Acta Polytechnica CTU Proceedings **49**:85–91, 2024
https://doi.org/10.14311/APP.2024.49.0085

Published by the Czech Technical University in Prague

RECONSTRUCTION OF CONCRETE MORPHOLOGY USING DEEP LEARNING

Ondřej Šperl, Jan Sýkora*

Czech Technical University in Prague, Faculty of Civil Engineering, Department of Mechanics, Thákurova 7, 160 00 Prague, Czech Republic

* corresponding author: jan.sykora.1@fsv.cvut.cz

Abstract. In this contribution, the concrete morphology is reconstructed with a simple algorithm selecting a pixel value based on the small set of surrounding pixels. A deep neural network (DNN) is used as a classifier, and the authors focus on studying different DNN architectures. The performance of the proposed algorithm is evaluated on several statistical descriptors and the grain size distribution curve.

Keywords: Reconstruction, concrete, deep learning, convolutional neural network.

1. Introduction

The reconstruction of material morphology has become an essential part of the numerical computational modeling of heterogeneous materials. The first reconstruction algorithms employed optimization routines to minimize selected statistical descriptors, see [1–4]. The drawback of such an approach is the exhaustive computational time caused by sequential pixel-by-pixel replacement with subsequent evaluation of statistical descriptors. The second strategy concentrated on material description using random fields with optimized image-based correlation kernels, see [5]. This method dramatically reduces the dimensionality of the problem [6], however, it has the disadvantage of preserving maximal second-order statistical moments. The last approach is texture synthesis, which exploits the relatively simple principle of texture locality and stationarity for material reconstruction, see [7–9]. Locality means that a random pixel inside the texture is related only to a small set of surrounding pixels (see [10]), and stationarity can be explained by the following example. Assume a movable window placed in different positions in the texture. The regions of texture marked by the movable window seem to be always similar contrary to the situation of the general image, where the observed regions are clearly different. Our aim is to reconstruct the image using the latter-mentioned approach, focusing on different DNN architectures, see [11], and the influence of their hyperparameters on the resulting concrete morphology. The authors are aware of more progressive techniques, such as generative adversarial networks or diffusion models, see [12–14], requiring a relatively rich dataset of images used for their training process, that might not always be available.

2. Methodology

The reconstruction of the concrete cross-section is based on the algorithm shown in Figure 1. As an input to the computation, we use a sample of the concrete cross-section characterized by a two-dimensional image, introduced in the algorithm by a matrix of values 0 and 1 representing the black and white colors. The input and output data used for training the model are generated from the original image. The size of the dataset depends on the parameters of the computation; however, it can be estimated that for images ranging from 200×200 to 400×400 px, the size of the dataset is around tens of thousands. The model here is defined as a classification tool determining, based on a given pixel neighborhood, whether a pixel at a given position has a value of 0 or 1. The reconstruction algorithm then uses the trained classifier and the edges of the original image for the prediction of a new concrete cross-section. The quality of the new morphology is evaluated on our selected statistical descriptors. The aim of the reconstruction is to generate a similar sample with a statistically and physically equivalent structure with respect to the studied medium.

The pixel neighborhood is an important parameter of the reconstruction algorithm. Figure 2 shows the template as a function of the parameters w and h. These two parameters define its dimension, which significantly affects the properties of the entire reconstruction. The pixel neighborhood has two major features:

(1.) It specifies the number of input values of pixels needed for the output value. Yellow-labeled pixels in the scheme are our input values, and the red-labeled pixel is the output value. Since the values of the yellow-labeled pixels have to be known, the reconstruction algorithm must always have the edges of the original image enabling the prediction of the first red pixel. Thus, the parameters w and h also characterize the sizes of the boundary regions of the original image, which are needed for the start of the reconstruction algorithm.

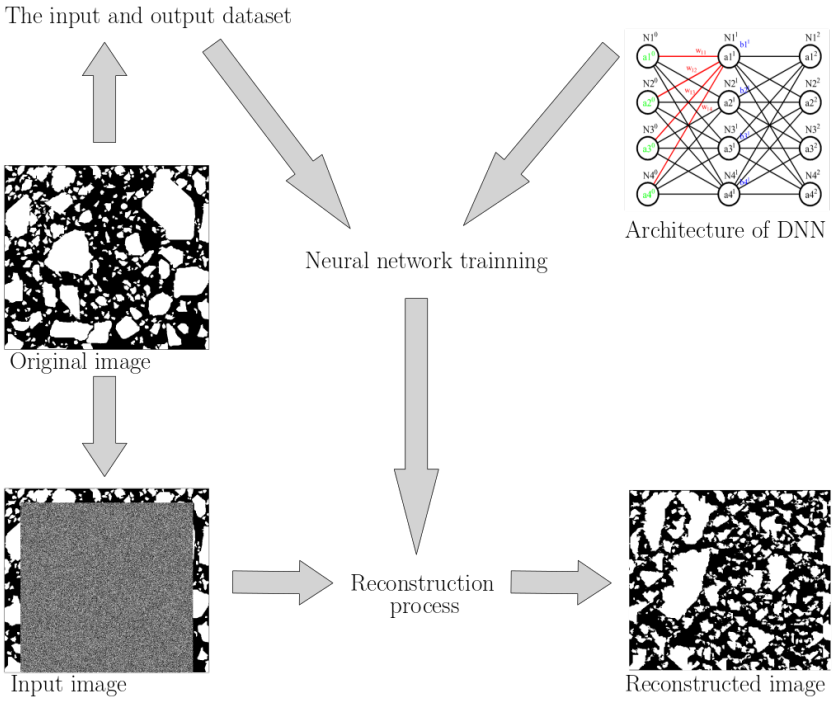


Figure 1. Algorithmic framework.

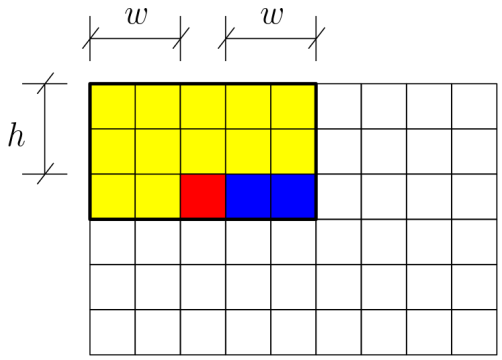


Figure 2. Template determining set of surrounding pixels.

(2.) Moreover, it defines a template that moves pixel-by-pixel within the reconstructed image and predicts the value of the red-labeled pixel. It can be seen from the scheme that the rectangle of size $(2w+1)\cdot(h+1)$ is used as a template, and for this reason, we have to use blue-labeled and red-labeled pixels as input values. These are the values that are not taken from the original image but are randomly and independently generated from a uniform distribution satisfying the volume representation of 0 and 1 values in the original image.

Once the pixel neighborhood is set and the classifier is trained, the reconstruction algorithm starts the execution with the given template from the upper left corner of the initial image. The initial image is shown in Figure 1, with the edges representing the morphology of the original media and the rest being randomly determined pixels so as to preserve the volume representation. The red-labeled pixel is computed with the help of the classifier specifying the probability of 0 and 1 for a given vicinity and the pseudo-random number generator determining the final value. The newly determined red-labeled pixel becomes the yellow-labeled pixel, and then the template is shifted one pixel further to the right. The value of all reconstructed pixels is successively identified in this fashion. The reconstruction algorithm terminates in the lower right corner of the image. Subsequently, the metrics of the statistical and physical descriptors are evaluated.

2.1. Architecture of DNN

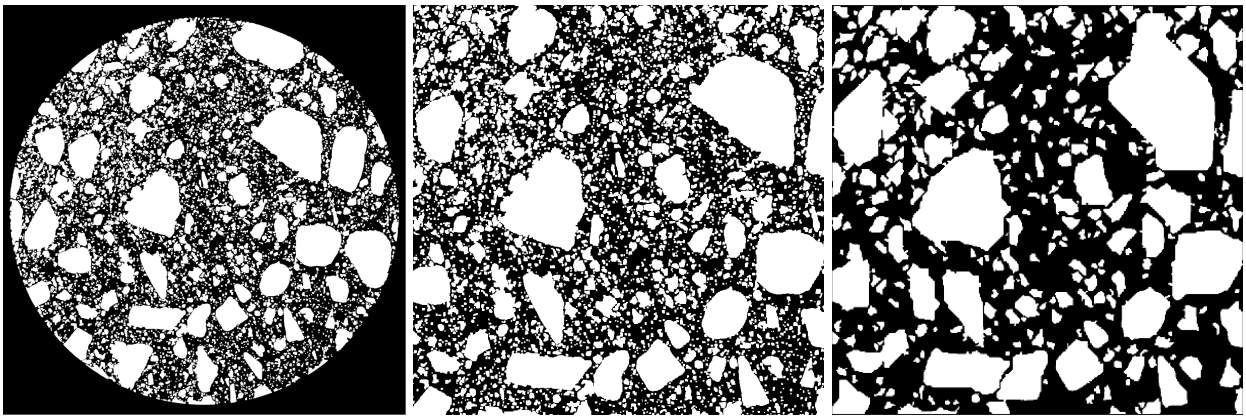
Deep neural learning is the subclass of machine learning using DNN as a tool for solving complex problems. Each DNN has multiple layers of interconnected nodes enabling one to learn complex representations of data by discovering hierarchical patterns and features, see [15]. The regular DNN is composed of three types of layers:

(1.) Input layer processing input data into the model,

(2.) hidden layers representing the key part of DNN and applying weights to the inputs and directing them through an activation function as the outputs,

(3.) output layer is the last layer in the neural network calculating the final probability scores of desired outputs.

The special case of DNN is the convolutional neural network (CNN) developed for image classification and data visual interpretation, see [16]. It is based on the principles of mathematical convolution extracting the important features from the image with the help of learnable filters. The CNN is primarily composed of



(A). Original segmented image – 1 200 × 1 225 px.

(B). The largest possible square image cut from the original segmented image – 816 × 816 px.

(C). The final image adjusted for redundant pores – 400 × 400 px.

FIGURE 3. Input image preparation.

convolutional layers and max pooling layers reducing the number of weights in the neural network. Our designed classifiers are built upon the standards of DNN and CNN architectures.

3. EXAMPLE

The example devoted to the reconstruction of the concrete structure using deep learning techniques is illustrated in this section. The sample of concrete morphology that is being reconstructed is depicted in Figure 3a. It is a CT scan of the concrete transformed by a segmentation algorithm into a monochrome image. The white pixels indicate the aggregates and the pores, while the cementitious binder in between is represented by black pixels. The sample image was captured on a 74 mm diameter concrete cylinder. For algorithmic simplicity, the largest possible square image with dimensions of 816×816 px is selected from the original segmented image, see Figure 3b. Although the training process of our computational model is accelerated on a graphics card (Nvidia GeForce RTX 2060), it turns out that it is not possible to work with such a huge dataset collected from the largest possible square image. Therefore, we reduced the image to a final resolution of 400 × 400 px minimizing the work with large data files. Unfortunately, the segmentation algorithm does not allow to identify between the pore and the aggregate and labels both phases identically as white pixels. Since the original grain size distribution curve is known, a portion of the small white pixels has been removed so that the grain size distribution curve is as close as possible to the original mixture curve of the concrete sample. The histogram shows the grain size distribution curve of the final image determined by the principal component analysis (PCA) and Feret's characteristics (Feret), and the real volume distributions of aggregates in the investigated concrete, see Figure 4. The imperceptible differences in the histogram are probably mainly caused by the absence of the third grain dimension in the calculation of the grain size distribution curves. The final 400 × 400 px image, adjusted for redundant pores, is shown in Figure 3c.

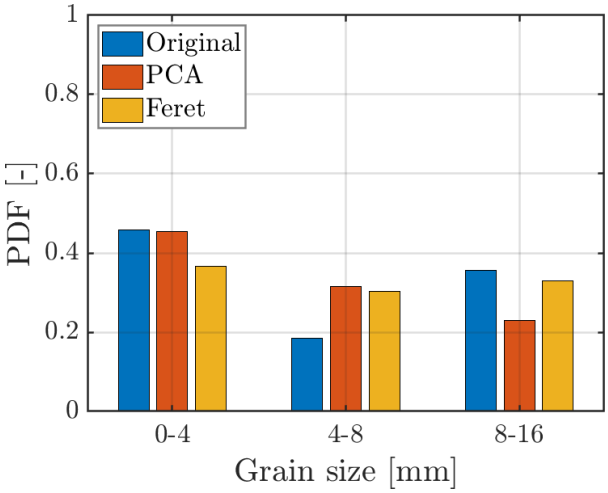


FIGURE 4. Grain size distribution curve of final image utilized for the reconstruction process. The original data (blue-labeled) represents the concrete mixture used for the CT scanning. The red-labeled and orange-labeled grain size distribution curves display the results computed for the PCA and Feret's algorithm, respectively.

The computational image reconstruction algorithm offers various solutions with different success rates influenced by several variables. The variables affecting the reconstruction process are mainly:

(1.) Hyper-parameters of the model. These are the parameters controlling the neural network architecture, and some of their aspects are examined here in detail. In particular, the type of neural network layer and its input parameters such as the number of neurons, the composition of the layers in a sequence, or the total number of layers.

(2.) The size of the pixel neighborhood is the second essential parameter that significantly affects

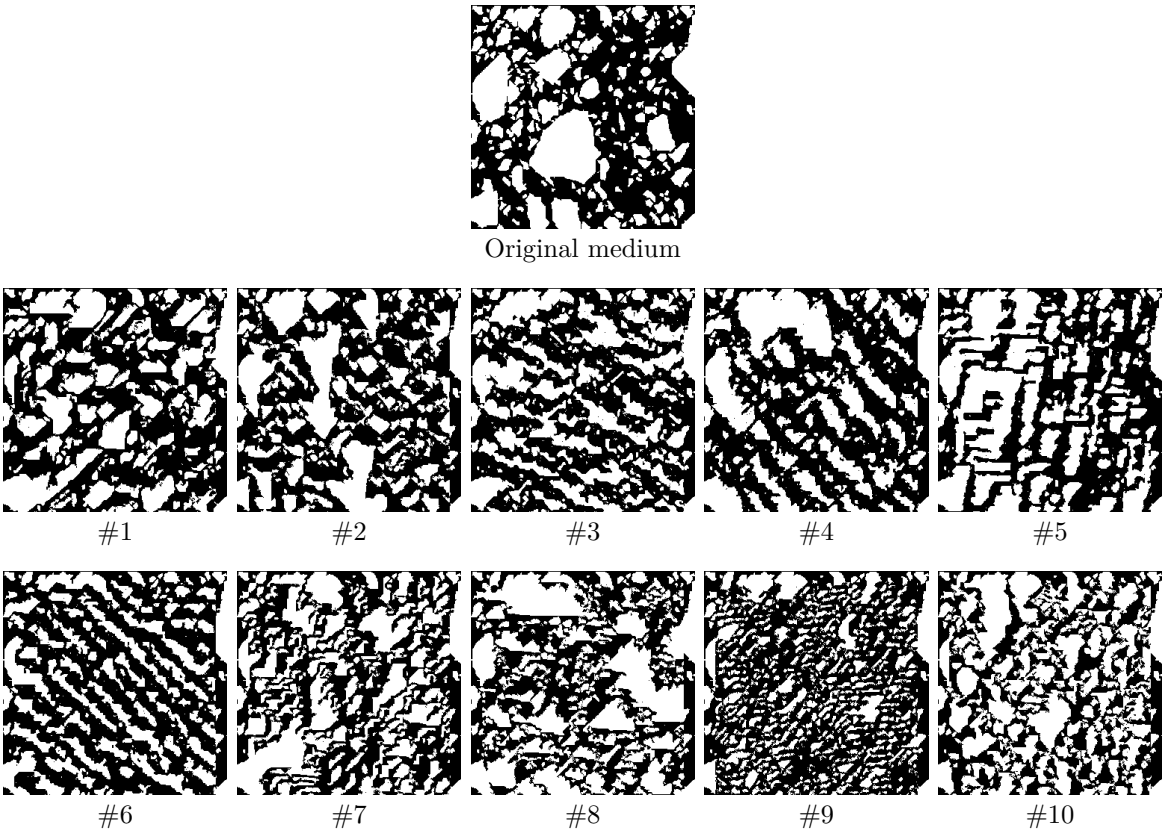


Figure 5. The original and resulting reconstructed images (#1–#10).

Model ID	Architecture of DNN
#1	15x15_c(16,3x3)_c(16,3x3)_m(2x2)_d(64)_d(64)_d(64)
#2	15x15_c(16,3x3)_c(16,3x3)_m(2x2)_d(128)_d(128)_d(128)
#3	15x15_c(32,3x3)_c(32,3x3)_m(2x2)_d(64,l1_0.01)_d(64)_d(64)
#4	15x15_d(128)_d(128)_d(128)
#5	15x15_d(128)_d(128)
#6	15x15_d(64)_d(64)_d(64)
#7	15x15_c(16,3x3)_c(16,3x3)_m(2x2)_d(128)_d(128)_d(128)
#8	15x15_c(16,3x3)_c(16,3x3)_m(2x2)_d(128)_d(128)
#9	15x15_c(8,3x3)_c(8,3x3)_m(2x2)_d(128)_d(128)_d(128)
#10	15x15_c(8,3x3)_c(8,3x3)_m(2x2)_d(64)_d(64)_d(64)

Table 1. Model ID's and abbreviated names of DNN architecture: For illustration, 15×15 stands for the pixel neighborhood, i.e. the dimensions of w and h in pixels, c(16,3x3) is the convolutional layer with 16 filters of spatial size 3x3, m(2x2) is the max pooling layer with 2x2 filter, and d(64) is the dense layer with 64 neurons.

the success rate of the entire model. This parameter has also been investigated; however, due to the limited space of the paper, we limit ourselves to claiming that for the problem illustrated here, a pixel neighborhood of 15×15 px seems optimal.

The hyper-parameters of the model are investigated on the image with dimensions of 300×300 px, see Figure 5. The reason for reducing the image to acceptable dimensions is again the higher computational effort of the entire study, which is performed on a series of 10 architectures of the neural network with differently composed convolutional, dense, and max pooling layers. The model ID's and abbreviated names of DNN architecture are summarized in Table 1. It is obvious that the possibility of designing the DNN architecture is enormous, especially in terms of filter size or number of neurons; thus this study is limited more to the comparison between the CNN and DNN architectures.

The resulting reconstructed images for ten different DNN architectures are depicted in Figure 5. Besides the visual comparison, the following table (Table 2) displays the values of errors in the statistical descriptors computed for the original and reconstructed structures. From the wide range of statistical descriptors, three fundamental metrics are selected for our study – the volume fraction (ϕ), the two-point proba-

Model ID	$\epsilon_{\phi}^{\text{white}}$	$\epsilon_{S2}^{\text{white}}$	$\epsilon_{S2}^{\text{black}}$	$\epsilon_{L2}^{\text{white}}$	$\epsilon_{L2}^{\text{black}}$
#1	0.001797	0.000157	0.000158	0.000171	0.000106
#2	0.009209	0.000185	0.000220	0.000205	0.000117
#3	0.010342	0.000238	0.000284	0.000202	0.000148
#4	0.011240	0.000247	0.000303	0.000213	0.000153
#5	0.054454	0.002175	0.002939	0.000796	0.000557
#6	0.055820	0.001978	0.003782	0.000727	0.000627
#7	0.060252	0.002672	0.003554	0.000884	0.000614
#8	0.092767	0.006558	0.007819	0.001403	0.000919
#9	0.094060	0.004631	0.010755	0.001173	0.001081
#10	0.113770	0.010151	0.011235	0.001756	0.001106

TABLE 2. The resulting values of errors calculated for the volume fraction, the two-point probability function, and the lineal path function.

bility function ($S2$), and the lineal path function ($L2$). These statistical descriptors have been used in reconstruction problems for a significant period and have become an essential part of numerical studies devoted to material morphology. Their particular formulations are skipped here, and the interested reader referred to, e.g., [17] or [2] providing basic characteristics, illustrative examples, and implementation strategies. The values of errors are calculated according to the following formulas expressing the differences between values of selected statistical descriptors calculated for the original and reconstructed image:

$$\epsilon_{\phi}^{\text{phase}} = \phi_{\text{orig}}^{\text{phase}} - \phi_{\text{rec}}^{\text{phase}}, \tag{1}$$

where

$\phi_{\text{orig}}^{\text{phase}}$ is the volume fractions of a given phase corresponding to the original image,

$\phi_{\text{rec}}^{\text{phase}}$ is the volume fractions of a given phase corresponding to the reconstructed image.

The values of errors for complex statistical descriptors are evaluated as:

$$\epsilon_{p}^{\text{phase}} = \frac{\sum_{r=1}^{R}\sum_{s=1}^{S}(p_{rs,\text{orig}}^{\text{phase}} - p_{rs,\text{rec}}^{\text{phase}})^2}{SR}, \tag{2}$$

where

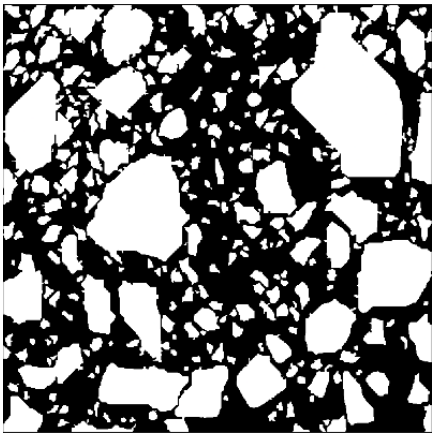
R and S are the image dimensions,

p is the chosen statistical descriptor, here $p = \{S2, L2\}$.

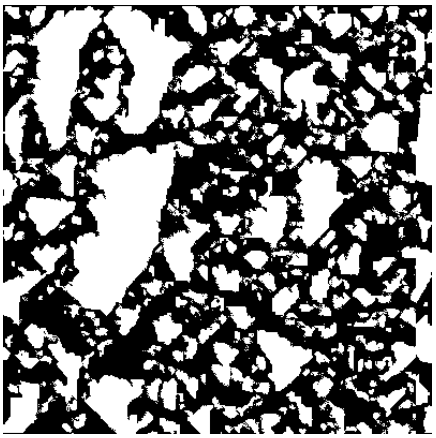
From the resulting values of errors and comparison of the reconstructed images, two models #1 and #2 have been studied in detail and more runs of reconstruction algorithm have been performed focusing on the statistical moments of error analysis. The most stable classifier is model #2, which is further utilized for reconstructing the larger image of the concrete cross-section in the following paragraph.

3.1. Large image Evaluation

The final reconstruction is carried out on an image of dimensions 400×400 px, the original image is shown in Figure 6a, and the reconstructed result is in Figure 6b. It is apparent that the reconstruction algorithm is quite successful from the visual comparison of these two images. However, some irregularities are observed in the shape of the grains compared to the original image. From the perspective of statistical descriptors, the following values of errors – $\epsilon_{\phi}^{\text{white}} = 0.001046$, $\epsilon_{S2}^{\text{white}} = 0.000171$, and $\epsilon_{L2}^{\text{white}} = 0.000069$ – indicate



(A). Original image – 400×400 px.



(B). Reconstructed image obtained with the model #2 – 400×400 px.

FIGURE 6. Final image reconstruction.

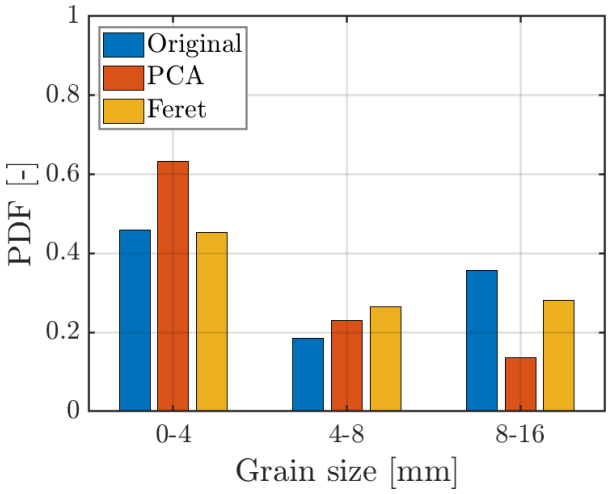


Figure 7. Comparison of grain size distribution curves obtained for the original and reconstructed concrete cross-sections.

again good performance of proposed strategy. The last comparison (see Figure 7) shows the grain size distribution curves of original and reconstructed images computed for the algorithm using Feret diameters (Feret) and principal component analysis (PCA). In the case of the PCA evaluation, the original and reconstructed images differ in the representation of the individual grain size as follows: size range of 0–4 mm – 17.7 %, 4–8 mm – 8.6 %, and 8–16 mm – 9.1 %. In the case of the Feret diameters, the grain size distribution differs as follows: size range of 0–4 mm – 8.6 %, 4–8 mm – 3.9 %, and 8–16 mm – 4.7 %. Thus, the reconstructed image contains more aggregates of 0–4 mm since both methods result in a higher volume percentage, and conversely, size ranges of 4–8 mm and 8–16 mm are less observed in the reconstructed image. From the perspective of the grain size distribution curve, the reconstruction process of concrete morphology is less effective.

4. Conclusion

The reconstruction of the concrete cross-section area using deep learning techniques is illustrated in this paper. The proposed model can generate reasonably accurate samples of concrete morphology in terms of the selected statistical descriptors. Some limitations are observed in the results of the grain size distribution curve, and the shapes of the aggregates which appear to be more irregular. Taking into account that the computational hardware has some limitations and also the fact that we only handled one image, the obtained result can be evaluated positively.

Acknowledgements

The authors are thankful for financial support from the Student Grant Competition of CTU, project No. SGS23/152/OHK1/3T/11 and the Czech Science Foundation, project No. 22-35755K.

References

[1] C. L. Y. Yeong, S. Torquato. Reconstructing random media. *Physical review E* **57**(1):495, 1998. https://doi.org/10.1103/PhysRevE.57.495

[2] J. Havelka, A. Kučerová, J. Sýkora. Compression and reconstruction of random microstructures using accelerated lineal path function. *Computational Materials Science* **122**:102–117, 2016. https://doi.org/10.1016/j.commatsci.2016.04.044

[3] R. Bostanabad, Y. Zhang, X. Li, et al. Computational microstructure characterization and reconstruction: Review of the state-of-the-art techniques. *Progress in Materials Science* **95**:1–41, 2018. https://doi.org/10.1016/j.pmatsci.2018.01.005

[4] M. V. Karsanina, K. M. Gerke. Stochastic (re)constructions of non-stationary material structures: Using ensemble averaged correlation functions and non-uniform phase distributions. *Physica A: Statistical Mechanics and its Applications* **611**:128417, 2023. https://doi.org/10.1016/j.physa.2022.128417

[5] M. Lombardo, J. Zeman, M. Sejnoha, G. Falsone. Stochastic modeling of chaotic masonry via mesostructural characterization. *International Journal for Multiscale Computational Engineering* **7**(2):171–185, 2009. https://doi.org/10.1615/IntJMultCompEng.v7.i2.70

[6] J. Havelka, A. Kučerová, J. Sýkora. Dimensionality reduction in thermal tomography. *Computers & Mathematics with Applications* **78**(9):3077–3089, 2019. https://doi.org/10.1016/j.camwa.2019.04.019

[7] R. Bostanabad, A. T. Bui, W. Xie, et al. Stochastic microstructure characterization and reconstruction via supervised learning. *Acta Materialia* **103**:89–102, 2016. https://doi.org/10.1016/j.actamat.2015.09.044

[8] J. Fu, S. Cui, S. Cen, C. Li. Statistical characterization and reconstruction of heterogeneous microstructures using deep neural network. *Computer Methods in Applied Mechanics and Engineering* **373**:113516, 2021. https://doi.org/10.1016/j.cma.2020.113516

[9] K. Latka, M. Doškář, J. Zeman. Microstructure reconstruction via artificial neural networks: A combination of causal and non-causal approach. *Acta Polytechnica CTU Proceedings* **34**:32–37, 2022. https://doi.org/10.14311/APP.2022.34.0032

[10] L.-Y. Wei, M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 479–488. 2000. https://doi.org/10.1145/344779.345009

[11] V. Sze, Y.-H. Chen, T.-J. Yang, J. S. Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE* **105**(12):2295–2329, 2017. https://doi.org/10.1109/JPROC.2017.2761740

[12] K.-H. Lee, G. J. Yun. Microstructure reconstruction using diffusion-based generative models. *Mechanics of Advanced Materials and Structures* **31**(18):4443–4461, 2024. https://doi.org/10.1080/15376494.2023.2198528

[13] C. Düreth, P. Seibert, D. Rücker, et al. Conditional diffusion-based microstructure reconstruction. *Materials Today Communications* **35**:105608, 2023. https://doi.org/10.1016/j.mtcomm.2023.105608

[14] X. Lyu, X. Ren. Microstructure reconstruction of 2D/3D random materials via diffusion-based deep generative models. *Scientific Reports* **14**(1):5041, 2024. https://doi.org/10.1038/s41598-024-54861-9

[15] F. Chollet. *Deep learning with Python*. Manning, 2021. ISBN 9781617296864.

[16] Z. Li, F. Liu, W. Yang, et al. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems* **33**(12):6999–7019, 2022. https://doi.org/10.1109/TNNLS.2021.3084827

[17] J. Zeman. *Analysis of composite materials with random microstructure*. Ph.D. thesis, Czech Technical University, 2003. https://doi.org/10.13140/RG.2.1.2399.0004