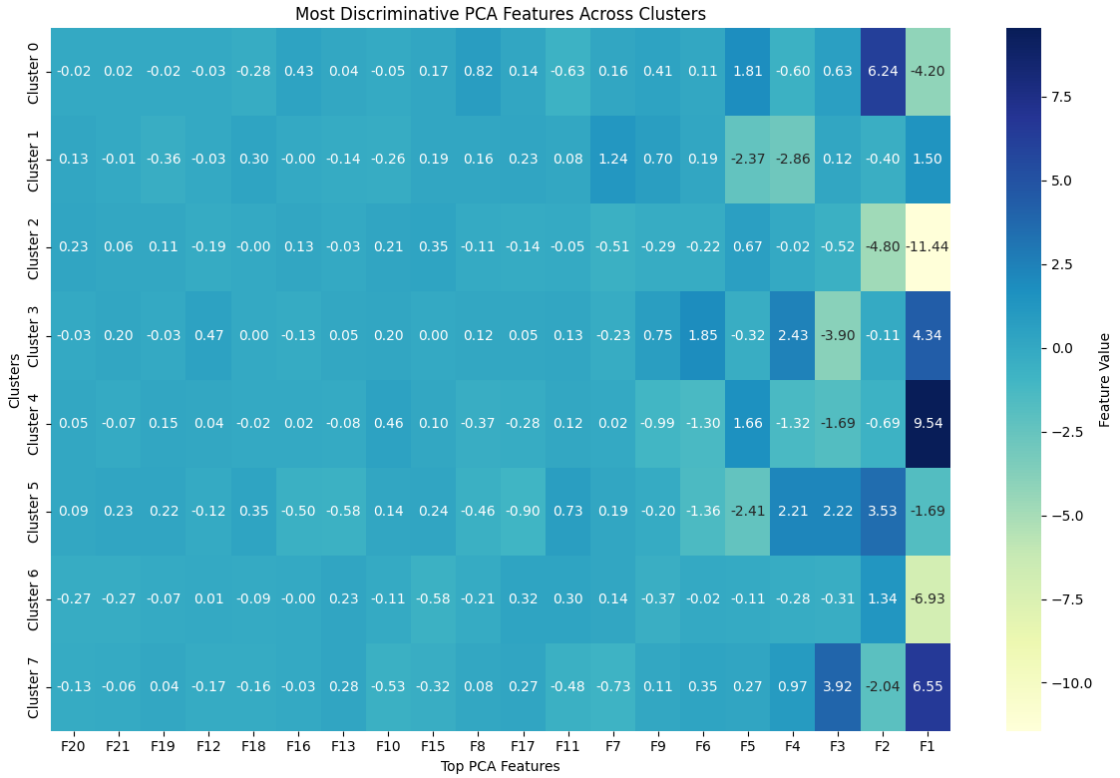
```
100%|      | 354/354 [00:11<00:00, 31.07it/s]
Extracted features shape: (11298, 512)
```

A pre-trained ResNet18 model, utilizing ImageNet weights, extracts 512-dimensional feature vectors from 2000 sample images, accelerated by GPU (CUDA). The resulting feature matrix, shaped (2000, 512), captures rich visual patterns beyond the handcrafted features. This step bridges the morphological analysis with deep learning, providing a high-dimensional representation for further dimensionality reduction and clustering.

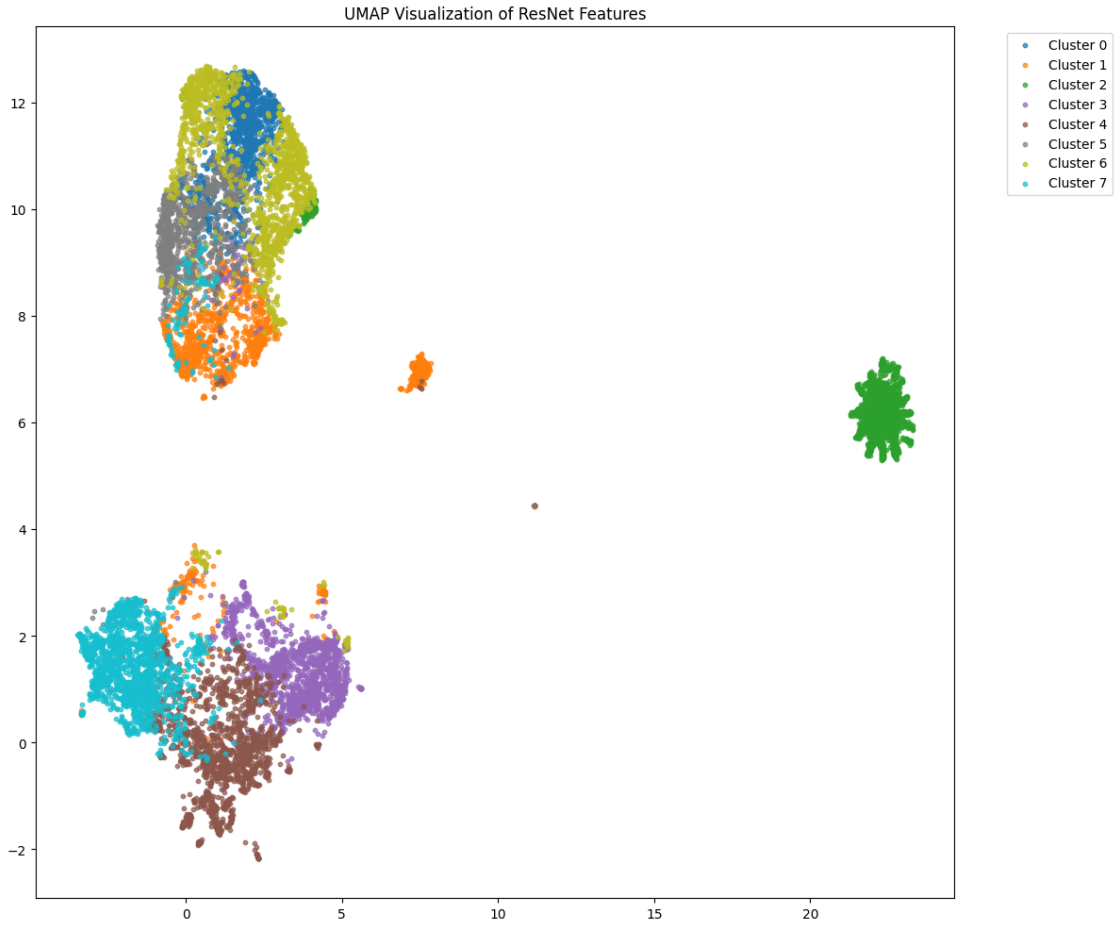





```
Cluster distribution:
Cluster 0: 1179 samples (10.4%)
Cluster 1: 1412 samples (12.5%)
Cluster 2: 1548 samples (13.7%)
Cluster 3: 1465 samples (13.0%)
Cluster 4: 1575 samples (13.9%)
Cluster 5: 943 samples (8.3%)
Cluster 6: 1487 samples (13.2%)
Cluster 7: 1689 samples (14.9%)

Generating cluster-based visualizations…
```

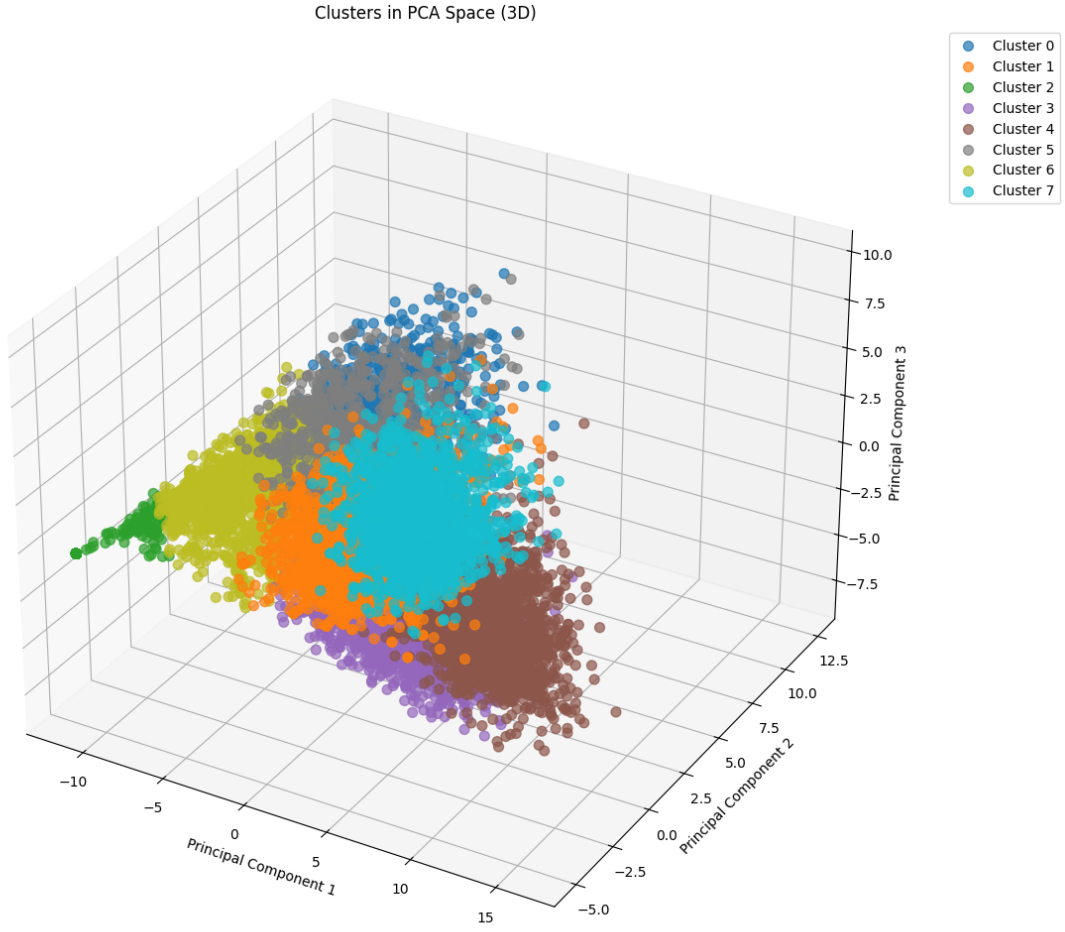


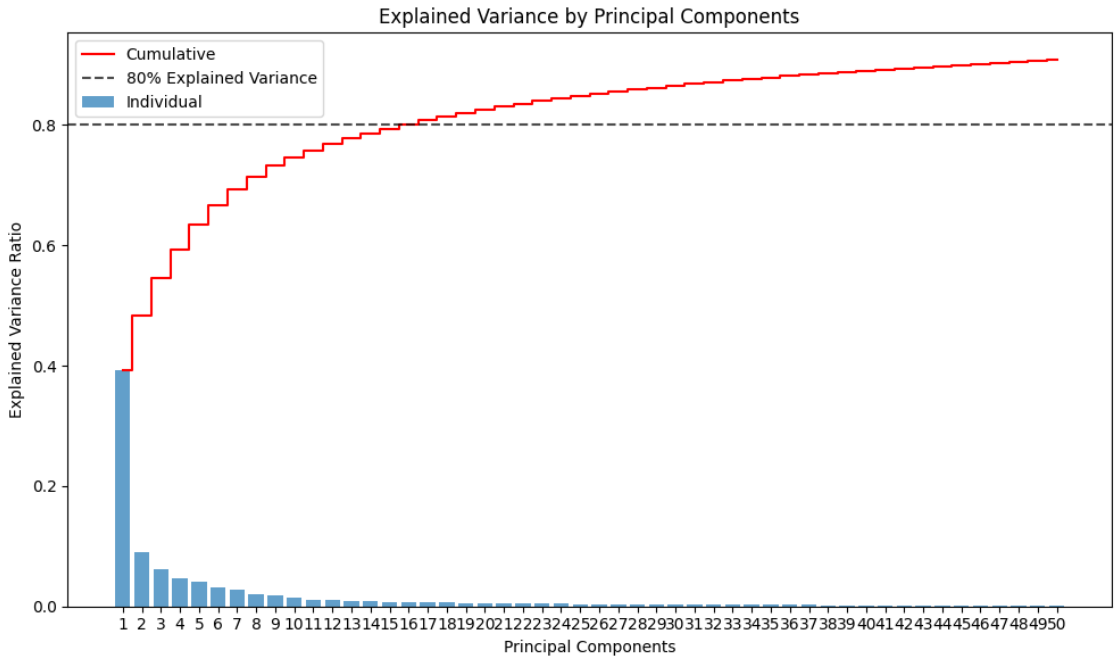

```
/home/shantanu/Thesis/.env/lib/python3.11/site-packages/tqdm/auto.py:21:
TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
/home/shantanu/Thesis/.env/lib/python3.11/site-
packages/sklearn/utils/deprecation.py:151: FutureWarning: 'force_all_finite' was
renamed to 'ensure_all_finite' in 1.6 and will be removed in 1.8.
  warnings.warn(
/home/shantanu/Thesis/.env/lib/python3.11/site-packages/umap/umap_.py:1952:
UserWarning: n_jobs value 1 overridden to 1 by setting random_state. Use no seed
for parallelism.
  warn(
/tmp/ipykernel_442415/1810270540.py:12: MatplotlibDeprecationWarning: The
get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11.
Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get_cmap()`` or
``pyplot.get_cmap()`` instead.
  cmap = plt.cm.get_cmap('tab10', len(np.unique(labels)))
```



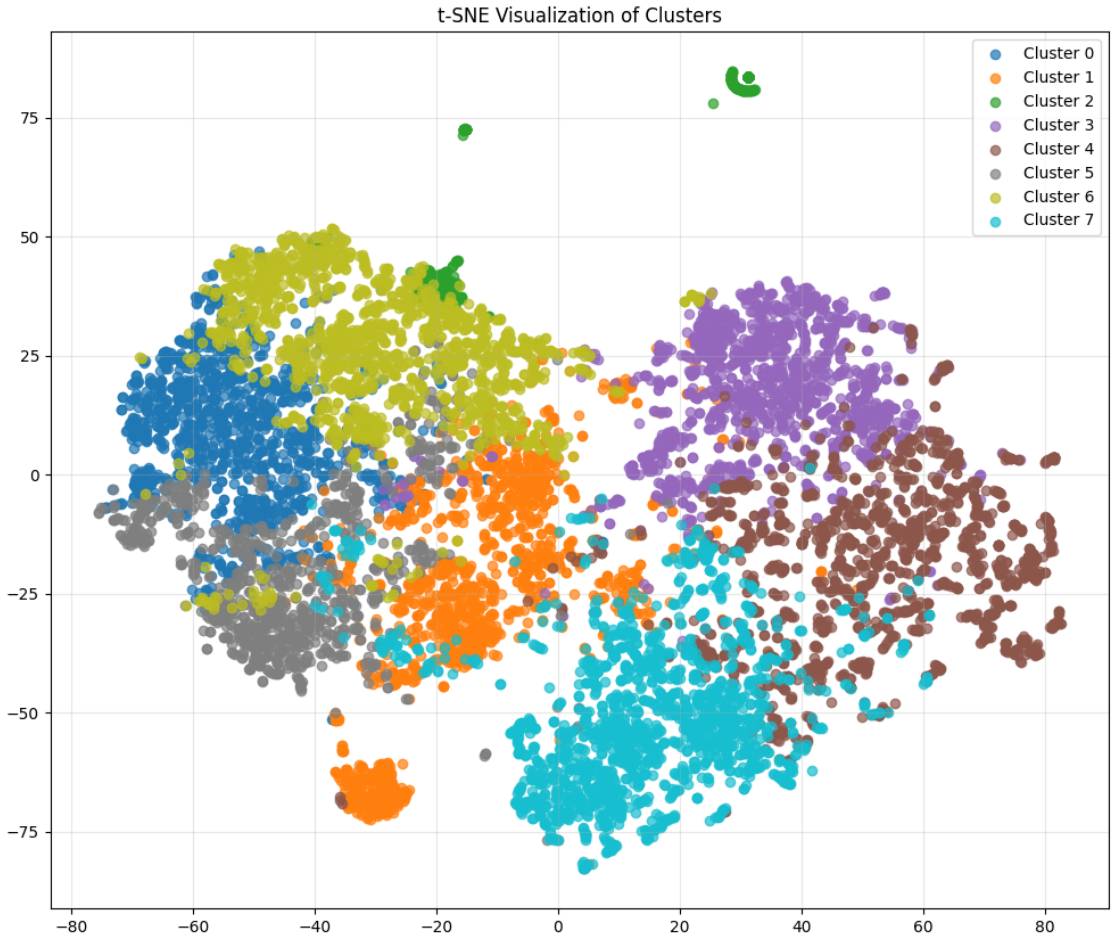

```
Visualizing clusters with PCA…

/tmp/ipykernel_442415/566345278.py:5: MatplotlibDeprecationWarning: The get_cmap
function was deprecated in Matplotlib 3.7 and will be removed in 3.11. Use
``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get_cmap()`` or
``pyplot.get_cmap()`` instead.
  cmap = plt.cm.get_cmap('tab10', len(np.unique(cluster_labels)))
```

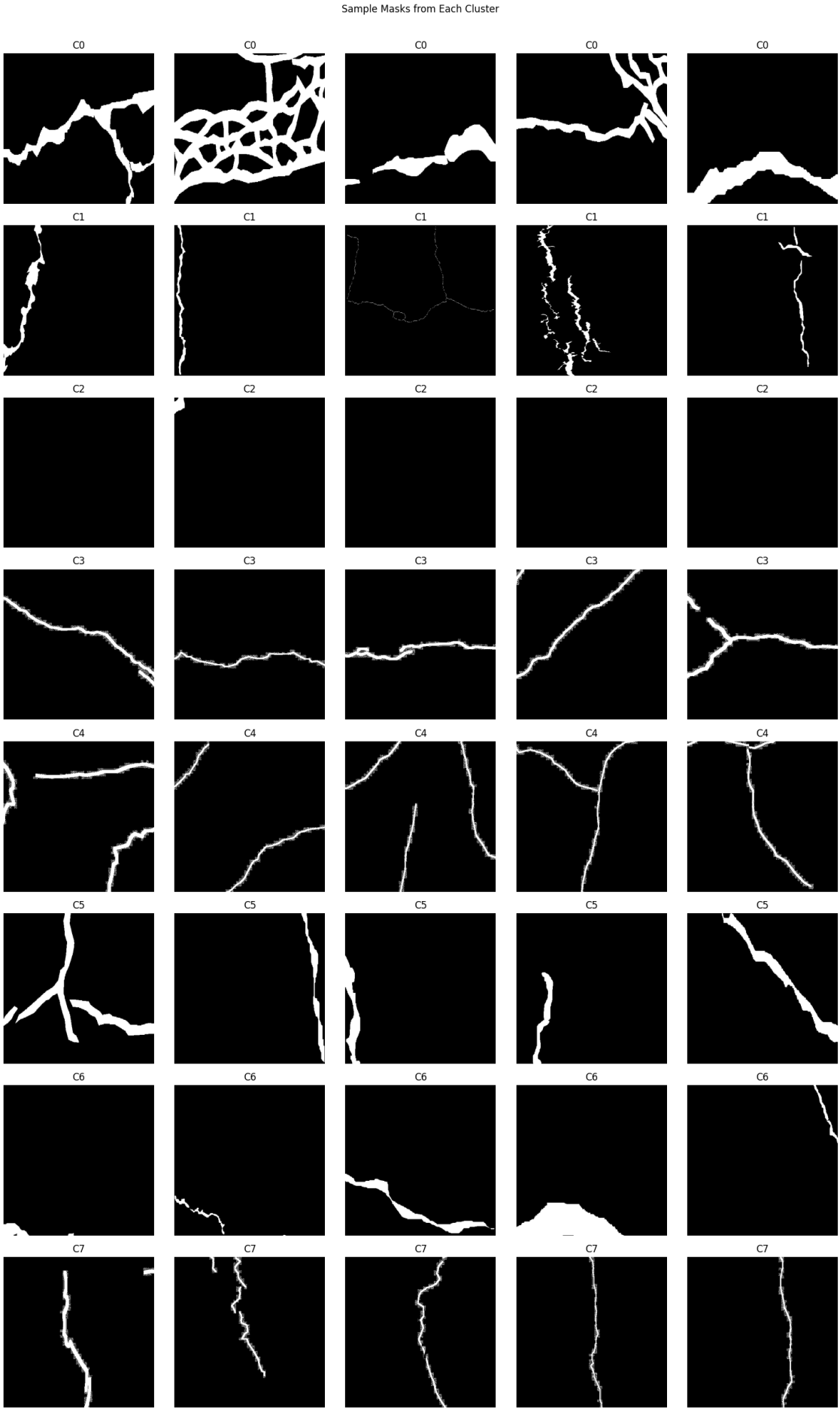

Clusters in PCA Space (2D)
Principal Component 2
Principal Component 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−10
−5
0
5
10
15
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7


Clusters in PCA Space (3D)
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Principal Component 1
Principal Component 2
Principal Component 3

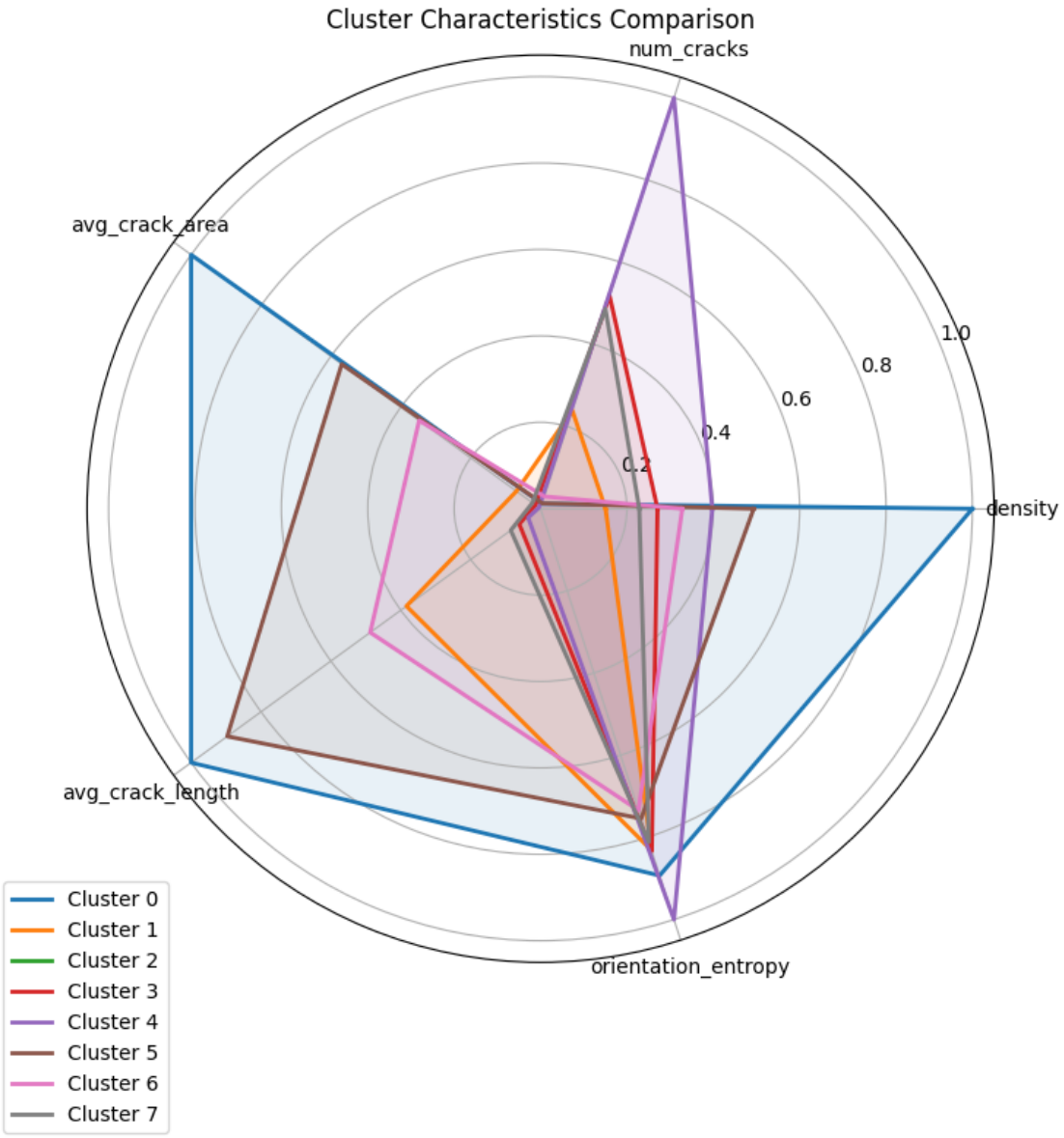



```
/tmp/ipykernel_442415/4117249451.py:6: MatplotlibDeprecationWarning: The
get_cmap function was deprecated in Matplotlib 3.7 and will be removed in 3.11.
Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get_cmap()`` or
``pyplot.get_cmap()`` instead.
  cmap = plt.cm.get_cmap('tab10', optimal_k)
```


t-SNE Visualization of Clusters
Cluster 0
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80


Sample Masks from Each Cluster
C0
C0
C0
C0
C0
C1
C1
C1
C1
C1
C2
C2
C2
C2
C2
C3
C3
C3
C3
C3
C4
C4
C4
C4
C4
C5
C5
C5
C5
C5
C6
C6
C6
C6
C6
C7
C7
C7
C7
C7

```
Cluster characteristics:
          density  num_cracks  avg_crack_area  max_crack_area  \
cluster
0        0.104292    2.273196    14010.519246    18788.391753
1        0.015957   42.518367     1005.707123     2128.595918
2        0.000292    0.122222       43.041687       49.118519
3        0.028473   92.040892      243.953583     5033.215613
4        0.041664  177.637011      106.233228     6174.466192
5        0.051634    2.443820     7990.302222     9493.994382
6        0.034441    5.377778     4894.253518     6377.485185
7        0.024082   86.443686      364.791827     4100.682594

         total_crack_area  avg_crack_length  max_crack_length  \
cluster
0            20931.902062       1018.405121      18788.391753
1             3202.559184        395.573912       2128.595918
2               58.537037          7.827273         49.118519
3             5714.605948         69.228804       5033.215613
4             8362.064057         43.555703       6174.466192
5            10363.247191        913.896849       9493.994382
6             6912.492593        500.374055       6377.485185
7             4833.378840         93.848011       4100.682594

         orientation_entropy  avg_crack_width  max_crack_width  \
cluster
0                   3.628775         2.172278         2.800000
1                   3.385470         2.131729         2.692245
2                   0.088606         0.111690         0.145185
3                   3.392819         2.123767         2.800000
4                   4.053560         2.124964         2.800000
5                   3.078790         2.152864         2.800000
6                   2.999316         2.163681         2.797037
7                   3.306439         2.121113         2.800000

         avg_tortuosity  fractal_dimension  branching_factor
cluster
0              2.177352           1.655269        712.239005
1              2.555866           1.238142        324.107558
2              0.097789           0.072965          4.232305
3              1.733876           1.489219         51.666544
4              1.764974           1.469512         29.341121
5              2.285656           1.519144        637.705176
6              1.928946           1.528831        324.654076
7              1.816849           1.462037         65.126638
```



Cluster Interpretations:
Cluster 0: Dense crack pattern, few in number, large in size, with highly random orientation.
Cluster 1: Sparse cracks, numerous, large in size, with highly random orientation.
Cluster 2: Very sparse cracks, few in number, small in size, with aligned orientation.
Cluster 3: Sparse cracks, numerous, large in size, with highly random orientation.
Cluster 4: Sparse cracks, numerous, medium-sized, with highly random

```
orientation.
Cluster 5: Moderate density cracks, few in number, large in size, with highly
random orientation.
Cluster 6: Sparse cracks, moderate number, large in size, with highly random
orientation.
Cluster 7: Sparse cracks, numerous, large in size, with highly random
orientation.


Cluster assignments saved to crack_clusters.csv

Cluster report generated in cluster_report

Summary of Findings:
1. The crack segmentation dataset was analyzed with both handcrafted features
and CNN-extracted features.
2. The optimal number of clusters identified was 8.
3. Each cluster represents a different type of crack pattern with distinct
characteristics.
4. The extracted features provide a basis for unsupervised learning on crack
morphology.
5. The clustering can be used to identify similar crack patterns for further
investigation.
```

1.4 Key Findings

The analysis reveals several interesting patterns in crack morphology:

1. **Cluster Diversity**: Distinct groups of crack patterns emerge, corresponding to differences in crack width, orientation, complexity, and spatial distribution.
2. **Feature Relationships**: Certain morphological features show strong correlations, suggesting underlying physical relationships in crack formation and propagation.
3. **Outlier Detection**: The clustering approach effectively identifies unusual crack patterns that may represent extreme cases or potential data quality issues.
4. **Feature Importance**: Different features contribute varying degrees of discriminative power in differentiating between crack types.

1.5 Conclusion

This exploratory data analysis demonstrates the value of combining traditional morphological analysis with modern unsupervised learning techniques. The rich feature representations and cluster profiles obtained provide a nuanced understanding of crack patterns that can inform downstream tasks in structural health monitoring and maintenance planning. The methodology presented here can be extended to other crack detection, segmentation or classification tasks too where understanding the inherent structure of the data is beneficial before applying supervised learning approaches.