Segmenting Cracks using classification algorithms on the low dimensional embedding of Data

Avik Kumar Das
Department of Civil Engineering
HKUST
Clear Water Bay, Hong Kong
akdas@connect.ust.hk

Neel Kanth Kundu
Department of ECE
HKUST
Clear Water Bay, Hong Kong
nkkundu@connect.ust.hk

1. Introduction:

In recent years, many vision-based methods have been proposed to assess the visual condition of reinforced concrete bridges, precast tunnel, underground concrete pipes, and asphalt pavement by exploiting the image processing techniques that have been developed in the computer vision. A detailed review of these vision-based methodologies is in the reference [1], [2]. With respect of damage detection for civil infrastructures (most popular) vision-based methodologies are classified into following segments a) Image processing methodologies (M-IP) [3] b) machine learning-based methodologies (M-ML) [1] b) methods utilizing deep convolutional neural networks (M-DCCN) [4], [5] . M-IP includes methods such as thresholding, Canny's edge detection. Other popular methods include percolation models and clustering. These methods are simple, do not require annotated data set and prior training if handcrafted features are appropriately designed. In M-IP such handcrafted features are replaced by machine-learned features then, concrete defects are identified using classification algorithms, such as support vector machine, naïve Bayesian classifier, and random forest in a supervised setting. That is, it often requires annotated datasets. For an annotated dataset, generally speaking M-DCCN makes more accurate inference as compared to M-ML [1], [4]-[6]. Therefore, M-DCCN is becoming popular among various researchers. Researchers have built in-house M-DCCNs by iteratively combining various computational units of convolutional neural network (CNN)[6] or utilized transfer learning to repurpose a learned representation [3] for their specific dataset to identify (concrete) defects. Expectedly, M-DCCNs make exceptionally good inference when datasets are similar and 'ground truth' is known. As of now, publicly large volumes of labeled datasets of all types of damages in concrete are normally not available. This could be due to labeling such 'engineering' datasets requires expert domain knowledge and a tremendous amount of time. This remains a challenge for the wider acceptability of M-DCCNs. To solve this problem a new methodology is explored which is based on the (linear and non-linear) embedding of the dataset followed by a clustering technique to segregate image dataset into relevant classes. We anticipate that the image dataset contains redundant information due to correlation among the neighboring pixels and the data can be represented in a much smaller dimension by using proper dimensionality reduction techniques. Since this task aims to classify the images as having a crack or not which being a binary classification problem we want to reduce the

dimensionality of the dataset and then use appropriate classification algorithms on the transformed data which can give faster inference and also reduce the storage cost of the large dimensional dataset. In this project, we studied various linear and non-linear dimensionality reduction techniques like PCA, MDS, ISOMAP, and tSNE.

2. Data Collection

2.1. Data Collection and Pre-Processing

Images of surface cracks are collected during various experimental studies at materials and structures laboratory at HKUST. Damage formation during standard tension and bending tests were photographed using smartphones and DSLR cameras. In this study, to achieve a good compromise between computational cost and accuracy of the detection results each sample is a 3-channel (RGB) 227×227 pixel image patch. These images were converted into greyscale through thresholding and then flattened. A total of 10000 image patches collected from various experimental studies were used for classification. Hereby, this dataset is referred to as the image dataset (I_{nxd}). Here, n is the number of samples i.e. 10000 and d is the dimension of the flattened image i.e. 227x227 (51529). The dataset consists of 5000 images labeled as damaged (D) and rest (5000) images labeled as non-damaged (ND). These image-level labels are assigned through manual inspection of each image. Some examples of D and ND images are shown in Figure 1. For concrete infrastructures, often failure (damage) is preceded by the formation of cracks therefore, almost all D images have cracks.

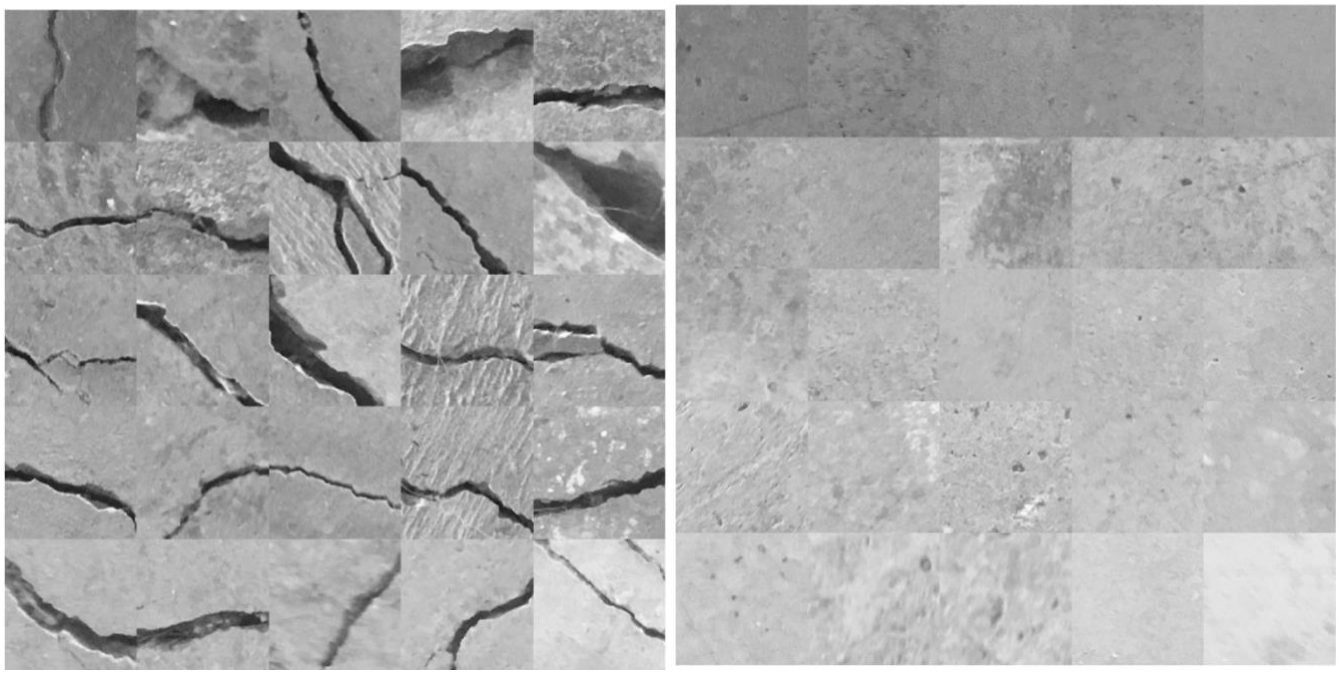

Figure 1 Some images of our dataset with annotation a) D(Damaged) b) ND (Not Damaged)

3. Methodology

3.1 Dimensionality Reduction: We use the following dimensionality reduction techniques on our real crack dataset to find the 2D embedding of our data for visualization and later for classification.

a) PCA: PCA is a classical full spectral technique for dimensionality reduction which transforms the data by embedding the data into a linear subspace of lower dimension. PCA embedding can be found by taking the leading eigenvectors of the covariance matrix of the leading right singular vectors of the data matrix. Although PCA has been used for a long time it has some drawbacks like the dimensionality of the covariance matrix can be huge for large dimensional data and hence the eigenvector computation can be difficult. Moreover, PCA tries to preserve the large pairwise distance instead of focusing on the smaller pairwise distances among the data points. Let the data matrix X be of dimension $n \times p$ where n is the number of data samples and p is dimension of each data point. PCA can be carried out, either by using singular value decomposition (SVD) of the data matrix X or by using eigenvalue decomposition of the covariance matrix $\Sigma = \frac{1}{n}(X^T X)$. Here we use SVD to reduce the dimension of our data points. SVD of $X = U\Lambda V^T$, where U and V are orthonormal matrices of dimension $n \times p$ and $p \times p$ called the left and right singular matrices respectively. Λ is a diagonal matrix of dimension $p \times p$. The dimension of the data matrix can be reduced to $k < p$, by taking the first k columns of U and the upper-left $k \times k$ part of Λ. $X_k = U_k \Lambda_k$ is the transformed data matrix of dimension $n \times k$.

b) MDS: MDS is also a classical algorithm that works by first finding the Euclidean distances between the data points and then finds the pairwise Euclidean distance matrix. MDS tries to find a low dimensional linear embedding of the dataset, which preserves the Euclidean distances of the data points in the original space. The low dimensional embedding is obtained by the eigen decomposition of the double-centered pairwise squared Euclidean distance matrix as detailed below. For a given data matrix $X = [x_1, x_2, \ldots x_n]^{p \times n}$, first, the squared distance matrix D, is calculated with$D_{ij} = \left|\left|x_i - x_j\right|\right|^2$ then the matrix $B = -\frac{1}{2}HDH^T$, is computed where H is the householder matrix. Finally find the eigen decomposition of $B = U\Lambda U^T$ is calculated with the eigenvalues being sorted in decreasing order. The top k eigenvalues and the corresponding eigenvectors are used to find the k dimensoopnal embedding for the data matrix as $X_k = U_k \Lambda_k^{1/2}$.

c) ISOMAP: Classical MDS and PCA only take into account the pairwise Euclidean distance and not the neighboring distribution of the data points. Many real data sets may lie on a curved manifold where the Euclidean distance is not the right metric which gives the distance between the data points. The distance should be calculated along the manifold which is known as the Geodesic distance. ISOMAP finds the geodesic distance by approximating the geodesic distance with Euclidean distance for nearby data points and then using the graph shortest path algorithm to find the geodesic distance between the other data points. In contrast to classical MDS, in ISOMAP the geodesic distance between the data points is used to construct the squared distance matrix whose eigendecomposition can be used to find the ISOMAP embedding. In this sense ISOMAP is a non-linear, manifold dimensionality reduction technique that tries to preserve the geodesic distance between the data points when finding the low dimensional embedding of the dataset [7]

Formatting... please wait. Here first the neighborhood graph $G = (V, E, d_{ij})$ is constricted where $V = \{x_1, x_2, \ldots x_n\}$ is the vertex set containing the data points , $E = \{(i,j)$: if j is neighbour of i$\}$ is the edge set and d_{ij} is the shortest patch distance between the data point x_i, x_j. d_{ij} is equal to the Euclidean distance if i, j are neighbors otherwise the distance is calculated as follows:

$$d_{ij} = \min_{P=(x_i,..x_j)} (\left|\left|x_i - x_{t_1}\right|\right| + \left|\left|x_i - x_{t_2}\right|\right| \ldots + ||x_{t_{k-1}} - x_{t_1}||).$$

Now the ISOMAP embedding is found using the same method as that of MDS by using this distance matrix.

d) tSNE: t-distributed stochastic neighbor embedding (tSNE) is a recent non-linear dimensionality reduction technique that has become increasingly popular for its success in 2D data visualization which has shown promising results in separating the different classes in the datasets in low dimensional embedding effectively. Unlike PCA and MDS, tSNE is a non-linear dimensionality reduction technique that uses the local relationships between the nearby data points. It uses Gaussian distribution to specify the distribution of the data points in the high dimensional space and it tries to preserve the distribution of the points in the low dimensional space.
For n data points $x_1, x_2, .. x_n$, the probability $p_{j|i}$, which denotes the conditional probability of x_j being a neighbor of x_i in the high dimensional space is calculated as follows:

$$p_{j|i} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)}$$

Now the probabilityp_{ij}, which is proportional to the similarity between x_i, x_j is defined as

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

The number of effective neighbors of a datapoint which is called the perplexity is an important hyperparameter which depends on the bandwidth hyperparameterσ_i^2. The authors of the original paper [3] suggested that perplexity should be set between 5 and 50. tSNE solved the crowding problem in the low dimensional space present in the previous stochastic neighbor embedding technique by using student t-distribution to recreate the probability distribution in the low dimensional space [8] . In the low dimensional space, tSNE finds $y_1, y_2, \ldots y_n$ such that the similarity measure $q_{ij}$is given by

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq i} \left(1 + \|y_i - y_j\|^2\right)^{-1}}$$

tSNE minimizes the KL divergence between the probability distributions p_{ij} and q_{ij}. Stochastic Gradient descent (SGD) based method is used to minimize the non-convex cost function given by

$$KL(P||Q) = \sum_{i \neq j} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right)$$

In our experiments we found that perplexity of 30 gives good visualization results.

3.2 Classification Algorithms: After reducing the dimensionality of our dataset from $D = 51529$ to $d = 2$, we use various classification algorithms to classify the data points as having a crack (1) or not having a crack (0).

a) k-NN: k-nearest neighbor is a classical simple machine learning model that finds the k-nearest neighbors of a test data point from the training dataset by computing the Euclidean distance between them and then it predicts the label by using the maximum vote of the labels of the nearest neighbors. K-NN does not learn any features from the training dataset and thus it requires huge storage for the complete dataset and then it also has a slow inference time.

b) Decision Tree: Decision tree is a non-parametric classifier that learns the classification rule by learning simple decision rules from the data features, hence it is a non-linear classifier. It can be easily interpreted by visualizing the tree formed from the different decision rules.

c) Logistic Regression: This is a linear classifier model where the probabilities of the output classes are modeled using a logistic function. The logistic regression loss function is convex which can be easily solved by using different algorithms like GD and various extensions like stochastic gradient descent (SGD), SAGA, etc.

d) Ridge Classifier: This classifier first converts the classification targets as ±1 and then uses ridge regression to find the optimum parameters of the linear classifier. Ridge regression minimizes the least-squares objective plus the l_2 regularized loss. The predicted class corresponds to the sign of the linear regressor's prediction.

e) SVM: Support vector machine is a very successful and popular classification algorithm and provided the best performance until last decade until the emergence of deep learning-based classifiers. SVM finds the maximum margin classifier where the support vectors are needed during the prediction task on the test dataset. SVM can easily handle those datasets which are not linearly separable by using kernel trick where the features can be easily transformed to a higher dimension and non-linear margins or decision boundary can be learned from the dataset in a supervised fashion. In our experiments we found that the 'rbf' (radial basis function) kernel provided the best classification performance since this kernel function can learn the non-linear decision boundary of our dataset efficiently.

f) DNN: Since deep learning-based classifiers have shown exceptional performance on classification tasks, in this project we use a 2 hidden layer feedforward neural network (FNN) as a classifier. The first hidden layer has 128 neurons, with the ReLu activation function and a dropout layer with a dropout probability of 0.3 to prevent overfitting of the training dataset. The 2nd hidden layer has 32 neurons with a ReLu activation function, and the final output layer has a single neuron with sigmoid activation which predicts the probability between 0 and 1. If the prediction is >0.5, it classifies it as class 1, otherwise, it classifies the data as class 0.

4. Results

4.1 Data visualization and Benchmarking of Classification Results

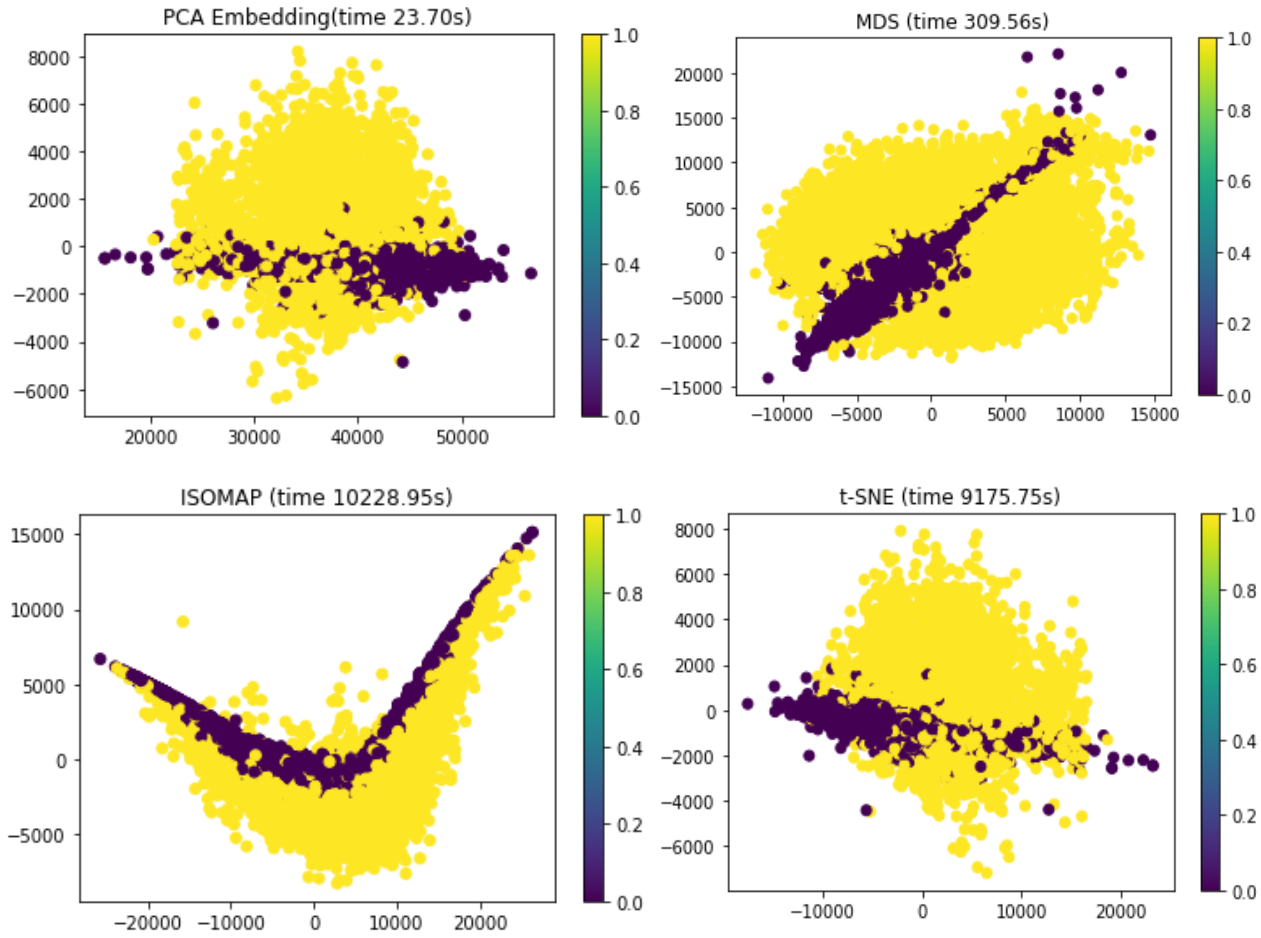

Figure 2 2D Embedding for a) PCA b) MDS c) ISOMAP d) t-SNE

Figure 2 shows the 2D embedding of the dataset using PCA, MDS, ISOMAP, and t-SNE dimensionality reduction techniques respectively. In these figures, the D annotated images are showcased in purple while ND annotated ones are in yellow. It was incident from the results that in all cases D images are clustered together as a streamline while ND images are more distributed within the embedding. This is because damage (crack) formation in concrete structure is diverse does not have is not structured and oriented. This is expected as the dataset was collected from a non-controlled experimental setup. However, as compared to other techniques T-SNE embedding is most computationally expensive. Unfortunately, none of these techniques could (qualitatively and) completely segregate these classes of images. To scientifically select the best possible combination of embedding and classifier for this case performance parameters i.e accuracy and F1 score for each combination was computed. This is reported in table 1. It should be noted that the entry of the table shows performance parameters accuracy and precision in (accuracy, precision) format. The result of this study clearly indicate that MDS based 2D embedding is most separable (highest performance) and DNN in general showcase best performance irrespective of the type of the embedding. That is, MDS-DNN and MDS-KNN are the 2 best performing system for this dataset when benchmarked against the inference capability. But, when computational complexity is also included as one of the benchmarking indexes MDS-KNN turns out to be the most attractive solution as, unlike other classification

methods the training of DNN usually takes in the order of 10^3s to complete. Henceforth, experiments are only performed with the MDS-KNN system.

Table 1 Performance Results of different combination of embedding and classifier

Classifier/Techniques	PCA	MDS	ISOMAP	t-SNE
KNN (k=5)	(0.85,0.85)	(0.95,0.95)	(0.92,0.92)	(0.84,0.84)
Decision Tree	(0.81,0.81)	(0.93,0.93)	(0.91,0.91)	(0.80,0.80)
Logistic Regression	(0.8,0.8)	(0.69,0.69)	(0.74,0.74)	(0.80,0.80)
Ridge Classifier	(0.8,0.79)	(0.70,0.69)	(0.73,0.73)	(0.78,0.79)
SVM-rbf	(0.87,0.86)	(0.94, 0.94)	(0.92,0.92)	(0.87,0.85)
DNN	(0.87,0.86)	(0.96,0.96)	(0.93,0.93)	(0.86,0.86)

4.2 Sensitivity Analysis

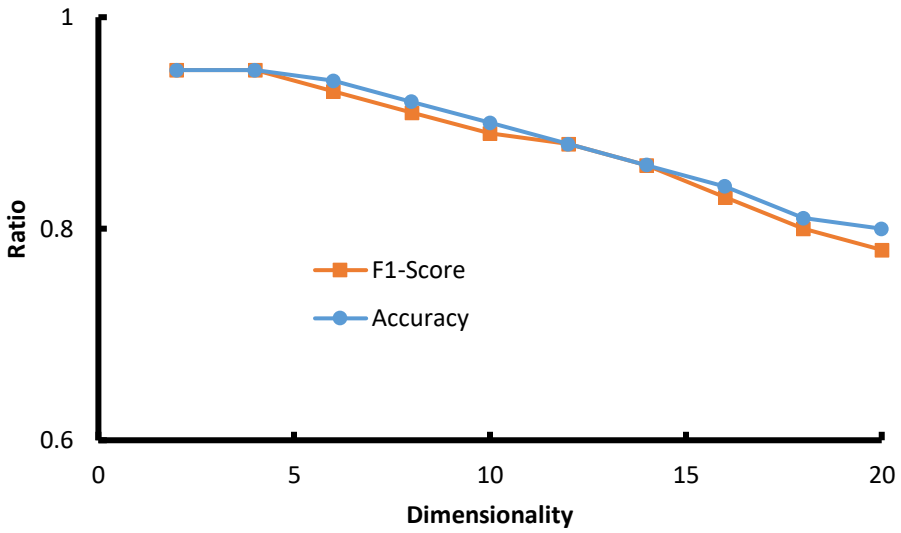

Figure 3 Effect of performance with dimensionality of the embedding

Damage formation is and randomly intractable [9]. As a result, to ensure that a system is resilient in 'D' annotated images; insensitivity to the external hyper-parameters is important. For the MDS-KNN system some of these hyper-parameters include a) dimension of the reduced dataset b) randomization in the order of the dataset. The effect on the performance parameter (accuracy and F1-Score) due to the choice of the embedding dimension is shown in Figure 3. It is observed that with increasing dimension after 4, the accuracy, as well as precision of this system, gradually decreases with increasing dimensionality of the embedding. Although it should be noted that in this study, embedding dimensions up to 20 were considered, however, the indicative results (might) still be valid even when larger embedding dimensions are considered. To understand the effect of randomization of the order of the input dataset 100 sets of Monte-Carlo simulation were performed. The mean accuracy and F1-score for the MDS-KNN system remained consistent at 0.95 and 0.95 respectively. The coefficient of variation (CV) values for both accuracy and precision were ~1% is therefore negligible. This shows that the

present system MDS-KNN is insensitive to the order of the dataset. This is because changing the order only changes the order of the rows in the distance matrix which does affect the spectral parameters (eigenvectors and eigenvalues). Therefore, the quality of the embedding and corresponding performance parameters remain unaffected.

4.3 Drawing Insights with MDS-KNN

Figure 4 shows the result of ordering images with MDS along the direction of the first eigenvector. In this figure 8 indicative randomly selected images for each D and ND annotated classes were selected for better visualization. Although the ordering process was repeated other images however, the qualitative trend for both ND and D images remains consistent with the one presented in Figure 4. Non-damaged segments are qualitatively ordered according to their relative brightness i.e. discoloration (of the concrete surface) as shown in Figure 4a. The discoloration is not natural and caused by many factors such as the quality of the constituent material [10]. In a controlled test where the effects of background light are compensated, such ordering could be associated with the quality of the material used in concrete and health of the concrete specimen. Similarly, the ordering of D images is shown in figure 4b. This order has a (faint) correlation with the amount of damage. It should be noted that amount of damage is referred to as the total area enclosed by the crack. This is because cracks are random and intractable and unique in terms of shape and direction of propagation (see Figures 1 and 4). Therefore, ordering such a physical system might not be possible using the top eigenvector. This could be the physical explanation behind D annotated images that do not show any correlation (alignment) between abscissa and ordinate for most of the types of embedding technique (as shown in Figure 2). These images are relatively widely distributed as compared to ND images.

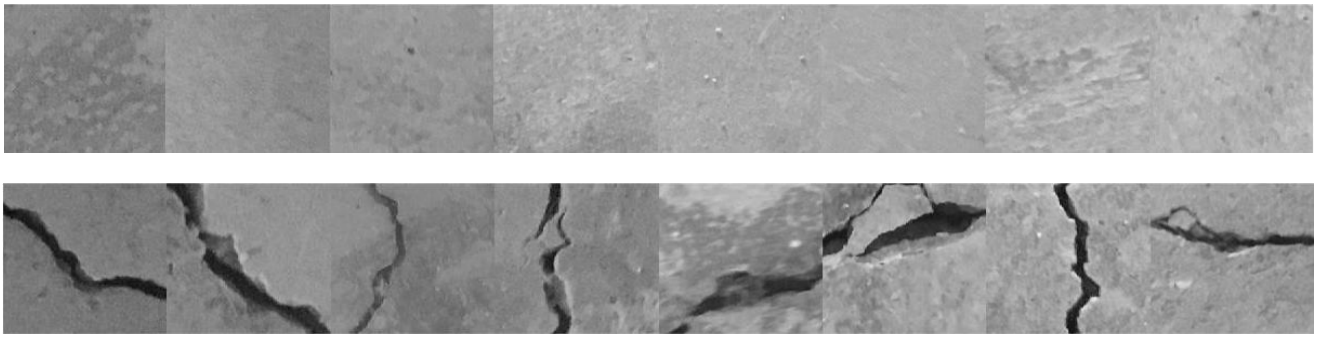

Figure 4 Arrangement of ND and D images using MDS

5. Conclusion and Future Direction

In this study, we applied various dimensionality reduction techniques to segregate damaged and non-damaged images. The following are the major results:

a. MDS+DNN and MDS+KNN showcase the highest accuracy and precision of our dataset. Due to lower computational cost, MDS+KNN is selected as a preferred solution.
b. Performance of MDS+KNN is not sensitive to the ordering of the dataset however, with increasing dimensionality, performance deteriorates.
c. Ordering of D images with MDS shows a faint correlation with the amount of damage whereas for ND reveals discoloration of the concrete surface and in a controlled test quality of the constituent material could be revealed.

Although ideally, we would like to compare all the non-linear dimensionality reduction techniques learned in this course, we have found that due to large dimensionality of our real crack dataset some other non-linear dimensionality reduction techniques like LLE, HLLE and LTSA are not able to handle our dataset and they run out of memory even for a smaller number of nearest neighbors. Hence, it would be an interesting future work where the efficient implementation of LLE, HLLE, and LTSA are investigated which can handle large dimensional real datasets. Recently, an efficient implementation of tSNE has been proposed in the literature [11], which utilizes GPU computation to speed up the baseline tSNE algorithm, similar work from the computer science domain is required to develop GPU accelerated implementation of LLE, HLLE, LTSA will be explored for a larger dataset.

Contribution:

1) Avik Kumar Das: He collected the dataset and put forward the initial project idea and helped in coding the different classification algorithms and perform sensitivity analysis and Ordering of the images using MDS and helped in writing the report.
2) Neel Kanth Kundu: He worked on the technical part of the project, writing the technical details of the different dimensionality reduction techniques and classifiers. He helped in coding the dimensionality reduction and classification of the low dimensional embedding. He helped in data analysis and writing up the report.

Presentation Video Link: https://youtu.be/z2LLNTQGVY8
Github Link: https://github.com/AVKDAS/Crack-Segmentation/tree/master

References

[1] C. Koch et al, "A review on computer vision-based defect detection and condition assessment of concrete and asphalt civil infrastructure," Advanced Engineering Informatics, vol. 29, (2), pp. 196-210, 2015. DOI: https://doi.org/10.1016/j.aei.2015.01.008.

[2] A. Mohan and S. Poobal, "Crack detection using image processing: A critical review and analysis," Alexandria Engineering Journal, vol. 57, (2), pp. 787-798, 2018. DOI: https://doi.org/10.1016/j.aej.2017.01.020.

[3] C. Lu, J. Yu, and C. K. Y. Leung, "An improved image processing method for assessing multiple cracking developments in Strain Hardening Cementitious Composites (SHCC)," Cement and Concrete Composites, vol. 74, pp. 191-200, 2016. DOI: https://doi.org/10.1016/j.cemconcomp.2016.10.005.

[4] K. Gopalakrishnan et al, "Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection," Constr. Build. Mater., vol. 157, pp. 322-330, 2017.

[5] Z. Fan et al, "Automatic Pavement Crack Detection Based on Structured Prediction with the Convolutional Neural Network," 2018.

[6] F. Chen and M. R. Jahanshahi, "NB-CNN: Deep Learning-Based Crack Detection Using Convolutional Neural Network and Naïve Bayes Data Fusion," IEEE Transactions on Industrial Electronics, vol. 65, (5), pp. 4392-4400, 2018.

[7] J. Tenenbaum, V. de Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," Science, vol. 290, (5500), pp. 2319-23, 2000.

[8] L. Van Der Maaten and G. Hinton, "Visualizing data using t-SNE," Journal of Machine Learning Research, vol. 9, pp. 2579-2625, 2008.

[9] A. K. Das and C. Leung, "Power spectral entropy of acoustic emission signal as a new damage indicator to identify the operating regime of strain-hardening cementitious composites," Cement and Concrete Composites, pp. 103409, 2019. DOI: https://doi.org/10.1016/j.cemconcomp.2019.103409.

[10] ACI, "ACI manual of concrete practice 2005: Cement and concrete terminology." vol. ACI 116R-00,

[11] D. M. Chan et al, "t-SNE-CUDA: GPU-Accelerated t-SNE and its Applications to Modern Data," 2018.