IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

IterLUNet: Deep Learning Architecture for Pixel-Wise Crack Detection in Levee Systems

Manisha Panta [1, 2], Md Tamjidul Hoque [1, 2], Mahdi Abdelguerfi [1, 2], Maik C. Flanagin [3]
[1]Canizaro/Livingston Gulf States Center for Environmental Informatics, University of New Orleans, New Orleans, LA 70148, USA
[2]Department of Computer Science, University of New Orleans, New Orleans, LA 70148, USA
[3]US Army Corps of Engineers, New Orleans District, LA, USA

Corresponding author: Md Tamjidul Hoque (e-mail: thoque@uno.edu).

ABSTRACT Deep learning has recently been extensively used for crack detection in structural health monitoring settings. However, detecting cracks in levee systems have yet to receive considerable critical attention. Thus, this study presents a novel encoder-decoder-based fully convolutional neural network to detect cracks from levee images at a pixel level automatically. We propose that the feature learning be strengthened using the decoder and bottleneck feature maps by concatenating them back to the encoder blocks. The addition reinforcement in the U-Net-like architecture results in a loop-like structure to exploit all the feature maps from encoders, bottlenecks, and decoders. The proposed architecture, Iterative Loop U-Net (IterLUNet), outperforms the state-of-the-art architectures on the image dataset of the levee system, achieving an increment of Intersection over Union (IoU) by 10.32% on average for a 10-Fold Cross-Validation (FCV) compared to the baseline U-Net model and 11.00%, 7.65%, and 7.43% with a range of latest models MultiResUnet, Attention U-Net, and Unet++ respectively. In addition, IterLUNet has at least 63% fewer parameters to be trained than the baseline model, thus, allowing less space consumption for pixel-wise crack detection in AI-based inspection of levee systems.

INDEX TERMS Crack Detection, Deep Learning, Floodwalls, Image Segmentation, Levees

I. INTRODUCTION

Recent deep learning methods have achieved state-of-the-art results on challenging computer vision problems like image classification, object detection, and image segmentation [1]. The Convolutional Neural Network (CNN or ConvNet) has significantly advanced deep-learning methods [2] by introducing three layers - the convolutional layer as a feature extractor, the activation layer to add non-linearity, and the pooling layer to maintain the spatial dimension. Consequently, CNN gained popularity mainly because it automatically extracted essential features through successive convolutional layers. On the grounds of components of CNN, Long *et al.* [3] proposed Fully Convolutional Network (FCN), a breakthrough in deep-learning-based end-to-end image segmentation methods without fully connected layers. The FCN was then extended to encoder-decoder architectures. The encoders in encoder-decoder architecture extract features from the images, and the decoders map low-level features from encoders to an output segmentation mask [3-5].

Several fully convolutional neural network-based architectures, FCN [3], SegNet [4], U-Net [5], MultiResUNet [6], Attention U-Net [7], and UNet++ [8], had been applied before to perform semantic or pixel-wise segmentation in medical imaging dataset. However, U-Net is a widely used encoder-decoder architecture that succeeded as the state-of-the-art model for image segmentation tasks in medical imaging [5]. The commonality in the variants of U-Net-like models is that they have skip connections from the encoder to the decoder to help retrieve any spatial information lost in the down-sampling path of the encoders. Hence, in this paper, we explore that the U-Net-like deep learning architectures have the potential to improve prediction on limited and complex datasets like levee crack images through two main hypotheses. First, the proposed model, IterLUNet, improves performance by utilizing learned features from the decoder and bottleneck layer to feed the feature map back to the encoder. Secondly, U-Net-like architectures can be made deeper without increasing the number of training parameters by implementing existing concepts in deep learning to design encoder and decoder blocks.

The proposed deep learning architecture directly learns meaningful underlying representations of cracks from the image dataset. Of course, the training process requires a considerable amount of labeled data which is a challenge in flood control systems where there need to be more images with cracks to train and evaluate models. Paradoxically, collecting levee crack images is labor and time intensive. In light of this, we aim to develop a deep learning architecture that can be trained using a small labeled dataset and assist during the field investigation performed through a handheld device or unmanned aerial vehicles. Furthermore, most deep-learning approaches detect cracks on concrete or asphalt surfaces, predominantly in civil infrastructure. Existing architectures have yet to address the complexities of surroundings in the levee system where cracks develop on the slopes, crest, concrete floodwalls, and areas nearby the structure.

Currently, the inspection of the flood water control system is done manually. Mostly, field investigators physically gather or fly drones to capture images, followed by hours of manual checking for any faults [9, 10]. The current inspection method is expensive, slow, and laborious. Thus, this research introduces a high-performance, fully automated AI-based inspection solution using an encoder-decoder-based fully convolutional neural network architecture to detect cracks from the levee images. Therefore, in this study, the U-Net model is further improved to address the limitation and intricacies of the levee crack dataset. The contributions of the proposed model in this paper can be summarized as follows:

- With the underlying hypothesis that decoder and bottleneck outputs can reinforce the model's learning, we propose Iterative Loop U-Net (IterLUNet), an encoder-decoder and a decoder-encoder combined deep learning model with three different high-performing model components.
- We present that the U-Net-like architectures can be constructed deeper and broader to extract relevant features without compromising on the model's size by deliberately including powerful contemporary deep-learning concepts.
- We propose a new benchmark dataset for performing image segmentation on levee crack images.

II. RELATED WORKS

The primary purpose of pixel-wise segmentation in this study is to separate crack pixels from non-crack pixels to accurately locate cracks in the levee from images and measure their size, provided the scale of the image. A considerable volume of literature has been published on automatically detecting cracks, ranging from U-Net architecture [11] to several variations of U-Net [11-24]. These approaches have a symmetrical contracting-expansive path with skip-connections concatenating encoder and decoder feature vectors. Likewise, Zou *et al.* [24] developed DeepCrack, a SegNet-like architecture, to demonstrate the utilization of multi-scale convolutional features for better results and model convergence. In DeepCrack, encoder and decoder outputs are connected to build a single-scale fused feature map. The hierarchical feature maps are combined to produce a multi-scale fusion map which is further used to compute loss and the final output mask.

Lately, detecting cracks in the levee system has gained interest [25] by using object detection methods. The authors in [25] analyzed machine learning and deep learning-based techniques and suggested a lightweight stacking-based model for edge devices like drones. The significant difference in this research is that, unlike in [25], where the authors detected a bounding box of cracks, the architecture developed in this study uses a pixel-based annotated levee dataset to perform semantic or pixel-level detection of cracks. Detection of cracks using a pixel-level approach qualifies for precise identification of crack regions on the levee systems, a clear advantage over using a bounding box approach.

III. PROPOSED ARCHITECTURE

The baseline architecture U-Net is symmetric because of the contracting path with blocks of encoder followed by max pooling layer to generate feature vector and expanding path that has blocks of decoder along with upsampling of the feature space. The feature vectors generated through encoder blocks contain fine-grained spatial information lost in the contracting path. So, in U-Net, the skip connections from the contracting path to expanding path are constructed by concatenating the feature vector from the encoder to the corresponding decoder to allow the architecture to propagate the spatial information from previous layers while accurately reconstructing the segmentation mask [5]. The fundamental hypothesis constructed for the architecture design of IterLUNet is that the higher-level features from expanding paths also have relevant information which could be helpful during training. Thus, the proposed architecture is based on building connections from the expanding path back from the decoder to the encoder to represent the complexity of cracks.

In a deep learning model, a considerable number of parameters are to be tuned during the training process. It requires thousands of training samples for a model to learn from so it can generalize well on unseen data. Training deep learning models with many parameters from scratch is prone to overfitting in real-world semantic segmentation tasks where the annotated images are limited, less than, or in hundreds. Additionally, a model with a higher number of training parameters increases the model’s overall size, making it unfeasible to perform nearly real-time accurate segmentation of crack pixels from the non-crack pixels. Hence, a depthwise separable convolution and iterative loop-like structure are introduced to address the growing number of parameters and optimize the architecture to achieve higher performance. The decoder and bottleneck feature maps are iteratively concatenated to the encoder's input at the next stage using simple skip connections in a U-like shape, hence named Iterative Loop U-Net (IterLUNet), as illustrated in Fig. 2.

A. BUILDING BLOCKS

The primary components of IterLUNet are InitialBlock, Squeeze and Excitation (SE) Block, IntermediateBlock, and Iterative Loop Block (IterLBlock). In Fig. 1, substructure A, substructure B, and substructure C depict InitialBlock, IterLBlock, and IntermediateBlock, respectively, which are discussed in detail in the following sections.

1) INITIALBLOCK

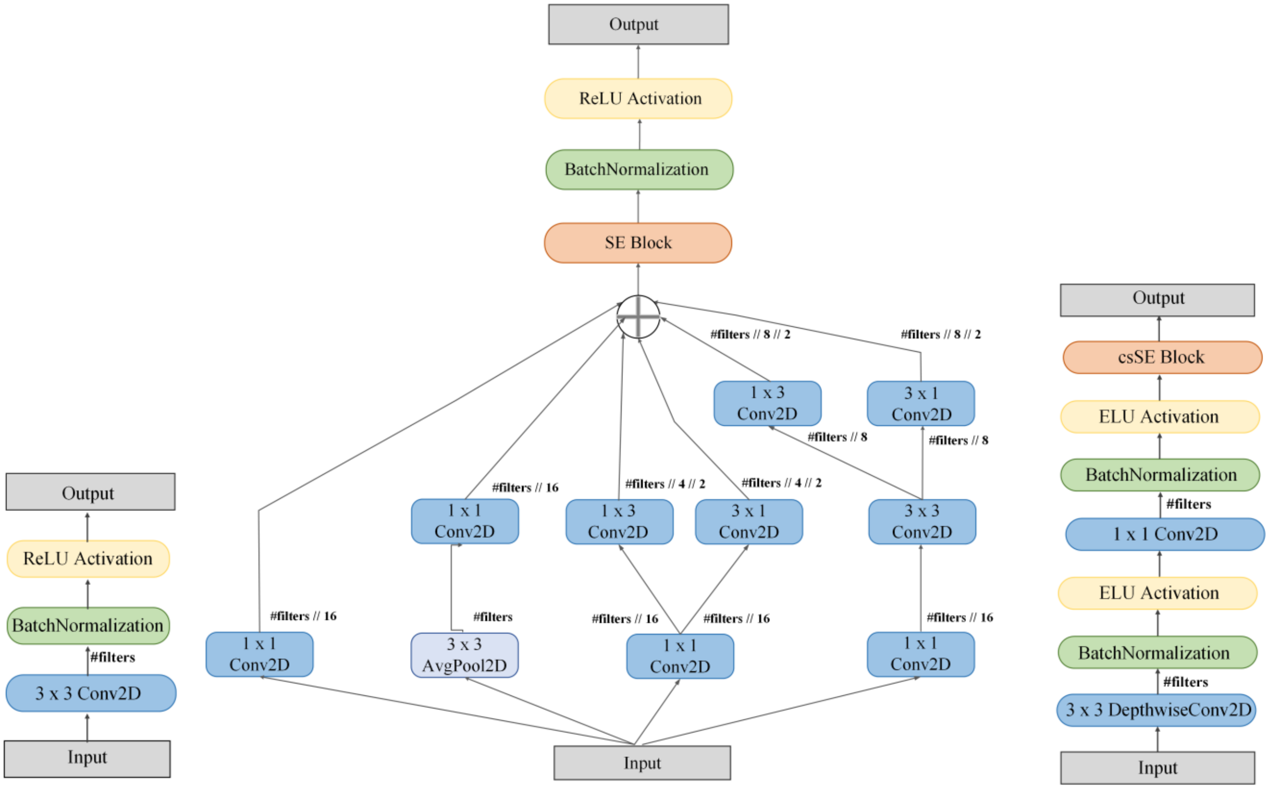
In [26], the authors show that the structure of an inception module with factorized asymmetric convolutions does not work well in the early layers. Since IterLUNet trains on a small dataset, the classic convolution layer in InitialBlock instead of an inception module helps reduce model complexity. The InitialBlock has one set of 3 × 3 convolution with a stride of 1, followed by batch normalization and ReLU activation as shown in Fig. 1. substructure A. It is the initial convolution block used in the first encoder in every iteration and produces 64 feature maps.

2) SE BLOCK

The skip connections combine low-level and high-level feature maps. Therefore, it is essential to recognize and prioritize meaningful latent representations. Thus, the Squeeze and Excitation (SE) block [27] and its variant, concurrent channel, and spatial SE (csSE) block proposed in [23] are used in the architecture. The SE block Squeezes along the spatial domain and Excites or reweights the channels. The advanced version of SE, csSE, on the other hand, emphasizes the use of proper channels and spatial information. Therefore, the SE and csSE blocks in the architecture recalibrate the feature space spatially and channel-wise, which is one way to optimize the network with a slight increment in model complexity and computational cost.

3) INTERMEDIATEBLOCK

The IntermediateBlock is comprised of a single Depthwise Separable Convolution followed by a csSE block, as observed in substructure C of Fig. 1. In Depthwise Separable Convolution (DSC) layer, the two separate cascaded



Substructure A
(InitialBlock)

Substructure B
(IterLBlock)

Substructure C
(IntermediateBlock)

FIGURE 1. **Substructure A is the standard initial convolutional block, Substructure B is the Inception-like module, IterLBlock, used in the encoder-decoder layers of IterLUNet. and Substructure C is the intermediate block with depthwise separable convolution followed by concurrent channel and spatial SE block. Here, #filters represent a total number of output filters after convolution operation or average pooling.**

operations generate latent representations of the concatenated intermediate feature maps. The first operation is 3 × 3 depthwise Convolution with a stride of one, dilation of one, and a depth multiplier to perform channel-wise spatial convolution.

Later 1 × 1 point-wise convolution operation with stride one follows batch normalization operation and ELU activation in the intermediate block, as shown in Fig. 1. The performance using ELU activation and batch normalization is a little enhanced and consistent compared to using ReLU activation mostly because ELU avoids dying ReLU problem and improves generalization through faster learning [28]. The DSC layer in the intermediate block performs similarly to the traditional convolution layer; however, the layer's significant advantage is that it lowers the number of training parameters. Finally, adding the csSE block after convolution operations ensures that concatenated filters are relevant both spatially and channel-wise to add value to the performance gain of the model.

4) ITERATIVE LOOP BLOCK (ITERLBLOCK)

Based on second hypothesis we propose IterLBlock. The balance of width and height in the proposed architecture is accomplished by managing a number of output filters produced throughout the network and recalibrating the importance of filters for optimal performance. Accordingly, the convolutions of larger spatial filters are factorized while retaining a growing number of filters in IterLUNet. The proposed substructure, iterative loop block (IterLBlock), follows the design principles introduced in [26], factorizing more extensive filter-sized operations into asymmetric convolutions. The inception module-like substructure B has 1 × 1, 3 × 3, and 5 × 5 convolutions, as shown in Fig. 1. The 5 × 5 convolution operation is computationally expensive and slow, so it is replaced with 3 × 3 convolutions, which are further factorized into two asymmetric convolutions, 1 × 3 and 3 × 1 convolution. The order of operations is illustrated in Fig. 1. Substructure B. After each convolution operation, ReLU non-linearity follows a batch normalization layer. After each convolution operation, ReLU non-linearity

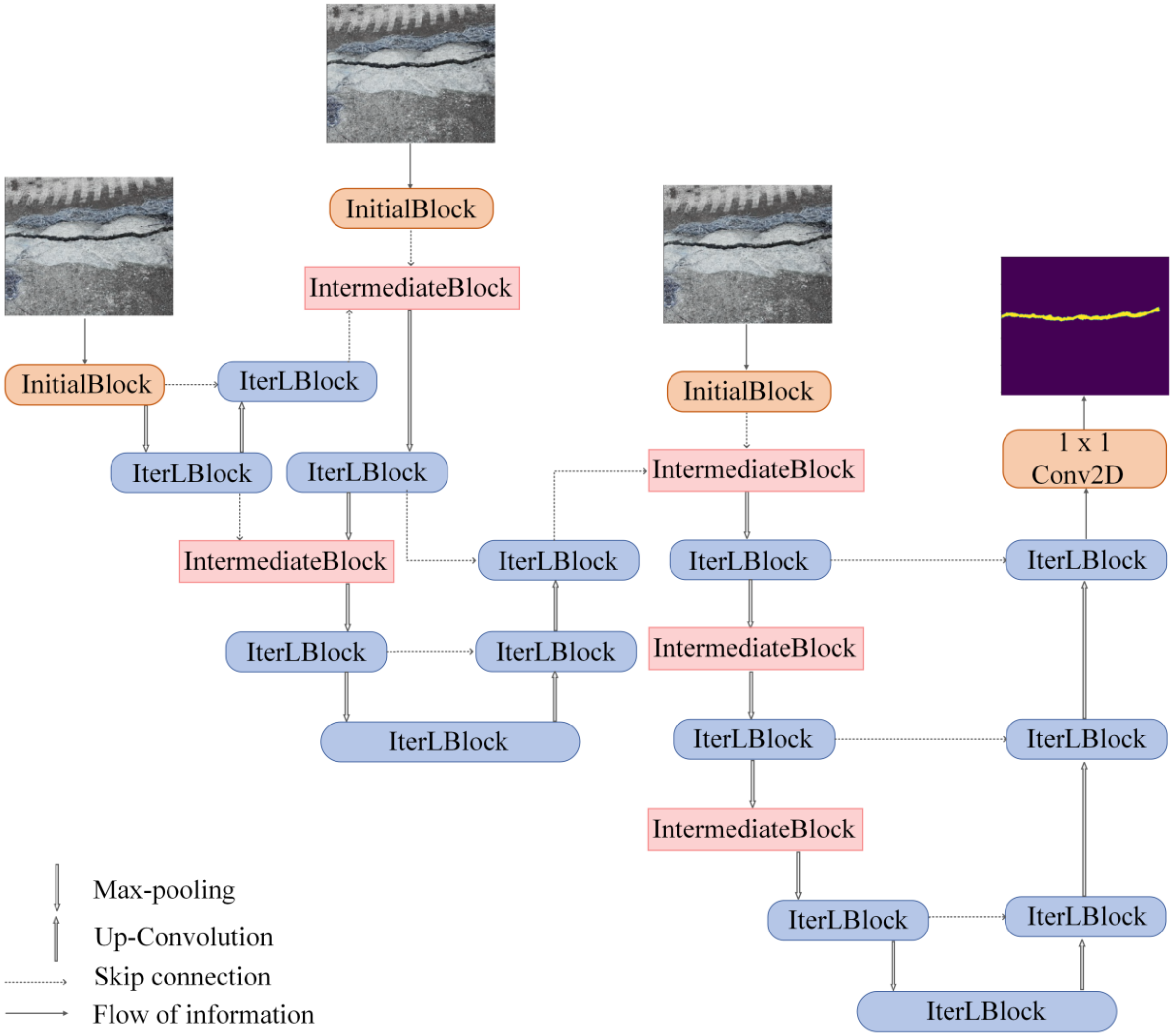


FIGURE 2. Proposed Iterative Loop U-Net (IterLUNet) Architecture. The loop structure allows utilization of the output feature maps of decoders and bottlenecks. Simple feature concatenation is used as Skip connection. Features of the original image are extracted at the beginning of each loop. Different blocks used in the design are illustrated in Fig. 1.

follows a batch normalization layer. Throughout the network, the batch normalization layer after each convolution adds regularization, reducing the need for a dropout layer, subsequently avoiding overfitting the model on the levee crack dataset.

The substructure B operates as a feature extractor conceptually similar to a classic convolutional layer. As the network advances more in-depth, the input to IterLBlock eventually receives a higher-dimensional feature vector since features of different scales and dimensions are concatenated. The higher dimensional feature vector is predisposed to exploding during training without advanced computational resources. So, IterLBlock adds computational efficiency without compromising the model's performance through two factors. Firstly, 1 × 1 convolution aims to reduce the dimensionality of the feature vector by compressing channels. The 1 × 1 convolution has made it possible to perform further expensive 3 × 3 and 5 × 5 convolutions for higher-dimensional input feature vectors. Secondly, stacking SE block or its variation after concatenation in the inception module as shown in Fig. 1. Substructure B with batch normalization has rectified the learning and added regularization in the network [29].

B. LOOPS AND ITERATIONS

In IterLUNet, loops are created to support connections from the decoder to the encoder. As the links increase, the number of encoder-decoder blocks also grows, leading to three iterations to match output filter numbers with the baseline model. The initial encoder in each iteration uses InitialBlock with 64 output feature maps extracted from the input RGB image, whereas decoders and bottlenecks apply IterLBlock, as illustrated in Fig. 2. After the first iteration, the pooling layer output is concatenated with the output of the respective expanding path to maintain the spatial dimension of the input feature vector for the succeeding encoder.

The first iteration has a simple U-like structure with one set of encoder-decoder blocks and a bottleneck layer of total filters {64, 128}. The second iteration starts exploring the output vector of the decoder and bottleneck layer of the first iteration. Immediately from the second iteration onwards, the number of encoder and decoder blocks increases. After that, IntermediateBlock accepts concatenated feature vectors as input. The number of output filters in the second iteration evolves to {64, 128, 256}. In the third iteration, pursuing the same idea of concatenating feature vectors, the output filter numbers in the contracting path become {64, 128, 256, 512}. Finally, 1 × 1 Conv2D represents the network's final layer, which comprises convolution operation with a sigmoid activation function on the output of the final decoder of the third iteration to generate an image of the segmentation mask. Since the architecture is designed to predict binary segmentation mask, the final layer with a filter of size 1 × 1, having sigmoid activation and 1 channel output size, maps the channels to the crack and background classes.

IV. EXPERIMENTS

A. DATASET

The dataset of levee crack images has been collected over the years by the field inspectors of the New Orleans district of the U.S. Army Corps of Engineers (USACE). The collected levee images have cracks in the levee's crest, concrete floodwalls, slopes, and even on and surrounding areas of the levee system. It can be observed that the images have different shapes and sizes of cracks on diverse backgrounds and surroundings. Fig. 3. (a), (b), (c), and (d) is the set of sample images with their ground truth. The levee crack dataset was first introduced in [30], which comprises 1650 images, and is used to conduct 10-Fold Cross-Validation of the proposed model and compare it with the latest encoder-decoder-based image segmentation models.

We expanded the overall dataset by annotating 101 more levee crack images using the VGG Image Annotator tool [31]. The tool generates a JSON file with coordinates of manually labeled crack regions. Eventually, the python script converts the coordinates in the JSON file to corresponding masks of the input images. Separation of training and independent test images was manually performed to distribute samples with as equal representations as possible in both training and test datasets. Table I represents the number of training and independent test images for different experiments. One of the main reasons for splitting datasets and conducting several experiments is to assess the robustness of the models trained on the currently available levee dataset and computational resources, further enabling the selection of models diligently. The datasets have a dominance of non-crack pixels over crack pixels. On average, only five percent of pixels in the original images are crack pixels, and the remaining ninety-five percent are background pixels. To further analyze the robustness of models, we also used the road crack dataset named DeepCrack, proposed by Liu, Yahui, et al. in their crack detection paper [22]. The DeepCrack test dataset has 237 images with their respective masks.

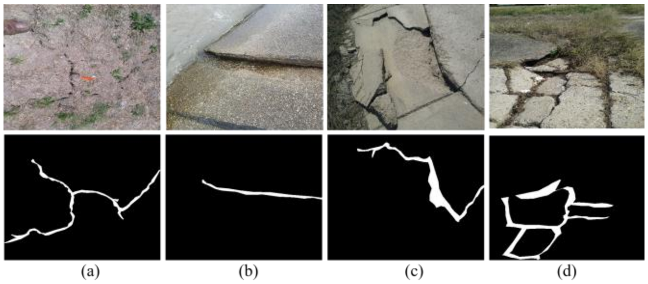


FIGURE 3. (a), (b), (c), and (d) are each set of one sample image and its corresponding segmentation mask.

TABLE I
TOTAL NUMBER OF IMAGES SEPARATED FOR TRAIN AND TEST

Dataset For	Training Images	Independent Test Images	Augmented Images
Loss Experiment	97	15	1746
Experiment 1	55	10	1650
Experiment 2	125	26	3750
Experiment 3	114	21	2850

B. PRE-PROCESSING

A significant challenge in building a deep learning model for real-world scenarios is maintaining the quality of training and evaluation datasets. Fig. 3. shows that the sample dataset has diverse textures and scenes, cracks of different scales, and undefined boundaries. The deep learning models should be robust enough to generalize on such a dataset. Thus, the preprocessing approach included carefully selecting original images, generating ground truth, applying augmentation techniques [32], and analyzing the performance of the baseline method. Based on the iterative approach, images and augmentation techniques contributing to the model learning process were determined. The twenty-nine augmentation techniques selected include affine, elastic, and pixel-level transformations such as ColorJitter, GaussianBlur, GaussianNoise, OpticalDistortion, and ElasticTransform, to name a few. Through the iterative approach, we identified that not all augmentation techniques contribute to the learning process, especially on the dataset in which background pixels are comparatively higher than the object to segment. Therefore, in the sample experiment and experiment 3, only seventeen and twenty-four augmentations were applied to the original training images and masks. Additionally, augmented levee crack images were resized to 256 × 256 due to computational constraints. Table I presents the statistics of the datasets for each experiment.

C. LOSS FUNCTIONS AND EVALUATION METRICS

The choice of the loss function and evaluation metrics highly determines the training process and robustness of the models. A pixel accuracy alone cannot reflect the performance of segmentation models. Thus, the models were assessed based on the accuracy of locating crack pixels and computing overlap scores between a predicted mask and ground truth. Equations (1), (2), (3), and (4) represent Intersection over Union (IoU) for crack pixels, Dice Coefficient, F1 Score, and Tversky Index as metrics to evaluate semantic segmentation models. The Dice Coefficient from (2) and F1 Score from (3) acts similarly during binary segmentation task such as segmenting crack pixels from the background. Dice loss in (5), based on the Dice Coefficient, attempts to address the class imbalance problem between crack and non-crack pixels to achieve the expected performance, as the loss function only considers the segmentation region during the training process [33]. However, the weights for both false positive and false negative detections are equally distributed, which makes dice loss less suitable when the class imbalance in the dataset is high. Therefore, in the experiments, we also introduced another loss function based on the Tversky index, the focal Tversky loss function in (6), to generate a balance between precision and recall in highly imbalanced datasets by adjusting values of $\alpha, \beta, and\ \gamma$ [34]

$$IoU\ Crack\ = \frac{|Y_{predicted}\ \cap\ Y_{groundtruth}|}{|Y_{predicted}\ \cup\ Y_{groundtruth}|} \qquad (1)$$

$$Dice\ Coefficient = \ 2\ x\ \frac{|Y_{predicted}\ \cap\ Y_{groundtruth}|}{|Y_{predicted}|\ +\ |\ Y_{groundtruth}|} \qquad (2)$$

$$F1\ Score\ = \frac{2\ x\ TP}{(TP + FP)\ +\ (TP\ +\ FN)} \qquad (3)$$

$$Tversky\ Index\ (TI)\ = \frac{TP}{TP\ +\ \alpha FN + \beta FP)} \qquad (4)$$

$$Dice\ Loss\ =\ 1\ -\ Dice\ Coefficient \qquad (5)$$

$$Focal\ Tversky\ Loss\ = (1\ -\ TI)^{\gamma} \qquad (6)$$

$$BCE\ Loss\ = -(\ Y_{groundtruth} \log(\ Y_{predicted}) + (\ 1 -\ Y_{groundtruth}) \log(\ 1 -\ Y_{predicted})) \qquad (7)$$

Here, $Y_{\text{predicted}}$ and $\mathrm{Y}_{\text{groundtruth}}$ represents predicted sets of pixels and ground truth. Likewise, TP, FP, and FN represent true positive, false positive, and false negative segmentation of crack pixels. $\alpha\ =\ 0.7$ and $\beta\ =\ 0.3$ are two parameters to penalize the model based on FNs and FPs, respectively, where their sum is 1. $\gamma\ =\ 0.75$ parameter controls the non-linearity of the loss.

It is evident from Fig.3. that the levee crack dataset is highly imbalanced since the percentage of crack pixels is less than that of non-crack pixels. Therefore, to understand the effects of different loss functions such as Dice Loss in (5), Binary Cross-Entropy (BCE loss) in (7), BCE Dice loss, and Focal Tversky loss in (6), adapted from [33], we further performed experiments on a sample dataset and recorded the evaluation metrics.

D. EXISTING MODELS

We compared IterLUNet to the U-Net [5] as the baseline model and the three advanced methods MultiResUNet [6], Attention U-Net [7], and UNet++ [8]. These methods implement encoder-decoder concepts and maintain filter numbers {32, 64, 128, 256, 512} which are the primary reasons for comparative analysis. Additionally, the selected models are well established in medical image segmentation, where the datasets have irregular shapes and variable sizes of objects with noisy or ill-defined boundaries. Table II shows all models' total number of parameters and Floating-Point Operations per Second (FLOPs). It can be observed that the IterLUNet has seventy percent fewer parameters to train on average than the base models. The design of the proposed model significantly reduces the number of training parameters because of the Depthwise Separable Convolution Layer (DSC). The previous research works indicate that DSC layers reduce the model's complexity by maintaining fewer parameters than standard CNN, as shown in Table II.

TABLE II

STATISTICS OF THE TOTAL NUMBER OF TRAINING AND NON-TRAINING PARAMETERS OF ALL ARCHITECTURES

Models	Trainable parameters	Non-trainable parameters	FLOPs (G)
U-Net (M1)	7.76E+06	5.88E+02	**12.11**
MultiResUNet (M2)	7.24E+06	2.45E+04	15.81
Attention U-Net (M3)	8.90E+06	9.73E+03	17.24
UNet++ (M4)	9.16E+06	7.30E+03	34.54
IterLUNet (M5)	**2.87E+06**	**1.53E+04**	16.41

E. EXPERIMENTAL SETUP

All segmentation models were implemented using the Keras framework and trained on NVIDIA K80 GPU. The convolutional layers in each model were initialized using He Initialization [37] and a batch size of 4. For a 10-Fold CV, the models were trained to minimize binary cross-entropy with logits with an Adam optimizer using a batch size of 4 for 150 epochs. The initial learning rate (LR) was 1e-3 but decayed by 0.25 after every five epochs when the validation F1 score plateaued to the minimum value of 15e-6. Furthermore, early stopping was included to avoid overfitting during the model's training for each fold set.

For the second experiment, fifteen percent of an extended dataset of 3750 augmented images was used to validate and save the best-performing model. All models were trained to minimize dice loss with an Adam optimizer using a batch size of 4. We used an initial LR of 1e-4, which was reduced on a plateau by 0.15 after every five epochs until a minimum value of 15e-8. Finally, the model with the lowest validation loss over 80 epochs was saved to evaluate on independent test datasets.

The loss experiments for loss functions were conducted to analyze the effects of loss functions on highly imbalanced datasets like levee crack datasets. The training samples were eight percent of 1750 augmented images, and the remaining twenty percent was used as validation data. The initial learning rate for the Adam optimizer is 2e-3, which decreases by fifteen percent after eight epochs when validation loss ceases to decrease till 15e-8. We trained IterLUNet, our proposed model, and UNet++, the best among existing models, to 150 epochs and saved the best model. All the best models are evaluated on 15 independent levee crack images.

Likewise, images and augmentation techniques were carefully selected in the third experiment based on the analysis of results from experiment 1, experiment 2, and the sample experiment on loss functions. The training samples were eight percent of 2850 augmented images, and the remaining twenty percent was used as validation data. Here, the Focal Tversky loss function was minimized using the training hyperparameters similar to that used in the sample experiment of loss functions.

V. RESULTS

A. 10-FOLD CV PERFORMANCE

The trained models are evaluated using a held-out test dataset. The evaluation metrics - mean Io (mIoU), IoU for crack pixels, and F1 score (F1) for each fold were also recorded. Table III shows the average metrics presented in percentage ratios (%) of 10-Fold Cross-Validation (FCV) and hold-out test images for all models. The performance of the proposed architecture based on the metric F1 measure, on average, is 7.4% greater than the baseline U-Net (M1) model.

TABLE III

PERFORMANCE COMPARISONS OF THE PROPOSED ITERLUNET AND U-NET MODELS BASED ON A 10-FCV (VALID) AND A HOLD-OUT TEST DATASET (TEST)

Models	mIoU (%)	IoU Crack (%)	F1 (%)
M1 Valid	87.18	71.08	80.33
M2 Valid	87.78	70.54	79.92
M3 Valid	87.16	73.19	81.76
M4 Valid	87.50	73.37	81.86
M5 Valid	**90.75**	**79.26**	**86.73**
M1 Test	85.86	70.13	79.70
M2 Test	87.77	70.19	79.90
M3 Test	86.90	72.80	81.67
M4 Test	86.97	72.80	81.53
M5 Test	**90.06**	**78.91**	**86.64**

Furthermore, the best-performing model from 10-FCV was also evaluated on an independent levee crack dataset. It is observed in Fig. 4 MultiResUNet (M2) detected non-crack pixels better than crack, regardless of the higher mIoU. Both Attention U-Net (M3) and UNet++ (M4) performed well on independent levee crack images while generating segmentation masks, as shown in Fig. 4.

Nevertheless, IterLUNet consistently achieved impressive IoU and showed superiority in complex backgrounds over all the latest models. The proposed model detected the boundaries of the cracks more precisely, while the other models struggled to do so. Meanwhile, the best-performing model for each architecture with the lowest gap between training and validation dice-coefficient was selected to evaluate on an independent test dataset. As shown in Fig. 4, results indicate that pixel-wise prediction of cracks on completely independent test data is relatively low for all models. Every model faced

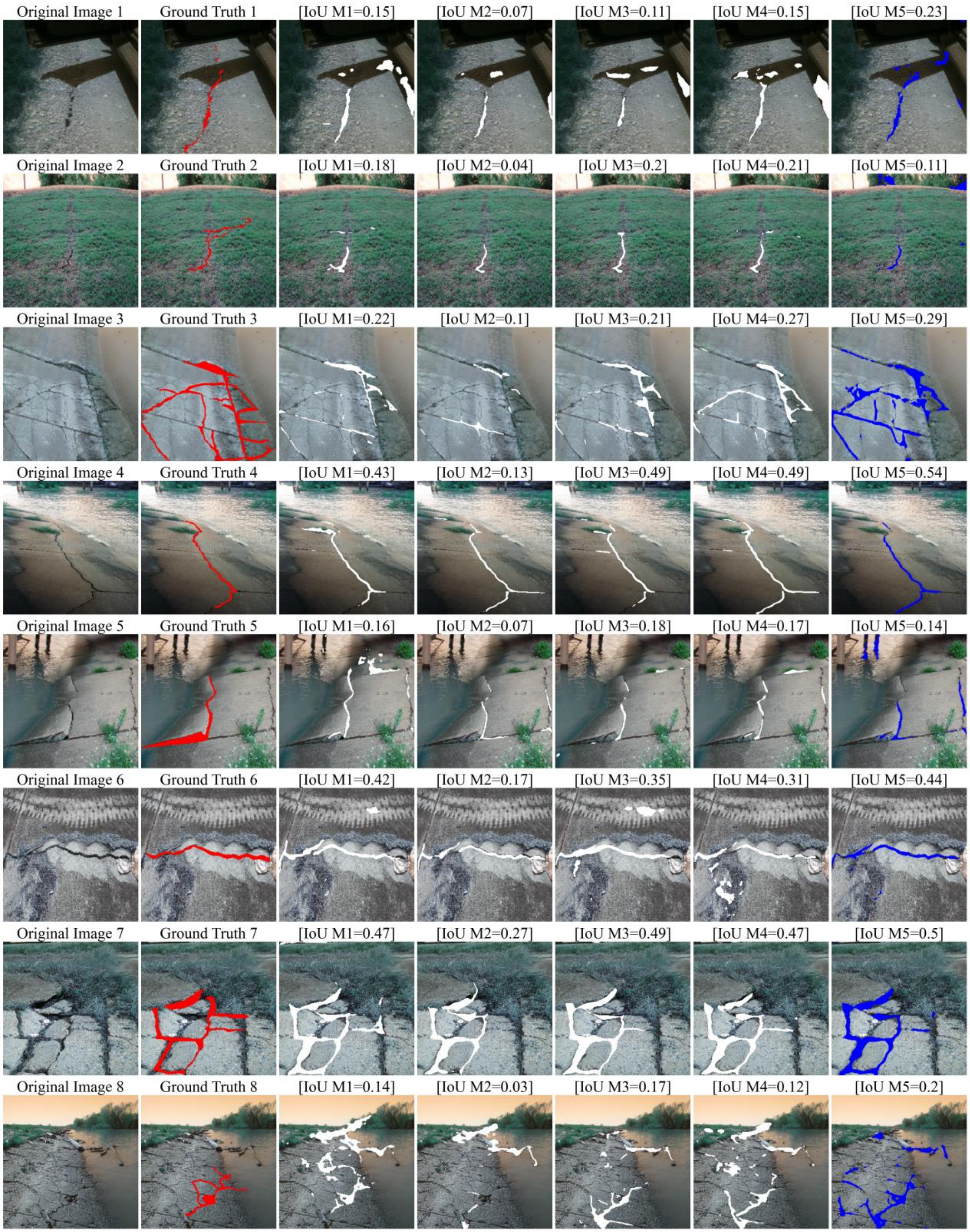


FIGURE 4. Examples from the independent levee crack test dataset from Experiment 1. Each colored column above represents a mask overlaid on the original image. White-colored masks are predicted segmentation masks for U-Net (M1), MultiResUnet (M2), Attention U-Net (M3), and UNet++ (M4). The red-colored mask is the ground truth, and the blue mask is the predicted segmentation mask by IterLUNet (M5).

difficulties locating crack pixels for some images. Given the limited proportions of the levee crack dataset, ten independent test images did not represent the training and validation images adequately. The challenge was also due to the difference in the distribution of crack regions, shapes, and background texture between the independent levee crack dataset and the training data. It requires additional original images with well-defined crack areas to yield a robust and high-performing model. This is the primary reason for performing augmentation and 10-Fold CV to show a need for a robust architecture that generalizes well on unseen levee crack images.

TABLE IV
EFFECTS OF LOSS FUNCTIONS ON U-NET, UNET++, AND ITERLUNET EVALUATED USING INDEPENDENT TEST DATASET

Models	Dice Coefficient (%)	Precision (%)	Recall (%)	IoU Crack (%)
M1 – A	41.29	55.99	30.84	28.16
M1 – B	37.33	54.72	25.45	25.43
M1 – C	36.95	**59.68**	25.04	24.98
M1 -D	**42.70**	55.27	**33.17**	**29.62**
M4 - A	38.61	54.88	29.63	26.00
M4 - B	**43.76**	**58.84**	33.15	**30.68**
M4 - C	40.87	56.42	31.39	27.74
M4 - D	41.80	49.02	**36.06**	28.42
M5 – A	43.80	50.52	37.33	29.94
M5 – B	41.86	51.82	34.46	28.08
M5 – C	**45.52**	**54.60**	38.94	31.04
M5 - D	44.81	46.37	**43.65**	**31.35**

Here, M1 – U-Net, M4 – UNet++, and M5 – IterLUnet are trained to minimize loss functions A, B, C, and D, representing BCE Loss, Dice Loss, BCE Dice Loss, and Focal Tversky Loss, respectively.

B. ANALYSIS OF LOSS FUNCTIONS

Levee crack dataset is high class imbalanced as crack pixels to be segmented are in a tiny percentage compared to the background pixels. From the analysis of the performance of models in 10-Fold CV and experiment 1, we observe that the models are learning better to classify non-crack pixels than crack pixels. Therefore, understanding the effect of objective function used during the training process appears crucial. In Table IV, A, B, C, and D are BCE loss, Dice loss, BCE Dice loss, and Focal Tversky loss, respectively. All models, U-Net (M1)- the base model, UNet++ (M4) - the best-performing model among U-Net-based models, and IterLUnet (M5) - the proposed model, are trained on sample data size to minimize these loss or objective functions. Fig. 5 illustrates a performance comparison between IterLUNet and UNet++ models trained with different loss functions. Fig. 5 also emphasizes that IterLUNet regularly performs well for the different experimental setups.

BCE loss being distribution-based log loss, measures the closeness of predicted pixels with the actual pixels and penalizes accordingly. However, all the other loss functions are region-based and directly try to maximize respective evaluation metrics. Table IV illustrates that models trained using the Focal Tversky loss function provide a better balance of precision and recall.

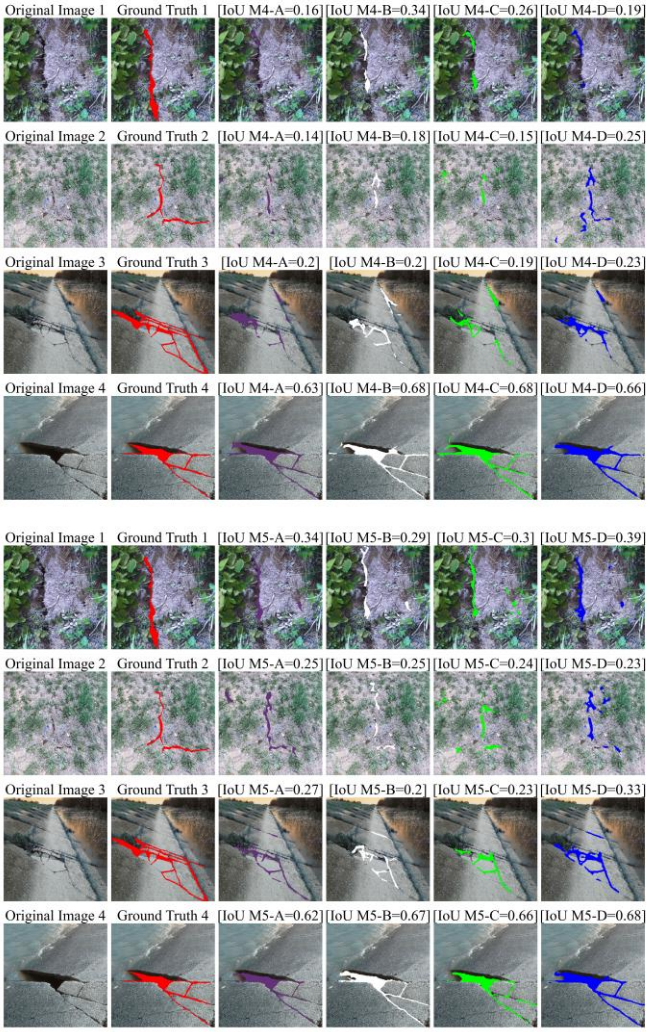


FIGURE 5. Comparison of the UNet++ (M4) and IterLUNet (M5) models trained with different loss functions and achieved IoU Crack for each example of the independent test image. Each column above represents a mask overlaid on the original image. The red-colored mask is the ground truth, and the blue mask is the predicted segmentation mask by model trained with Focal Tversky Loss. Purple-colored, white-colored, and green-colored masks are predicted segmentation masks for M4 and M5 trained with BCE loss, Dice loss, and BCE Dice loss, respectively.

C. COMPARATIVE PERFORMANCE ANALYSIS

Comparative performance analysis includes results and evaluation from experiment 2 and experiment 3. All architectures are trained on augmented images and evaluated with two independent test datasets. Table V shows metrics on the independent levee crack test datasets for experiment 2. The proposed model, IterLUNet, outperformed baseline architecture and the three latest best-performing models. We noticed that the increase in the number of original crack images and their ground truth had increased the performance of models. Fig. 6 depicts the proposed model's training and validation dice-loss and dice-coefficient curves over 80 epochs for experiment 2. With the trend of decreasing the gap between training and validation metrics, the complexity of the proposed model stands fit for the levee crack dataset. It also hints that since the dice loss value is still decreasing, increasing training epochs can lead to better results.

A public benchmark dataset to evaluate road crack detection system, DeepCrack [22], was used to assess trained models on the levee crack dataset. Table V and Table VI show the metrics, and Fig. 8 represents a few sample results on the independent test dataset from out of the domain. The differences in predicted segmentation masks overlaid on original images are shown in Fig. 8. The outcomes indicate that IterLUNet consistently predicts cracks and has a better detection ability on unseen images. It can also be observed from Fig. 8 that the models trained on the levee crack dataset are robust to predict crack regions on a highly textural background and blurred or unclear images. Together these results provide insights into boundary information and the shapes of cracks better predicted by the proposed architecture.

TABLE V
PERFORMANCE OF TRAINED MODELS OF EXPERIMENT 2 ON INDEPENDENT LEVEE CRACK TEST DATA AND DEEPCRACK BENCHMARK TEST DATASET

Models	mIoU (%)	IoU (%)	P (%)	R (%)	DC (%)
	Independent Levee Crack Test Data				
M1	61.76	28.19	61.89	38.48	41.62
M2	**63.48**	24.98	**64.42**	31.66	36.37
M3	61.92	28.02	61.61	39.68	41.72
M4	62.54	29.34	59.77	39.75	43.01
M5	62.22	**32.30**	59.81	**45.68**	**47.00**
	DeepCrack Benchmark Test Data				
M1	68.32	43.68	76.70	52.14	58.75
M2	68.20	39.53	**80.52**	43	53.35
M3	68.47	42.11	70.46	52.89	56.45
M4	68.20	45.15	77.17	54.23	60.04
M5	66.58	**49.13**	75.25	**61.69**	**64.14**

Here, P, R, and DC refer to Precision, Recall, and Dice Coefficient, respectively.

The most striking finding of this experiment was that IterLUNet is capable of separating the region of interest even from the rough background, observed in Fig. 8 and Fig. 9. Correspondingly, because of the inception-like module reinforced by SE-block, IterLBlock can focus on crack regions witnessed in an example rows 6, 8, and 9 of Fig. 7. Furthermore, the proposed model has higher balanced precision and recall avoiding false detection of true positives that may result in a devastating outcome. Since a model with a higher recall or true positive rate is crucial in an automatic crack detection system, such a model can potentially diminish the misidentification of crack pixels leading to an AI-based inspection solution.

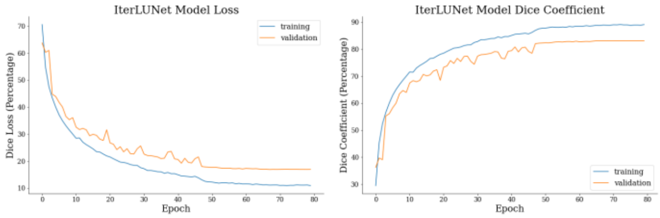


FIGURE 6. Dice-losses and dice-coefficients for IterLUNet at each epoch for training and validation dataset of experiment 2.

TABLE VI
PERFORMANCE OF TRAINED MODELS OF EXPERIMENT 3 ON INDEPENDENT LEVEE CRACK TEST DATA AND DEEPCRACK BENCHMARK TEST DATA

Models	mIoU (%)	IoU (%)	P (%)	R (%)	DC (%)
	Independent Levee Crack Test Data				
M1	60.28	28.38	60.05	37.18	41.14
M2	59.08	22.71	**62.08**	27.70	33.58
M3	**61.83**	30.35	58.41	45.10	43.03
M4	61.20	30.87	61.00	41.99	43.82
M5	60.15	**35.11**	49.39	**53.88**	**48.75**
	DeepCrack Benchmark Test Data				
M1	64.02	42.34	72.88	42.34	56.60
M2	64.62	37.42	**85.18**	37.36	51.09
M3	65.61	45.46	68.38	59.96	59.60
M4	65.75	45.93	74.09	55.11	60.34
M5	**68.14**	**57.34**	71.74	**74.62**	**70.85**

Here, P, R, and DC refer to Precision, Recall, and Dice Coefficient, respectively.

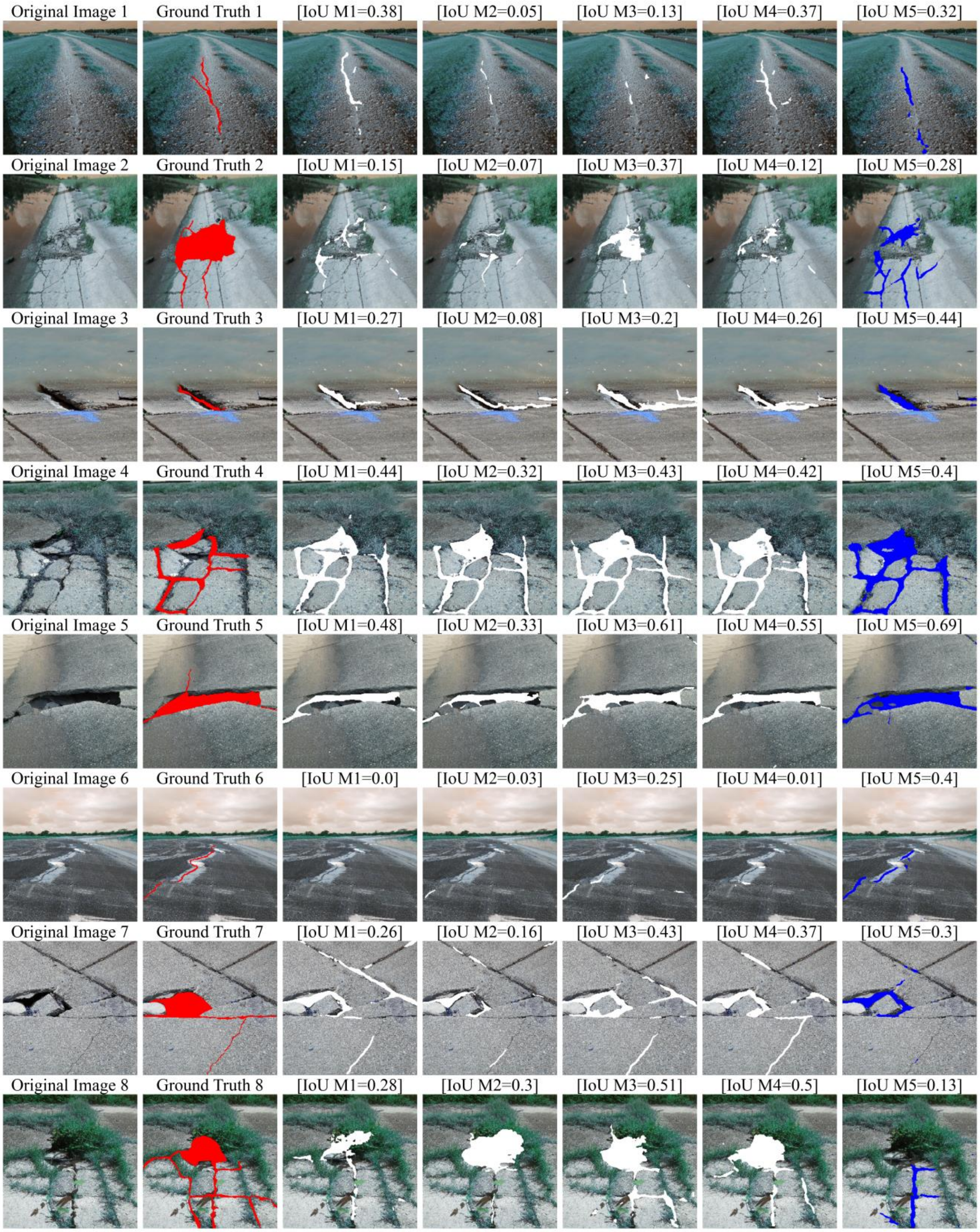


FIGURE 7. Examples from the independent levee crack test dataset of Experiment 3. Each column above represents a mask overlaid on the original image. White-colored masks are predicted segmentation masks for U-Net (M1), MultiResUnet (M2), Attention U-Net (M3), and UNet++ (M4). The red-colored mask is the ground truth, and the blue mask is the predicted segmentation mask by IterLUNet (M5).

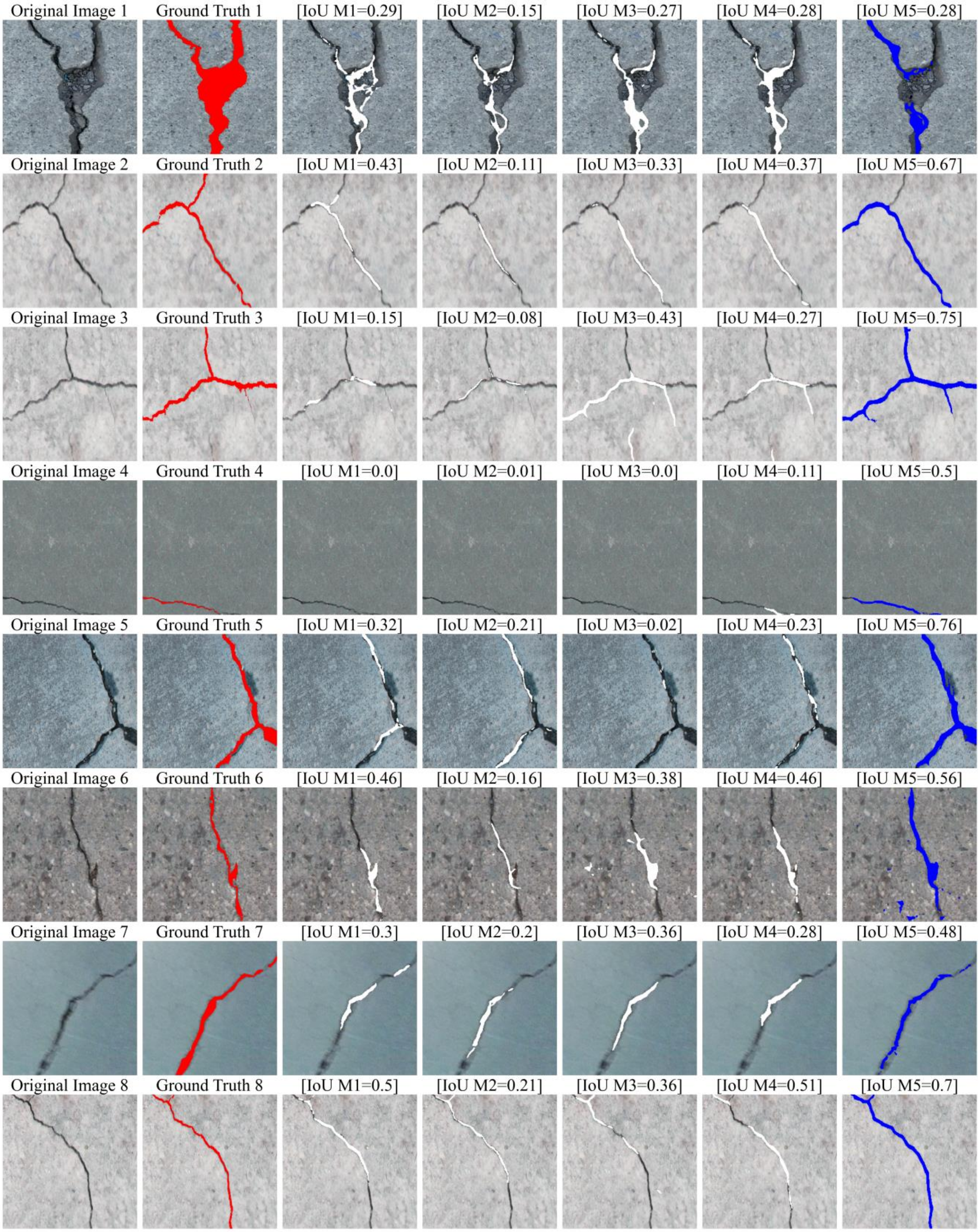


FIGURE 8. Examples from the independent DeepCrack Benchmark test dataset of Experiment 3. Each column above represents a mask overlaid on the original image. White-colored masks are predicted segmentation masks for U-Net (M1), MultiResUnet (M2), Attention U-Net (M3), and UNet++ (M4). The red-colored mask is the ground truth, and the blue mask is the predicted segmentation mask by IterLUNet (M5).

V. CONCLUSION

This paper experimentally established that expanding the path of an encoder-decoder architecture by connecting the decoder and bottleneck outputs back to the encoder increases model performance. We also demonstrated that an inception-like module, using only informative channel and spatial features through squeeze and excitation block variations, enhances the model's ability to focus on regions to detect. Therefore, we proposed an encoder-decoder-based fully convolutional neural network architecture, IterLUNet, to automatically detect cracks on the levee using a pixel-wise segmentation approach. Additionally, a benchmark dataset with levee crack images and corresponding ground truth segmentation masks was also introduced, which resulted in a substantial increase in Dice Coefficient and IoU, validating our hypotheses experimentally. The proposed architecture outperformed all the advanced architectures in terms of 10-Fold CV metrics and metrics on independent test datasets despite having nearly 63% fewer training parameters. Thus, the proposed concept helps improve overall IoU across semantic segmentation tasks. Availability of code and data here.

REFERENCES

[1] R. Szeliski, *Computer vision: algorithms and applications*: Springer Science & Business Media, 2010.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, pp. 2278-2324, 1998.

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431-3440.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 39, pp. 2481-2495, 2017.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234-241.

[6] N. Ibtehaz and M. S. Rahman, "MultiResUNet: Rethinking the U-Net architecture for multimodal biomedical image segmentation," *Neural Networks,* vol. 121, pp. 74-87, 2020.

[7] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, *et al.*, "Attention u-net: Learning where to look for the pancreas," *arXiv preprint arXiv:1804.03999,* 2018.

[8] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*, ed: Springer, 2018, pp. 3-11.

[9] R. A. de Albuquerque Nóbrega, J. Aanstoos, B. Gokaraju, M. Mahrooghy, L. Dabirru, and C. G. O'Hara, "Mapping weaknesses in the Mississippi river levee system using multi-temporal UAVSAR data," *Revista Brasileira de Cartografia,* vol. 65, 2013.

[10] O. Fernandes, R. Murphy, J. Adams, and D. Merrick, "Quantitative data analysis: CRASAR small unmanned aerial systems at hurricane Harvey," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2018, pp. 1-6.

[11] J. Cheng, W. Xiong, W. Chen, Y. Gu, and Y. Li, "Pixel-level crack detection using U-net," in *TENCON 2018-2018 IEEE Region 10 Conference*, 2018, pp. 0462-0466.

[12] R. Augustaukas and A. Lipnickas, "Pixel-wise road pavement defects detection using U-net deep neural network," in *2019 10th IEEE international conference on intelligent data acquisition and advanced computing systems: Technology and applications (IDAACS)*, 2019, pp. 468-471.

[13] J. Huyan, W. Li, S. Tighe, Z. Xu, and J. Zhai, "CrackU-net: A novel deep convolutional neural network for pixelwise pavement crack detection," *Structural Control and Health Monitoring,* vol. 27, p. e2551, 2020.

[14] M. D. Jenkins, T. A. Carr, M. I. Iglesias, T. Buggy, and G. Morison, "A deep convolutional neural network for semantic pixel-wise segmentation of road and pavement surface cracks," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2120-2124.

[15] S. L. Lau, E. K. Chong, X. Yang, and X. Wang, "Automated pavement crack segmentation using u-net-based convolutional neural network," *IEEE Access,* vol. 8, pp. 114892-114899, 2020.

[16] G. Li, B. Ma, S. He, X. Ren, and Q. Liu, "Automatic tunnel crack detection based on u-net and a convolutional neural network with alternately updated clique," *Sensors,* vol. 20, p. 717, 2020.

[17] S. Li and X. Zhao, "Automatic crack detection and measurement of concrete structure using convolutional encoder-decoder network," *IEEE Access,* vol. 8, pp. 134602-134618, 2020.

[18] Z. Liu, Y. Cao, Y. Wang, and W. Wang, "Computer vision-based concrete crack detection using U-net fully convolutional networks," *Automation in Construction,* vol. 104, pp. 129-139, 2019.

[19] Y. Pan, G. Zhang, and L. Zhang, "A spatial-channel hierarchical deep learning network for pixel-level automated crack detection," *Automation in Construction,* vol. 119, p. 103357, 2020.

[20] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, and X. Yang, "Automatic pixel-level crack detection and measurement using fully convolutional network," *Computer-Aided Civil and Infrastructure Engineering,* vol. 33, pp. 1090-1109, 2018.

[21] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *2016 IEEE international conference on image processing (ICIP)*, 2016, pp. 3708-3712.

[22] Y. Liu, J. Yao, X. Lu, R. Xie, and L. Li, "DeepCrack: A deep hierarchical feature learning architecture for crack segmentation," *Neurocomputing,* vol. 338, pp. 139-153, 2019.

[23] A. G. Roy, N. Navab, and C. Wachinger, "Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks," in *International conference on medical image computing and computer-assisted intervention*, 2018, pp. 421-429.

[24] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, and S. Wang, "Deepcrack: Learning hierarchical convolutional features for crack detection," *IEEE Transactions on Image Processing,* vol. 28, pp. 1498-1512, 2018.

[25] A. Kuchi, M. Panta, M. T. Hoque, M. Abdelguerfi, and M. C. Flanagin, "A machine learning approach to detecting cracks in levees and floodwalls," *Remote Sensing Applications: Society and Environment,* vol. 22, p. 100513, 2021.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818-2826.

[27] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132-7141.

[28] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289,* 2015.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448-456.

[30] M. Panta, M. T. Hoque, M. Abdelguerfi, and M. C. Flanagin, "Pixel-Level Crack Detection in Levee Systems: A Comparative Study," in *2022 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2022.

[31] A. Dutta, A. Gupta, and A. Zissermann, "VGG image annotator (VIA)," *URL:* https://www.robots.ox.ac.uk/~vgg/software/via/, 2016.

[32] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *Information*, vol. 11, p. 125, 2020.

[33] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, 2020, pp. 1-7.

[34] N. Abraham and N. M. Khan, "A novel focal tversky loss function with improved attention u-net for lesion segmentation," in *2019 IEEE 16th international symposium on biomedical imaging (ISBI 2019),* 2019, pp. 683-687.

[35] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[36] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510-4520.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026-1034.

MANISHA PANTA received an M.S. degree in Computer Science from the University of New Orleans, New Orleans, LA, USA, in 2019. She is currently a Ph.D. Student in the Department of Computer Science at the University of New Orleans, New Orleans, LA, USA. Her research concentration is Deep Learning for Computer Vision. She is interested in Real-time Scene Understanding and Visual Question Answering.

MD TAMJIDUL HOQUE received a Ph.D. in Information Technology from Monash University, Melbourne, VIC, Australia, in 2008. He is currently an Associate Professor with the Computer Science Department, University of New Orleans, New Orleans, LA, USA. From 2011 to 2012, he was a Post-Doctoral Fellow with Indiana University–Purdue University Indianapolis, Indianapolis, IN, USA. From 2007 to 2011, he was a Research Fellow with Griffith University, Brisbane, QLD, Australia. His current research interests include deep/machine learning, evolutionary computation, and artificial intelligence, applying them to complex optimization problems, especially for bioinformatics problems such as protein structure-prediction, disorder predictor, and energy function.

MAHDI ABDELGUERFI is a professor and chairperson of the University of New Orleans Computer Science Department. He is the founder and executive director of the Canizaro Livingston Gulf States Center for Environmental Informatics (GulfSCEI).

MAIK FALANAGIN graduated from MIT with a Masters of Engineering in Electrical Engineering and Computer Science and UNO with a Ph.D. in Engineering and Applied Sciences. He is Louisiana's first professionally-licensed software engineer. Maik has worked for the Corps of Engineers since 2001, including his time as a contractor. He is the GIS Lead for the New Orleans District and Technical Lead of their Software Development team.