Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

Article

Deep Learning Based Feature Silencing for Accurate Concrete Crack Detection

Umme Hafsa Billah [1,‡], Hung Manh La [1,‡] and Alireza Tavakkoli [2,†,‡,*]

[1] University of Nevada, Reno; ubillah@nevada.unr.edu
[2] University of Nevada, Reno; {hla,tavakkol}@unr.edu
* Correspondence: tavakkol@unr.edu; Tel.: +1-775-682-8426 (A.T.)
† Current address: 1664 N. Virginia St. MS 171, Reno, NV 89557, USA
‡ These authors contributed equally to this work.

Abstract: An autonomous concrete crack inspection system is necessary for preventing hazardous incidents arising from deteriorated concrete surfaces. In this paper, we represent a concrete crack detection framework to aid the process of automated inspection. The proposed approach employs a deep convolutional neural network architecture for crack segmentation from concrete image. The proposed network alleviates the effect of gradient vanishing problem present in deep neural network architectures. A feature silencing module is incorporated in the crack detection framework, for eliminating unnecessary feature maps from the network. The overall performance of the network significantly improves as a result. Experimental results support the benefit of incorporating feature silencing within a convolutional neural network architecture for improving the network's robustness, sensitivity, and specificity. An added benefit of the proposed architecture is its ability to accommodate for the trade-off between specificity (positive class detection accuracy) and sensitivity (negative class detection accuracy) with respect to the target application. Furthermore, the proposed framework achieves a high precision rate and processing time than crack detection architectures present in literature.

Keywords: Convolutional Neural Network; Encoder-Decoder Architecture; Semantic Segmentation; Feature Silencing; Crack Detection

1. Introduction

Proper inspection and maintenance of civil infrastructures is essential to avoid man-made disasters. Manual inspection has been employed for a long time using heavy and large equipment by civil engineers to assess structural defects. The time-consuming and labor-intensive nature of this type of inspection system causes traffic disruption. Furthermore, the manual assessment procedure is perilous for humans in inaccessible regions of civil infrastructures such as under bridge decks and underwater beams. On the contrary, an autonomous civil infrastructure inspection system monitors structural health continuously with least human intervention. Such an autonomous robotic system is able to capture data for surface-level visual inspection and defect identification of civil infrastructures [1–3]. In this paper, we have proposed a defect (crack) detection architecture from concrete surface images to aid the autonomous crack inspection process.

1.1. Literature Review

A number of different techniques for crack detection have been proposed in the recent past. Several image processing techniques, such as thresholding [4–8], morphological operations [9–14] and edge detection algorithms [15–18], or crack detection in civil infrastructure evaluation have been reported in the earlier literature. The defected areas of concrete surface exhibit two important properties. First, these defects do not have any definite shape or pattern. As a result, these defects do not represent any distinct feature properties cannot be accurately detected using statistical estimation.

Second, they appear in a very small portion of concrete images. The rare occurrence and aberrant shape of concrete defects such as cracks closely resemble the properties of anomalies. Therefore, concrete crack detection can be contemplated as an anomaly detection problem, in which anomaly detection techniques [19,20] are applicable. The aforementioned techniques extract both local and global features for effective concrete crack detection.

Although these techniques are computationally inexpensive, they have several disadvantages. These techniques generate unnecessary feature points because of the highly textured nature of concrete images. Removal of these feature points (using median or mean filtering) eliminates significant defect locations. The precondition of selecting a unique set of parameters for different image set is a challenging task for these image processing methods. Apart from this, the feature properties of cracks and edges are different from each other. The pixel connectivity of edges is smoother than cracks. Unlike edges, crack pixels are not continuously connected. As a result, traditional image processing techniques fail in crack classification [3,16,21]. Examples of a crack detection results using traditional image processing methods such as Difference of Gaussians (DoG) and Canny edge detection, on a clean concrete image is shown in Figure 1(b,c). These crack locations represent crack like appearance in the concrete image. As shown from these figures, traditional image processing techniques detect sharp changes in color and intensity of the image, which include other color/contrast transitions along with crack pixels.

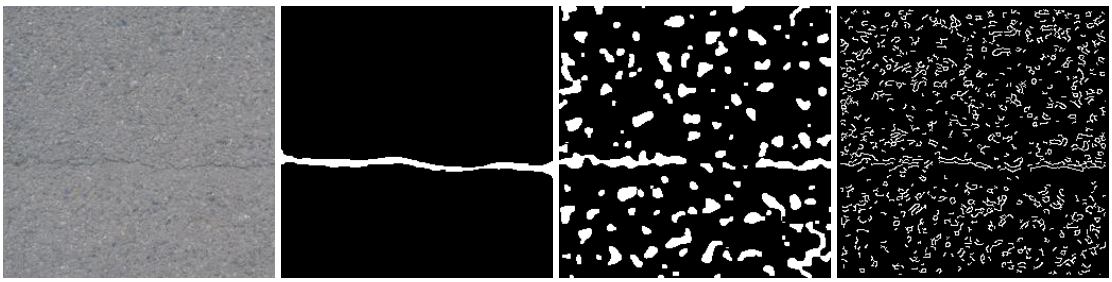

Figure 1. Challenges in crack detection using image processing techniques: Although cracks are edge-like features, they usually represent defects not associated with other edges. (a) Original image, (b) cracked pixels ground truth, (c) sharp transition changes using Difference of Gaussians (DoG), (c) Canny edge detection. As it can be seen edge detection produces other sharp color/contrast transitions along with crack pixels.

Moreover, concrete defect images are affected by environmental non-uniformity such as illumination, noise, shading and many more. The image processing methods are highly sensitive to these environmental factors [3,21].

Machine learning architectures were employed for crack detection to achieve robustness toward crack detection. Machine learning architectures such as Support Vector Machines (SVM) [9,17,22], Adaboost [17] and Multi-Layer Perceptron (MLP) [23] networks, were used preliminary for concrete crack identification. Later, a combination of machine learning and image processing techniques were employed to improve the defect detection result [24,25]. Although the accuracy of defect detection improves with these methods, they inherit the complexity of appropriate parameter selection.

Since artificial neural networks (ANN) update the parameter weights autonomously, they are nominally affected by the aforementioned parameter selection problem. Therefore, many ANN structures were employed for concrete distress identification [26,27]. The astounding performance of convolutional neural networks (CNNs) [28] in many image classification and object recognition applications, aspired the researchers to employ CNN architectures for concrete defect identification. CNN architectures closely simulate the functionality of the biological visual cortex by representing cortical areas as individual layers. These layers extract unique feature sets for a specific data-set and learn their statistical properties. Some of the CNN architectures employed for concrete crack

identification applications are ResNet [29–31], AlexNet [32,33] and VGG-Net [34]. Apart from this, CNN architectures were designed specifically for concrete crack identification purpose [3,21,35,36].

The aforementioned CNN architectures consider crack identification as an image classification problem. Although these techniques are highly efficient for crack feature extraction, they need to utilize image processing algorithms for identifying the exact location of crack [3,21]. The image processing technique used for defect localization suffers from the parameter selection problem. Moreover, the computational complexity of deep networks stems in the gradient vanishing problem [30], resulting in a significant performance drop.

Semantic segmentation architectures alleviate the problem of localizing and classifying concrete crack pixels at the same time. There exist several semantic segmentation architectures for scene parsing, object instance segmentation, and many other applications. Among them, FCN [37], Mask-RCNN [38], UNet [39] and SegNet [40] have achieved significant performance in the field of segmentation. The object instance segmentation applications widely employ Mask-RCNN architecture. Scene segmentation and image restoration applications employ the SegNet, UNet, and FCN architectures. These architectures learn important feature attributes of the regions through a series of encoding and decoding operations. SegNet, UNet, and FCN architectures are widely employed for scene segmentation and image regeneration applications. These architectures perform a series of encoding and decoding operations for segmentation. The SegNet architecture outperformed UNet and FCN significantly in terms of accuracy, computational complexity, and memory usage.

On the other hand, the ResNet [32] represented the property of extracting efficient features in complex image classification applications. The ResNet architecture can be employed as an encoder decoder manner for segmentation purpose. This approach involves a huge number of parameters, which results in the gradient vanishing problem. Apart from this, the fully connected layer used in ResNet, Mask-RCNN, UNet, FCN architectures increase the computational complexity. The decoder network introduced by the SegNet architecture eliminates the need for using the computationally expensive fully connected layers. Furthermore, excessive computations can loose important crack feature attributes in deeper layers (through max-pooling). The effectiveness of the aforementioned architectures are greatly hampered in that case.

To alleviate the aforementioned problem of crack segmentation, a very few encoder-decoder architectures were proposed in [41–47]. These architectures represented more precise and robust performance than the state-of-the-art semantic segmentation networks.

There are certain drawbacks of encoder-decoder architectures (both crack detection and semantic segmentation) that impede crack detection performance. First, the computations performed by these architectures are twice of a regular CNN architecture. These sheer amount of computations enhances the network's sensitivity to gradient vanishing problem as well as environmental non-uniformity. Extracting discernible feature sets from a highly imbalanced crack detection data-set is a challenging task. Therefore, in this paper, we proposed a CNN architecture addressing the aforementioned drawbacks of existing literature.

1.2. Contributions

The main contributions of the proposed architecture are represented as follows:

- In this paper, we propose an efficient framework for class imbalanced data-sets such as crack detection. The proposed framework incorporate an encoder-decoder network architecture for reducing the effect of gradient vanishing problem. The network architecture (referred to as ANet-FSM) is elaborately discussed in section 2. Additionally, our investigation reveals that using this specific type of architecture is effective in applications suffering from high degrees of class imbalance –crack identification (non-crack pixels are much more than crack pixels). Empirical evidence supporting the propositions in this work are represented in section 2.
- The second main contribution of this study, is the incorporation of a feature silencing module (FSM). The FSM module alleviates the sensitivity of the deep architectures toward feature maps

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

that do not contribute effectively to the optimization of the network loss. The incorporation of the FSM with the proposed CNN architecture significantly reduced the false classification rate of the concrete crack detection process. We have explained the functionality of the FSM elaborately in section 2. The efficiency of the FSM on concrete crack identification data-set is represented in section 3.

The classification-based CNN architectures for concrete crack identification classifies a sub-block of an image as crack or non-crack. For example, the network architectures proposed by [3,21,36] divide a large image into smaller sub-blocks of size 256×256. Each of these sub-blocks is a combination of 65536 crack and non-crack pixels. If a crack block contains only 100 crack pixels, the remaining pixels are falsely classified (empirical evidence is shown in section 3). Although the false classification rate was alleviated with architectures based on encoder-decoder models proposed by [41–47], these methods significantly suffer from gradient vanishing problem. Additionally, these methods are highly sensitive to environmental non-uniformity. The proposed architecture in this paper significantly reduces the false classification rate of image classification methods as well as alleviates the drawbacks of encoder-decoder based architectures. In the following sections, we elaborately discuss the proposed architecture and provide a comparative analysis of different existing architecture for crack detection.

2. Methodology

In this section, we explain the modules of our proposed architecture. To represent our network architecture comprehensively we briefly discuss the functionality of a CNN and encoder-decoder based architecture in the following sections. Later, different modules of the proposed architecture are explained elaborately.

2.1. Convolutional Neural Network Architecture

CNN [28] architectures are a composition of various connected layers such as input, convolution, pooling, activation and output layers. Some auxiliary layers such as batch normalization and drop out layer are also associated according to application purpose. This arrangement of various layers and their functionality extracts discernible feature sets for each object on CNN. Among all the layers of CNN, the convolutional layer is primarily responsible for the salient feature extraction of a specific object. Each convolutional layer performs multiple convolution operations using receptive fields (kernels) of fixed size and variable weights. A unique feature set is generated as an outcome of each convolution operation. The feature maps generated by all the different weighted kernels from a feature space. The feature space generated by a convolutional layer represents complex non-linear relationships among the input set and output label. To obtain a linear mapping from these non-linear relationships, an activation function is used at the end of the convolutional layers. Moreover, the output feature space of a convolutional layer is sensitive to a specific feature position. As a result, a down-sampling operation (known as pooling) is performed after each convolutional layer to obtain a position invariant feature space. The pooling operations preserves strong feature responses and eliminates the weak ones using a fixed pooling window size. CNN architectures generate a robust and position invariant feature space through several convolution and pooling operations. A soft-max layer is added at the end of the CNN to assign a normalized probabilistic distribution to the feature responses.

2.2. Encoder Decoder Architecture

Convolutional neural networks have achieved significant performance in applications ranging from object recognition to gesture and pose recognition. In addition to classification and recognition, semantic segmentation and image reconstruction applications employ encoder-decoder based CNN architectures (SegNet [40], UNet [39], FCN [37]). These architectures encode the feature space into a lower dimension through a series of encoders. This feature space is decoded into a higher dimension

through corresponding decoders. Salient feature attributes of each pixel of an input data set are learned through these encoding and decoding operations.

For crack identification purpose a number of different encoder-decoder based architectures were proposed recently such as InspectionNet [41], DeepCrack [43], SDDNet [44] and SegNet-SO [42]. Though these architectures represent promising results on crack classification, they have some drawbacks. Firstly, these architectures extract crack feature attributes using multiple 3×3 convolution operation. Multiple convolution operations of small receptive field extract both global and local feature attributes of the input image as reported in [34]. While going through multiple convolution operations in deeper layers, the gradient stability of a network is lost. The network stops updating the weights of these parameters, which results in the wrong classification of crack pixels. As a result, the network attains a very high loss value. An example of this phenomenon is represented in figure 2. In this figure, the loss of two network architectures involving two different convolution operation (3×3 and 7×7) is plotted. We have extracted a 20 epoch window from training for better visualization of the loss. It is evident from the figure that, for multiple 3×3 convolution operation the loss is not stable between any two epochs. The loss doesn't change gradually for this convolution operation. For example, the loss in epoch $N + 1$ jumps to a higher loss in comparison to the loss in epoch N. This phenomenon is visible through the entire 20 epoch window and represents the high gradient instability of this network. On the other hand, for the network with 7×7 kernel size, the loss changes gradually. This represents the stable gradients of the network.

Figure 2. Training loss of two different type convolution networks. The training loss was extracted from a 20 epoch window while training. N depicts the start of the window and the loss were plotted for each epoch. Training loss of convolution network with two different kernel size (3×3 and 7×7) is plotted.

The loss value of both the network in figure 2 lies within $\mu \pm \sigma$, where μ is the mean value of the loss in current epoch window and σ is the standard deviation. For the network using 3×3 convolution kernel the loss lies within 0.002 ± 0.012, where $\mu = 0.02$ and $\sigma = 0.012$. The maximum and minimum loss value of this network is 0.05 and 0.002 respectively. On the other hand for the network with 7×7 convolution kernel the loss lies within 0.006 ± 0.0007, where $\mu = 0.0006$ and $\sigma = 0.0007$. The maximum and minimum loss of this network is 0.0072 and 0.005 respectively. These statistics represent that

the loss generalization of a single large convolution kernel (e.g., (7×7)) is better than multiple small kernels (e.g., multiple 3×3 kernel). The loss value of the network with multiple small convolution kernels is seven times (approximately) higher than the network with a large convolution kernel. Due to gradient instability, the network didn't update many parameter weights, resulting in a very high loss value of 0.05. Furthermore, the high standard deviation also represents the highly unstable gradient values or the gradient vanishing problem.

Secondly, the concrete defect identification data sets vary significantly from state-of-the-art image classification databases because crack pixels show anomalous behavior (aberrant patterns and shapes) in comparison to healthy concrete pixels. Additionally, these anomalous pixels appear only in a small portion ($2 - 10\%$ approximately) of a concrete image (known as class imbalance problem). Therefore, the relationship between crack and non-crack pixels can be extracted more efficiently if the convolution operation is performed concerning a large neighborhood. An example of the aforementioned phenomenon is represented in figure 3. The image in this figure is extracted from the Crack260 [48] Data-Set. The enlarged neighborhood of a crack location is shown in this figure 3. It is evident that, with the small spatial neighborhood, we can extract the properties of crack only. Though the global and local features can be extracted using multiple spatial neighborhoods, it is crucial to extract the anomalous relationship between crack and non-crack pixels. This relationship can be extracted using a relatively large neighborhood [44].

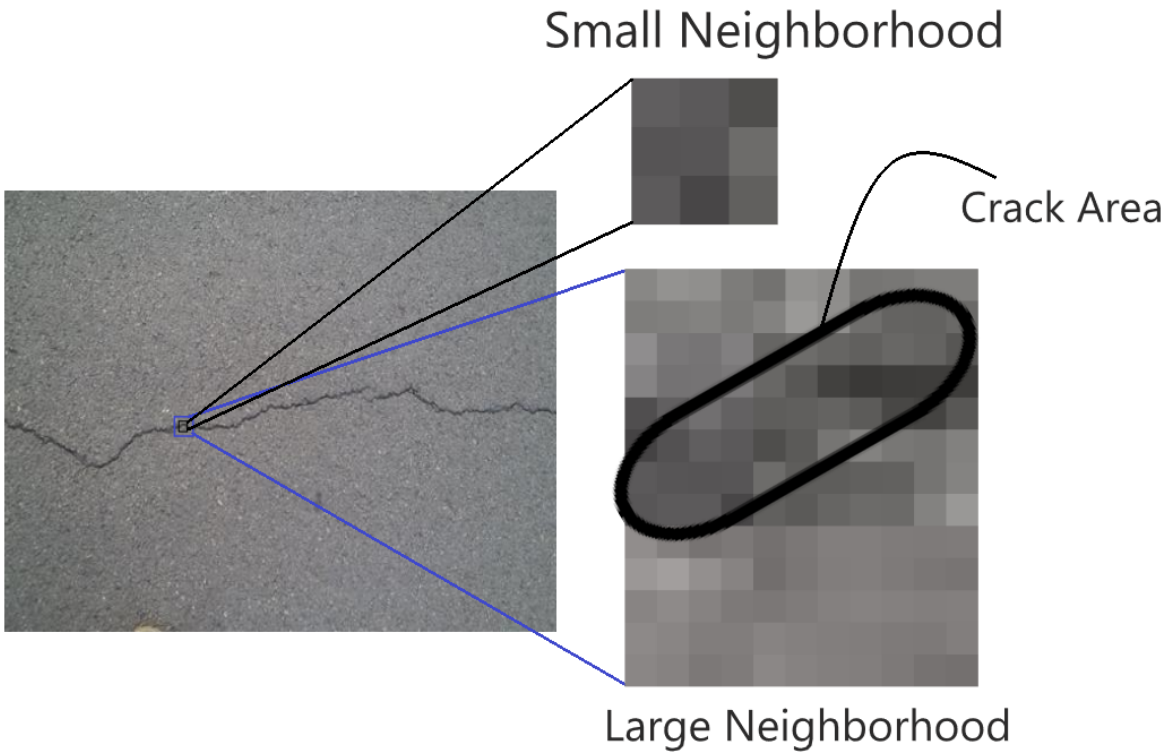

Figure 3. The comparison of two neighborhood of crack location from a small neighborhood and large neighborhood. Since crack pixels occur a very small amount of time, the small neighborhood only contains crack pixels. On the other hand, in the large neighborhood, the statistical relationship between crack and non-crack pixels can be captured more appropriately.

For concrete crack classification, the deep crack [43] network architecture achieved a significant performance using multiple receptive fields of size 3×3. This approach has the advantage of extracting efficient local and global features of crack pixels. As discussed earlier in figure 2, the gradient vanishing problem is very prominent in these types of approaches. To address this issue another deep architecture was developed for crack detection namely SDDNet [44]. This architecture uses a combination of different convolution operations such as separable, atrous, and point-wise convolutions. They used

a large receptive field for extracting the crack and non-crack pixel relationship. In this paper, we proposed a crack detection framework with only one type of convolution operation. We used a 7×7 convolution kernel with only one convolution layer in each encoding operation. The large receptive field is very crucial for extracting the imbalanced relationship between crack and non-crack pixels as shown in figure 3.

2.3. ANet-FSM Architecture

The proposed ANet-FSM architecture is comprised of an encoder-decoder module, a feature silencing module (FSM) and a concatenation module. An overview of the whole architecture is shown in figure 4. Important crack feature attributes are learned through the encoding and decoding operation. The FSM eliminates weak feature maps generated from the encoder module and passes the strong feature maps to the concatenation module. These feature maps are up-sampled and concatenated together in the concatenation module. The up-sampled feature maps along with the output feature map of the decoder module are passed to a 1×1 convolution and soft-max layer. The encoder module in figure 5 is composed of five encoder layers.

Each encoder layer assembles a convolutional layer and a max-pooling layer. Two auxiliary layers (Batch Normalization and ReLU) are added at the end of each convolutional layer. The convolution layers in each encoder use $4, 8, 16, 32$ and 64 kernels of size 7×7, respectively. The max-pooling operation down-samples the feature space using a 2×2 pooling window. Additionally, this operation saves the pooling indices in memory to be used in the up-sampling operation in the decoder module.

A representation of the decoder layers for each corresponding encoder is represented in figure 5. Each decoder layer performs a bi-linear up-sampling operation on its input feature space using the pooling indices saved during max-pooling operation. A convolutional layer is employed at the end of each up-sampling operation to decode the feature space of its corresponding encoder.

A spatial neighborhood of size 7×7 is used for each convolution operation throughout the encoding and decoding operation. As discussed earlier in section 2.2, this neighborhood is also efficient for eliminating the gradient vanishing problem of state-of-the-art crack detection and semantic segmentation architectures. Therefore, each encoder layer in this architecture is comprised of only one convolutional layer.

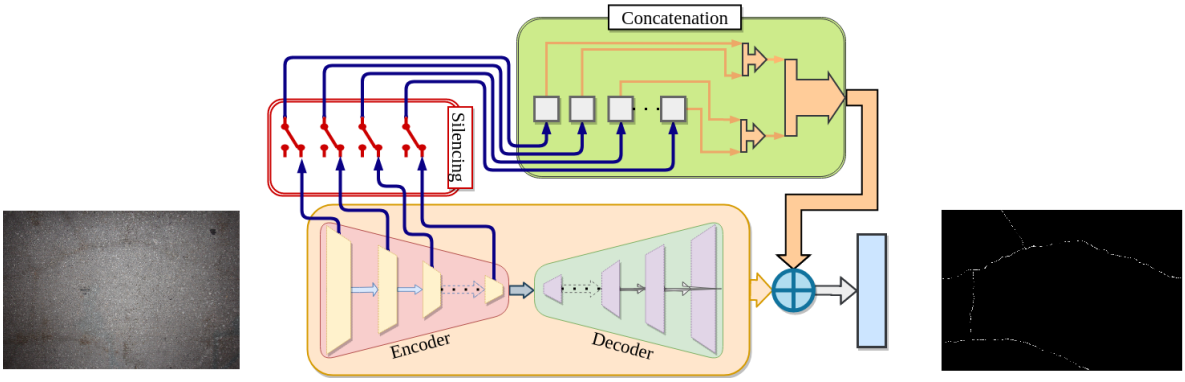

Figure 4. The proposed network architecture overview.

On the other hand, the max-pooling operation eliminates some feature responses in the course of generating position invariant feature space. This feature loss is nominal when enough instances of different classes are present. The concrete defect data-sets are highly imbalanced due to the low occurrence of defected pixels. Removal of these feature responses significantly affects the performance of the network. A transient solution to this problem is to up-sample the feature spaces into the original dimension after each encoding operation [42].

Empirical analysis performed on the feature spaces after each encoding operation reveals that all the features in the feature space do not contribute equally to the crack pixel identification. An example

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 July 2020 doi:10.20944/preprints202007.0474.v1

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

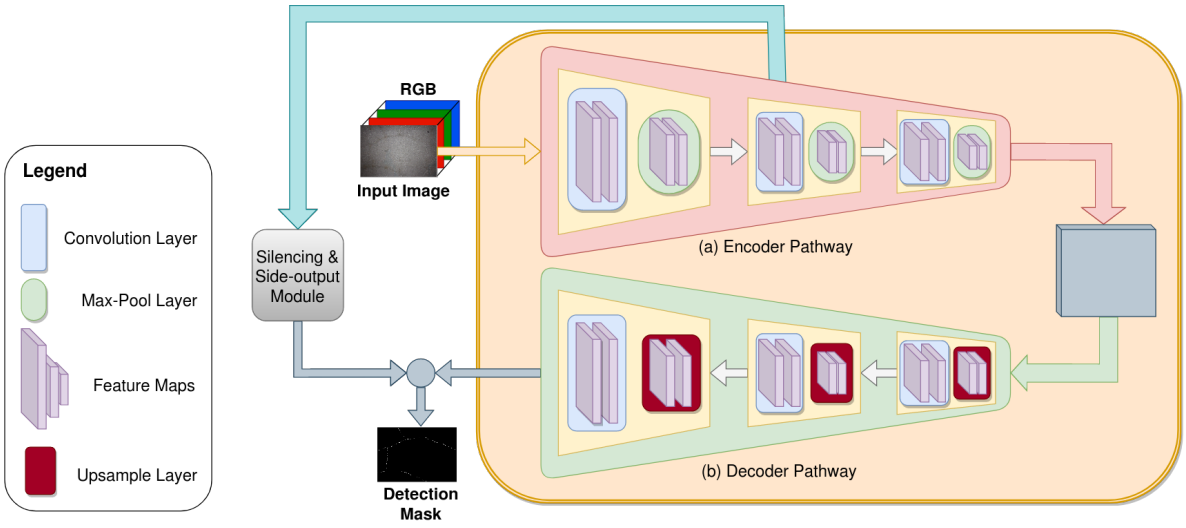

Figure 5. Encoder and Decoder Module of ANet-FSM architecture. Each encoder performs a 7×7 convolution and max-pooling operation. The decoders up-sample this low dimensional feature space into upper dimension using bi-linear interpolation. The feature space decoding is performed by the convolution operation in each decoder.

of a feature space extracted after a single encoding operation is shown in figure 6. It is evident from the figure, the feature maps in figure 6(1-3)(a), (b), (c), (f), (h) represent very strong crack features. These feature maps contribute significantly to final crack pixel identification. The remaining feature maps in figure 6(1-2)(d), (e), and (3)(g) are responsible for overlapping the salience property of the feature space Moreover, the weight matrix of these feature space contains gradient values close to zero, which results in false classification. To enhance the precision of crack classification, the FSM module in ANet-FSM architecture eliminates these feature maps from the network feature space. A graphical representation of the FSM module is shown in figure 7.

The FSM module in figure 7 uses equation 1 for extracting significantly contributing feature maps from each encoder feature space. Equation 1 is defined as follows:

$$F_p = \bigcup_{f_i \in F}^{i=1 \rightarrow N} f_i \leq th$$
$$F_s = F \setminus F_p \tag{1}$$

where F_p is the pruned feature space, F is a feature space from an encoder, f_i is the i^{th} feature map in F, th is a threshold value, F_s is the selected features for passing to the concatenation module and N is the number of feature maps in feature space F.

The encoding operation generates some significantly contributing feature maps for crack detection. Using equation 1, we extract these feature maps for up-sampling. The rest of the feature maps are considered weak by this equation. As a result of gradient vanishing, these feature maps have values close to zero. Hence, in equation 1, we select the feature maps for pruning (F_p) having values less than a threshold th. A new feature space F_s is formed by eliminating the feature spaces in F_p from the original feature map F.

The decoder layer in ANet-FSM uses the pooling indices from the max-pooling operation of the corresponding encoder for performing up-sampling (introduced by [40]). The decoding operations are directly dependant on the feature space of the corresponding encoders. Propagating the pruned feature space from one encoder layer to another encoder layer effects the pixel connectivity of the cracks as well as the precision of the network. The feature spaces in encoder and decoder propagate without any pruning operation. Additionally, the pruned feature space (F_s) is passed to the concatenation module shown in 4. This concatenation module performs bi-linear up-sampling on the pruned feature

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 July 2020 doi:10.20944/preprints202007.0474.v1

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

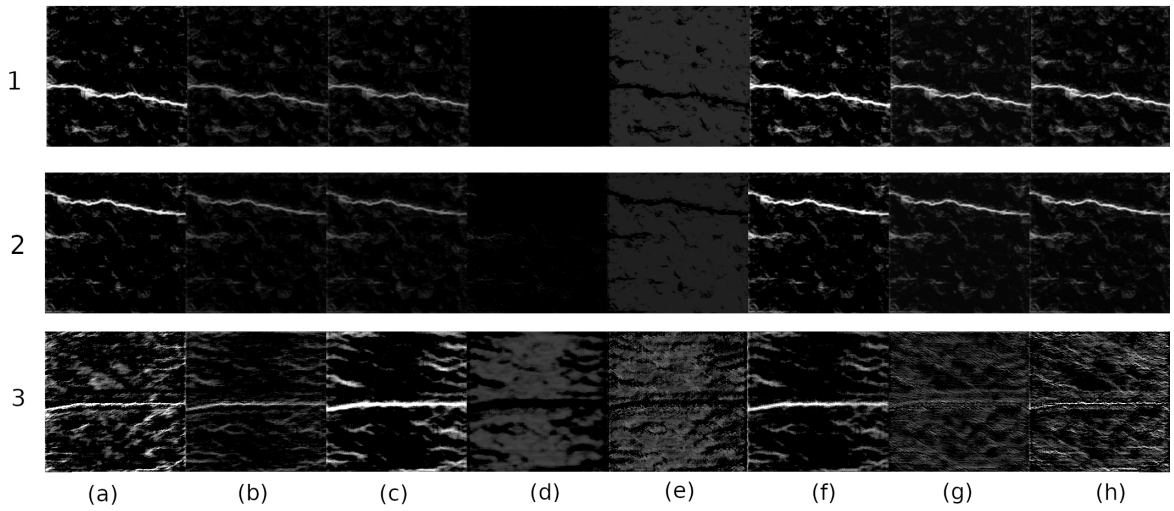

Figure 6. An example of a feature space of different crack image during an encoding operation. The weak feature responses generate weight matrix close to zero (approximately). This feature maps are eliminated using equation 1

.

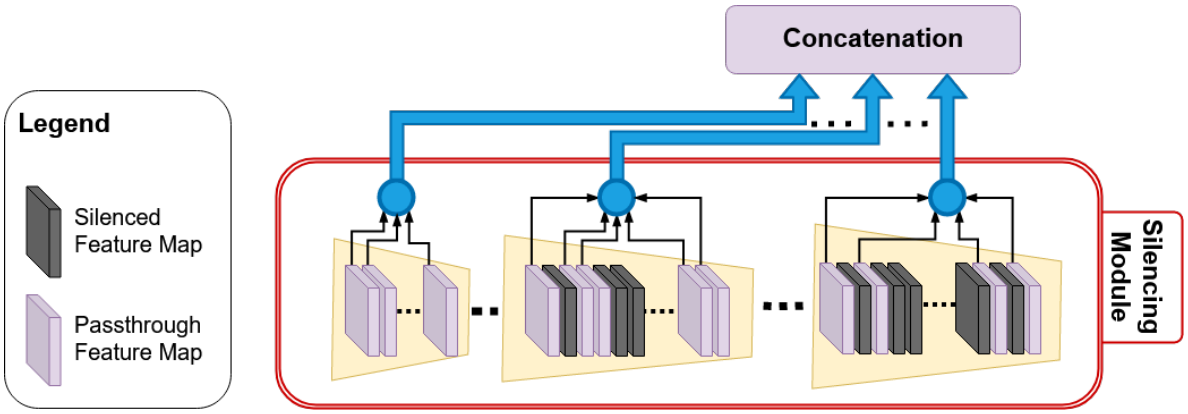

Figure 7. FSM of the proposed architecture: the silenced feature maps are represented with black colors.

space from each encoder (except the first encoder). These up-sampled pruned feature spaces are concatenated with the feature space of the final decoder. Since the crack classification is a one-class classification problem, we use a 1×1 convolution operation at the end of the concatenation operation. This convolution operation helps to estimate the crack location from the concatenated feature space.

An example of eliminated feature maps from an instance of the Illinois Bridge Data-set is represented in table 1. To represent the effectiveness of the FSM module in reducing the complexity of a network, we have introduced three measures such as feature silencing rate (FSR), feature space size before pruning (FSS_b) and feature space size after pruning FSS_a. Since there is a scarcity of standardization metrics of feature space pruning, we have incorporated the above three measures for evaluation only. The FSR in table 1 is defined in equation 2.

$$FSR = \frac{B_s - A_s}{B_s} \tag{2}$$

where B_s represents the number of features before silencing and A_s represents the number of features after silencing. Additionally, we have analyzed the feature space size before and after silencing using equation 3.

$$\begin{aligned} FSS_b &= x \times y \times B_s \\ FSS_a &= x \times y \times A_s \end{aligned} \tag{3}$$

where x is height of an image, y is width of an image, FSS_b is the the feature space size before silencing and FSS_a is the feature space size after silencing. In table 1, we have calculated the value of equation 3 for a 256×256 image. Since, each encoder operation reduces the image size to half, the value of x and y is reduced to half. Therefore, image size is $(256, 256)$, $(128, 128)$, $(64, 64)$, $(32, 32)$, $(16, 16)$ for Encoder1, Encoder2, Encoder3, Encoder4 and Encoder5 respectively in table 1.

Table 1. FSM from an instance of the Illinois Bridge Data-set: the numbers in FSM represent the i^{th} eliminated feature. The intial $(x, y) = (256, 256)$ for FSS_a and FSS_b. Once the image is passed through multiple encoders, the image size is reduced to half of the original. The (x, y) value for Encoder2, Encoder3, Encoder4 and Encoder5 are $(128, 128)$, $(64, 64)$, $(32, 32)$ and $(16, 16)$.

Encoder Number	Features Silencing Rate, FSR	Feature Space Size for 256x256 image		No. of Features, Fs		Silenced Feature Set by FSM
		Before Silencing, FSS_b	After Silencing, FSS_a	Before Silencing, Bs	After Silencing, As	
Encoder1	0.0%	262,144	262,144	4	0	♦
Encoder2	50.0%	131,072	65,536	8	4	⇔
Encoder3	43.8%	65,536	36,864	16	9	⇢
Encoder4	50.0%	32,768	16,384	32	16	↕
Encoder4	53.1%	16,384	7,680	64	30	⤭
Encoder Module	49.0%			124	59	

♦ = {}

⇔ = {1,4,6,7}

⇢ = {1,2,4,5,7,9,12}

↕ = {2,5,6,8,9,10,11,12,13,18,20,21,22, 24,26,29}

⤭ = {1,5,8,11,12,15,17,19,25,26,28-32, 38-44,47-49, 52, 53, 55,56,60, 61-63}

Considering the low number of feature maps generated by the Encoder1, the FSM module does not eliminate feature maps from the feature space generated by this encoder. The feature space from Encoder2 and Encoder3 was reduced to half of their original size. Encoder4 eliminates more than half of the feature maps from feature space. In total, the FSM eliminates 49% of the feature maps generated from the encoder module. Additionally, FSS_a value significantly reduces after silencing for Encoder2, Encoder3, and Encoder4. This represents the effectiveness of the FSM module in reducing the cost of the network. Therefore, we can infer from this phenomenon that almost half of the computations in encoder-decoder based architecture, does not contribute significantly to the prediction. Elimination of such features improve the performance as well as reduces the computations of the network remarkably.

3. Experiment Results

In this section, we discuss the results of different deep architectures on concrete crack identification. The data-set preparation and augmentation methods are described first. Then we represent our experimental setup and result in analysis in the following sections.

3.1. Data-set Preparation

The scarcity of a well-balanced data-set is a challenge for anomalous pixel identification applications such as concrete crack classification. Since the occurrence of crack pixels is much lower than healthy concrete pixels, it is essential to train a deep network architecture with an enormous amount of data instances containing crack pixels. Therefore, we have collected concrete images of different highway bridge decks from different parts of the USA using our previously developed nondestructive evaluation robots [2,3,49] at the Advanced Robotics and Automation Laboratory. We

captured images in various light illumination and different times of day and night to incorporate the non-uniformity of the environment as much as possible in our data-set (referred to as Illinois Bridge data-set).

(a) (b)

Figure 8. (a) Sample crack image and (b) annotated image from the Illinois Bridge data-set.

Appropriate annotation of the images in the data-set is very important for training and validating CNNs. This annotation process is arduous due to the difficulty involved in finding the crack pixels in an image (considering their low occurrence). As a result, the pixels were color-coded for each image individually from the Illinois Bridge data-set. The crack pixels were assigned a white color and the healthy concrete pixels were assigned a black color. An example of our annotated image data-set is shown in figure 8.

The annotated Illinois data-set consists of forty-six concrete crack images and their corresponding binary labels. Each image in the data-set has a resolution of size 5000×3000. We divided the original data-set into two separate data-sets such as training and testing. The training data-set includes 80% of the original data-set and the validation data-set includes the rest of the data-set. Additionally, a testing data-set was prepared to consist of 200 imaged of size 1024×1024 collected from different bridge decks.

There exists a number of published data-sets such as Crack260 [48], CrackForest [50] . These data-sets consist of very low resolution images (256×256, 512×512) in comparison to Illinois Bridge data-set. Additionally, the Illinois Bridge data-set consists of images with several types of noise such as vegetation, road paint, oil spilling and many more. An example of the annotated data set is shown in figure 8(a).

The crack inspection data-sets represents high-class imbalance property. To extract appropriate crack features, it is essential to provide the network with enough instances of crack samples. Therefore, we employed a data augmentation technique proposed by [42] for this purpose on our original data-set.

The data augmentation technique randomly selects an image from the original data-set (training or validation). A random sub-sample of the selected image is chosen for the data-augmentation operation. This operation is chosen randomly for each individual sub-samples. The data-augmentation operations used in this work are horizontal flipping, vertical flipping and gamma correction of intensity. The gamma value is also chosen randomly for intensity correction. This process is repeated for N times, where N represents the size of the data-set. As a result, this technique can generate N sub-samples of augmented data-set from the original high resolution data-set. A graphical representation of the data augmentation technique is shown in figure 9.

3.2. Experimental Setup and Result Analysis

The proposed network architecture was trained on a 1080 Gtx GPU with 10 Gb memory. For hyper-parameter optimization, Adam optimizer was used with a learning rate of 0.0001.

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

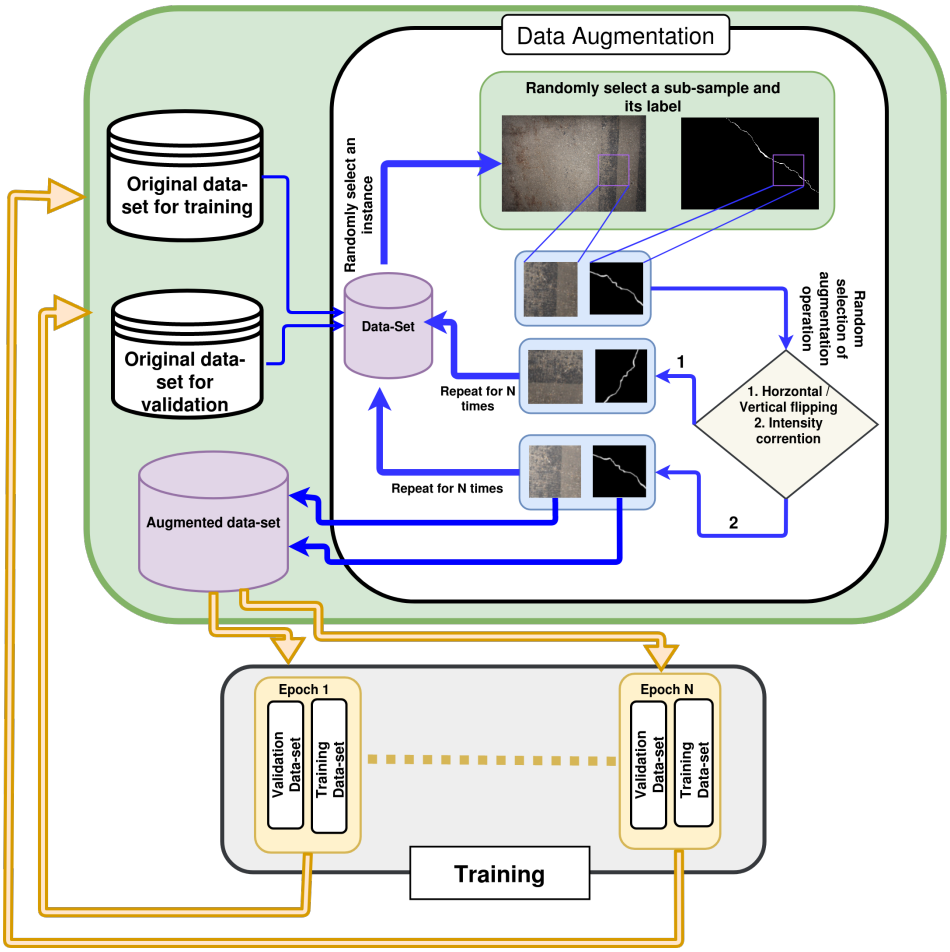

Figure 9. The training procedure with data augmentation technique.

The proposed deep network architecture was pre-trained on the Illinois Bridge data-set for 300 epochs. Many state-of-the-art CNN network training involves transfer learning from existing networks such as VGG-16 [34]. This transfer learning from the existing network has several disadvantages in our case. Firstly, transfer learning from state-of-the-art image processing data-sets (ImageNet) will give the network unnecessary information on various objects. Since crack pixels have a very small amount of occurrence in an image, the transfer learning process may overlap the features of crack. Secondly, the ANet-FSM architecture prunes feature space weights below a threshold (equation 1). Transferring weights from another module (or network) initialize the feature space with high weights. As a result, the FSM module cannot find the weak feature maps for pruning. For the aforementioned reasons, the network was pre-trained on the Illinois Bridge data-set instead of performing transfer learning.

The ANet-FSM network was trained for 300 epochs. At each epoch data-sets for training and validation are generated using the previously discussed data augmentation technique from the respective original training and validation data-set. In figure 9, we represented the process of data-augmentation and training visually. Each image of the training and validation data-set generated by the data-augmentation technique has a resolution of 512×512. The training data-set consists of 5000 images and the validation data-set consists of 1000 images. As a result, a data-set of 6000 images is generated in each epoch in training. In total, the network is trained and validated on 300 different data-sets respectively.

State-of-the-art data augmentation techniques generate augmented data-set from a handful of low-resolution images for crack detection. As result, training is performed on the same data-set for each epoch. This type of training procedure is highly susceptible to overfitting problem. Nonetheless,

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

the data-augmentation technique and training process incorporated in this paper generates different training and validation data-set for each epoch at the time of training. As a result, the network sees a range of different types of images. The overfitting problem of traditional training procedure is overcome through this training procedure.

We evaluate the performance of the pre-trained network using the test data-set of 200 images. The testing image size (1024×1024) was selected to be larger than the training image size (512×512) to represent the robustness of the network towards noise in large resolution images. Apart from this, the computational complexity involved with large images generates the gradient vanishing problem in a CNN architecture. The effect of the gradient vanishing problem is more evident in large scale images. Therefore, the resolution of test data-set images is twice larger than the train data-set.

The performance of ANet-FSM architecture was compared with two different type of encoder-decoder architectures such as semantic segmentation and crack detection architecture. The semantic segmentation architectures such as SegNet suffer from gradient vanishing problem as discussed in section 1. The effect of this problem is strongly evident in class-imbalanced data-sets (crack detection data-set). This effect was evaluated by comparing with semantic segmentation architecture (SegNet [40]). Moreover, several crack segmentation architectures were employed for performance comparison such as InspectionNet [41], SegNet-SO [42], Deep crack [43] and SDDNet [44].

The proposed network architecture was evaluated based on three different criteria such as network complexity, qualitative measurement and qualitative analysis. The complexity of different networks was analyzed in terms of the number of major computations performed. Later, a quantitative and qualitative analysis was performed on the results of different deep network architectures.

3.2.1. Network Complexity Analysis

One of the solution for tackling the gradient vanishing problem of very deep networks is the reduction of computational complexity. Since the convolutional layers are responsible for salient feature attribute extraction in CNN architecture [28], most of the computations are performed by this layer. As a result, the complexity of the network was measured based on the computations associated with the convolutional layers. The network complexity is defined as K_s in equation 4.

$$C = \sum_{i=0}^{\mathrm{E+D}} (N_{C_i} \times K_s) \times N_{k_i} \tag{4}$$

where N_C is the number of convolution layers, N_k is the number of kernels in each layer, K_s is a $m \times n$ dimensional kernel (m is height and n is width), E is the total number of encoders and D is the total number of decoders. Since the network architectures discussed in this work use five encoders and five decoders we set the value of $E = 5$ and $D = 5$.

To analyze the network complexity we take into consideration three encoder-decoder architectures in literature, i.e., SegNet [40] and InspectionNet [41]. Since the encoder and decoder networks in SegNet architecture follow the topology of VGG-16 architecture, we computed the complexity of this network. The number of computations required for the ANet-FSM architecture is compared with the aforementioned network architectures in table 2.

The number of total computations performed by VGG-16 architecture is 38016. Both the encoder and decoder network in SegNet follows VGG-16 architecture topology. Consequently, the number of convolution layers, kernels, and final computations increase by two times of the original VGG-16 architecture. The InspectionNet architecture performs 3806 fewer computations than SegNet. On the other hand, the ANet architecture performs 12152 computations. Despite using a larger convolution kernel this architecture performs 22408 fewer computations than InspectionNet. Usage of a single convolution layer in each encoder along with less filter numbers, reduced the computational complexity of ANet-FSM architecture. This aspect of the network helps eliminating the effect of gradient vanishing problem of deep networks. Furthermore, the precision of the network increases significantly due to

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 July 2020 doi:10.20944/preprints202007.0474.v1

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

Table 2. Comparison of network complexity of different encoder decoder based architecture. *Comp.*: network complexity, N_C: number of convolutional layers N_k: number of kernels in each convolutional layer, K_s: a $m \times n$ dimension kernel

Method	Nc	Nk	Ks	C
VGG-16[34]	{†}	⊕	3x3	38016
SegNet[40]	{†, ‡}	⁕	3x3	72576
InspectionNet[41]	⤫	ᛉ	3x3	34560
ANet-FSM	✜	⟁	7x7	12152
No. of Convs. per Encoder (Nc) : † = {2,2,3,3,3}, ‡ = {3,3,3,2,2}, ⤫ = {2,2,2,2,2}, ✜ = {1,1,1,1,1}				
No. of Filter Sets per Encoder (Nk) : ⊕ = {64, 128, 256, 512, 512} ⁕ = {64, 128, 256, 512, 512, 512,512,256, 128, 64} ᛉ = {64, 128, 256, 512, 512, 256, 128, 64} ◊ = {8, 16, 32, 64, 64, 64,64,32, 16, 8} ⟁= {4,8,16,32,64,64,32,16, 8,4}				

this fact. Therefore, ANet-FSM architecture is the most in-expensive network in terms of computation in comparison to the networks represented in table 2.

Table 3. Quantitative measures used for evaluating the results of deep network architectures.

Measure	Definition	Description
True Positive	TP	Number of accurately identified crack pixels
False Positive	FP	Number of falsely classified crack pixels
True Negative	TN	Number of accurately classified non-crack pixels
False Negative	FN	Number of falsely classified non-crack pixels
Accuracy	Acc	(TP+TN)/(TP+TN+FP+FN)
Error rate	Err	(FP+FN)/(TP+TN+FP+FN)
Specificity	Spc	TN/(TN+FP)
Sensitivity	Sens	TP/(TP+FN)
Precision	Precision	TP/(TP+FP)
Recall	Recall	TP/(TP+FN)
F1 score	F1	2 x (Precison x Recall)/(Precision+Recall)
Positive Class = Crack, Negative Class = Non-crack		

3.2.2. Quantitative Comparisons

This section presents the performance of different deep architectures in the literature for concrete crack identification along with ANet-FSM architecture quantitatively. Deep concrete crack detection architectures are categorized as image classification based architectures (Gibbs [3]) and encoder-decoder based architectures (SegNet[40], InspectionNet[41], SegNet-SO[42]). We evaluated the performance of these networks using the test data-set of 200 images of size 1024×1024.

In this paper, the state of the art statistical measures were applied to evaluate the performance of Deep Network architectures. . The statistical measures such as True positive (TP) rate, False positive (FP) rate, True negative (TN) rate, False-negative (FN) rate, Error rate, and Accuracy generate biased evaluation results toward non-crack pixels because of crack pixels low appearance. The use of effective statistical method, such as Specificity, Sensitivity, Precision, Recall, and F-1 measure was employed to distinguish between crack and non-crack pixels within imbalanced data-sets. These statistical measures were identified by defining crack pixels as positive class and non-crack pixels as negative

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 July 2020 doi:10.20944/preprints202007.0474.v1

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

Table 4. Overall Ranking of each method based on quantitative measures.

Method	Rank	TPRate	FNRate	TNRate	FPRate	Acc	Err	Spc	Sens	Prec	Rec	RI	F1
Gibbs[3]	8.0	25.2	74.8	77.0	23.0	51.1%	48.9%	77.0%	25.2%	52.3%	50.7%	51.1%	51.5%
SegNet[40]	6.5	71.9	28.1	98.9	1.1	85.4%	14.6%	98.9%	71.9%	98.5%	77.9%	85.4%	87.0%
SegNet-SO[42]	4.4	73.0	27.0	99.1	0.9	86.1%	14.0%	99.1%	73.0%	98.8%	78.6%	86.1%	87.5%
InspectionNet[41]	4.1	80.5	19.5	98.8	1.2	89.7%	10.4%	98.8%	80.5%	98.5%	83.5%	89.7%	90.4%
ANet	4.0	75.1	24.9	99.0	1.0	87.1%	13.0%	99.0%	75.1%	98.7%	79.9%	87.1%	88.3%
ANet-FSM (hi)	4.1	72.2	27.8	**99.7**	**0.3**	86.0%	14.1%	**99.7%**	72.2%	**99.6%**	78.2%	86.0%	87.6%
ANet-FSM (low)	*3.0*	**89.8**	**10.2**	98.7	1.3	**94.3%**	**5.8%**	98.7%	**89.8%**	98.6%	**90.6%**	94.3%	**94.4%**
ANet-FSM (opt)	**2.0**	*86.6*	13.4	*99.2*	0.8	*92.9%*	*7.1%*	*99.2%*	*86.6%*	*99.1%*	*88.1%*	92.9%	*93.3%*

Color Rank
1st
2nd
3rd
4th
5th
6th
7th
8th

Table 5. Ranking of each method based on individual quantitative measures.

Method	TPRate	FPRate	TNRate	FNRate	Acc	Err	Specificity	Sensitivity	Precision	Recall	F1-Measure
Gibbs[3]	8	8	8	8	8	8	8	8	8	8	8
SegNet[40]	7	7	5	5	7	7	5	7	7	7	7
SegNet-SO[42]	5	5	3	3	5	5	3	5	3	5	6
InspectionNet[41]	3	3	6	6	3	3	6	3	6	3	3
ANet	4	4	4	4	4	4	4	4	4	4	4
ANet-FSM (hi)	6	6	1	1	6	6	1	6	1	6	5
ANet-FSM (low)	1	1	7	7	1	1	7	1	5	1	1
ANet-FSM (opt)	2	2	2	2	2	2	2	2	2	2	2

class. We define the TP rate as the percentage of accurately identified crack pixels, whereas the TN rate is correctly identified non-crack pixels. The FP rates are delineated as the percentage of wrongly identified crack pixels and FNs are the incorrect identification percentage of non-crack pixels. We measure how exactly a method can distinguish between the crack and non-crack pixels with the Precision measure in an imbalanced data-set. We quantify the proportion of crack pixels identified accurately by a network with a recall score. The sensitivity measure evaluates the architecture's responsiveness toward the aberrant behavior of defected pixels. Specificity measure is used to quantify the behavior of non-crack pixels. The overall performance of a network is evaluated using the F1 score. A summary of the quantitative measures used in this work is shown in table 3. Apart from this, we assigned a ranking to the architectures based on both dependent and independent measures.

We evaluated the results of the proposed ANet-FSM architecture with three different thresholds. For applications in which high specificity and precision are required, higher threshold values should be utilized–see ANet-FSM (hi) in table 5. Lower thresholds would be effective in applications that require higher sensitivity and recall score is needed–see ANet-FSM (low) in table 5. In addition to these thresholds, we proposed the use of an optimal threshold, calculated using optimal thresholding algorithms such as the Otsu's method [51], when no preference is given for performance measure with respect to the positive or the negative classes–see ANet-FSM (opt) in table 5.

The overall results of different architecture and their average ranking on all the measures are shown in table 4. In addition to this, each network architecture is assigned ranking on individual measures on table 5, with 8 being the least accurate. Due to the block-based analysis technique for crack detection, [3] was ranked at 8, where lower rank corresponds to better performance of the algorithm and vice versa. This architecture classifies a sub-image of size 256×256 as crack or non-crack. Since crack pixels occur only a very small portion of an image, an enormous amount of pixels are falsely classified in these blocks. As a result, the false identification rate (both FP rate and FN rate), accuracy as well as the error rate of the Gibbs network is the worst among all the networks. This also represents the inefficiency of image classification methods in concrete crack identification and anomaly detection.

As expected, encoder-decoder architectures in table 4 and table 5, outperform classification networks such as the Gibbs architecture. Although SegNet [40] outperformed all the previous

architectures for semantic segmentation in the field of scene parsing, the extremely imbalanced nature of concrete defect data-set greatly drops the performance of this architecture. Additionally, the effect of the gradient vanishing problem (the result of an excessive number of layers) is reflected in evaluation measures such as FP and FN rates. The high false classification rate is also responsible for non-contributing feature maps generated due to the textured nature of the concrete surface (eliminated in ANet-FSM). Furthermore, the SegNet architecture under-performs in all but three measures in table 4. For the aforementioned reasons, SegNet architecture is not appropriate for solving the crack identification problem. As a result, SegNet achieves a low overall and individual ranking in all of the evaluation measures in table 5 and table 4.

The gradient vanishing problem affecting the SegNet was alleviated by up-sampling the feature space of each encoder layer in Segnet-SO [42] architecture. This approach achieved more TP rate and less error rate than SegNet architecture. Although the InspectionNet architecture improves the TP rate significantly, the highest FP rate in table 4 demonstrates the effect of gradient vanishing problem. SegNet-SO architecture is more robust to the gradient vanishing problem despite achieving a lower TP rate than InspectionNet. Therefore, it is worth mentioning that, none of the architectures discussed above represent robustness in all the measures.

The ANet-FSM architectures alleviate the drawbacks of SegNet, SegNet-SO, and InspectionNet architectures by eliminating redundant computation. These computationally expensive methods represent a fluctuation in results. For example, InspectionNet obtains outstanding correct classification with the cost of an unacceptable misclassification rate. SegNet-SO degrades the correct classification rate in the course of reducing incorrect classification. To evaluate the stability of the ANet-FSM architecture, we have analyzed its performance without incorporating the FSM (referred to as ANet). The robustness of ANet architecture is reflected by the ranking of 4 in each measure in 5. Although the InspectionNet architecture performs better in positive classification (TP), the low negative classification rate (TN) represents the unfeasible nature of this network towards imbalanced data-set. Therefore ANet architecture is substantially stable (the effect of using 7×7 spatial neighborhood in the convolution) despite achieving a lower ranking in some measures. However, the higher false-positive rate of this network than InspectionNet represents that it is affected by the vanishing gradient problem because of a 7×7 kernel size. The association of the FSM model significantly alleviates this problem as well as improves the performance in all measures.

To further investigate the result of feature silencing we performed thresholding operation on the result obtained from the ANet-FSM architecture. We discarded the crack pixels having a lower probability than a specific threshold in this operation. Three different threshold values were set experimentally to perform this operation such as high (15%), low (10%) and optimal (14%). It is evident from table 5 and 4 that ANet-FSM(low) architecture achieves highest performance in all but four measures. Specifically, ANet-FSM (low) have recognized the highest number of crack pixels among all the networks in table 5. However, the lowest TN and FP rates represent this network's bias toward only positive classification. Therefore, this thresholding is suitable for application requiring to classify only crack locations. On the other hand, when a higher threshold is applied to ANet-FSM architecture, the FP rate significantly drops with the cost of a low TP rate. This thresholding technique is appropriate for applications that need to know healthy concrete locations. An optimal threshold was set experimentally to achieve a better TP rate and moderately lower FP rate than the previous networks. The ANet-FSM (opt) architecture outperforms all the aforesaid networks in every measure with a rank of 2. Though ANet-FSM(opt) doesn't perform best in all of the measures, the second-best ranking represents its stability in identifying both crack and non-crack pixels.

The above discussion evaluates the result of different architectures based on dependent evaluation measures. As mentioned earlier, these measures are highly biased toward the classification of the overwhelming majority class (non-crack). Nonetheless, to perform fair evaluation we have taken into account some measures such as Precision, Recall, Sensitivity, Specificity, and F1-score.

Gibbs method [3] and SegNet [40] architecture have the lowest precision and recall score in table 5 and 4. SegNet-SO architecture has a higher precision rate than InspectionNet architecture. However, the recall score represents a reverse relationship between SegNet-SO and InspectionNet. This tension between precision and recall is a well-known phenomenon within classification problems suffering from class-imbalance issues. Moreover, specificity and sensitivity represent similar relationships as precision and recall due to excessive class imbalance present in concrete crack data-sets. If a network is highly specific, its sensitivity reduces (SegNet-SO) whereas a high sensitivity rate reduces the specificity of a network(InspectionNet). As a result, the F1 score is widely used to combine the effects of these measures for any machine learning architecture. The F1 score of SegNet-SO and InspectionNet represents that the former is better in terms of overall performance.

Table 6. Ranked methods based on dependent measures.

Rank	Specificity	Sensitivity	Precision	Recall	F1
1	ANet-FSM (hi)	ANet-FSM (low)	ANet-FSM (hi)	ANet-FSM (low)	ANet-FSM (low)
2	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)
3	SegNet-SO[42]	InspectionNet[41]	SegNet-SO[42]	InspectionNet[41]	InspectionNet[41]
4	ANet	ANet	ANet	ANet	ANet
5	SegNet[40]	SegNet-SO[42]	ANet-FSM (low)	SegNet-SO[42]	ANet-FSM (hi)
6	InspectionNet[41]	ANet-FSM (hi)	InspectionNet[41]	ANet-FSM (hi)	SegNet-SO[42]
7	ANet-FSM (low)	SegNet[40]	SegNet[40]	SegNet[40]	SegNet[40]
8	Gibbs[3]	Gibbs[3]	Gibbs[3]	Gibbs[3]	Gibbs[3]

Table 7. Ranked methods based on independent measures.

Rank	True Pos Rate	False Pos Rate	True Neg Rate	False Neg Rate	Acc Rate	Err
1	ANet-FSM (low)	ANet-FSM (low)	ANet-FSM (hi)	ANet-FSM (hi)	ANet-FSM (low)	ANet-FSM (low)
2	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)	ANet-FSM (opt)
3	InspectionNet[41]	InspectionNet[41]	SegNet-SO[42]	SegNet-SO[42]	InspectionNet[41]	InspectionNet[41]
4	ANet	ANet	ANet	ANet	ANet	ANet
5	SegNet-SO[42]	SegNet-SO[42]	SegNet[40]	SegNet[40]	SegNet-SO[42]	SegNet-SO[42]
6	ANet-FSM (hi)	ANet-FSM (hi)	InspectionNet[41]	InspectionNet[41]	ANet-FSM (hi)	ANet-FSM (hi)
7	SegNet[40]	SegNet[40]	ANet-FSM (hi)	ANet-FSM (hi)	SegNet[40]	SegNet[40]
8	Gibbs[3]	Gibbs[3]	Gibbs[3]	Gibbs[3]	Gibbs[3]	Gibbs[3]

On the other hand, the ANet architecture maintains a stable precision, recall, specificity, and sensitivity scores (all are assigned the same rank in table 5). Consequently, its F1 score is better than SegNet-SO and less than InspectionNet because of lower sensitivity. However, the ANet-FSM architecture with a low threshold achieves exceptional recall scores with relatively low specificity, resulting in the highest F1 score of all the methods. If extreme thresholding is applied, ANet-FSM architecture obtains the highest precision and specificity score with moderately low recall and sensitivity score among all the architectures. The optimal thresholding operation achieves higher precision, recall, specificity, sensitivity and F1 scores than all the networks in table 4 and table 5. The same ranking in all of the measures in table 5 also demonstrates the stability of the network. This architecture obtains the highest F1-score among all the computationally expensive networks (SegNet, InspectionNet, SegNet-SO). Therefore, it can be concluded that this network is appropriate for use in applications with highly class-imbalanced data.this

We have also represented a ranked position of different deep network architectures for dependent and independent measures in table 6 and table 7, respectively. According to our earlier propositions of dependent measures, it is evident from table 6, the ANet-FSM architecture outperforms the existing encoder-decoder architecture for defect identification despite the bias towards the non-crack class. The independent measures also represent that the ANet-FSM architecture outperforms the other existing networks in table 7. Moreover, ANet-FSM (hi) is suitable for applications requiring extremely specific and precise results. ANet-FSM (low) is appropriate for highly sensitive applications for concrete crack

detection. The robustness of ANet-FSM (opt) toward specificity, sensitivity, precision, and recall makes this suitable for applications that require stable results.

Table 8. Performance of crack identification in different data set. The measures represented in this table are from Berkely segmentation benchmark [52]. The lowest value of each measure represents the best.

Data Set	Method	BDE	GCE	VI
Illinois Bridge Dataset	ANet-FSM	**0.19**	**0.988**	**0.36**
	SegNet[40]	2.39	0.993	1.40
	InspectionNet[41]	1.42	0.995	0.39
Crack260[48]	ANet-FSM	1.84	0.992	1.53
	SegNet[40]	**1.72**	**0.991**	**0.92**
	InspectionNet[41]	2.40	0.992	1.01
CrackForest[50]	ANet-FSM	**0.89**	**0.992**	1.22
	SegNet[40]	2.20	**0.992**	0.79
	InspectionNet[41]	1.60	0.980	**0.78**

To perform an unbiased comparison, we have evaluated the performance of the proposed architecture using the evaluation metrics from Berkley segmentation benchmark [52]. Three evaluation metrics from the benchmark were employed to assess the performance of the networks such as boundary displacement error (BDE), global consistency error (GCE) and variation in information (VI). The BDE measures the distance between the boundary pixels between two segmented images. GCE represents how closely two segmented images can be shown as a representation of one another. The VI is used widely for data clustering applications. It measures the distance between two clusters (resembles mutual information). Since the probabilistic rand index replicates the same measurement as the accuracy of the algorithms, we have avoided this measurement. For comparison purposes, we have considered the SegNet, InspectionNet and the proposed ANet-FSM architecture. The SegNet architecture was chosen by us to evaluate the effect of gradient vanishing problem on state-of-the-art semantic segmentation network. We have chosen InspectionNet to evaluate the effect of gradient vanishing problem in crack detection architecture. The comparison of different methods on the aforementioned metrics on different data sets are shown in table 8.

We used three different types of data-set for performing a fair evaluation of the proposed method. The aforementioned metrics were applied on Crack260 [48], CrackForest [50] and Illinois Bridge data-set. The Crack260 and CrackForest data-sets are published annotated data set for crack classification. The Illinois Bridge data-set was collected and prepared by the researchers of Advanced Robotics and Automation Lab. In table 8, for Crack260 data-set SegNet achieves the lowest BDE, whereas InspectionNet achieves the highest BDE. Considering the sheer number of parameters involved in SegNet, this low error rate is reasonable. In InspectionNet the number of parameters is more than SegNet. Due to the parameter degradation problem, the BDE is highest in this network. The ANet-FSM architecture achieves a BDE close to SegNet, despite having almost half a number of parameters than SegNet. On the other hand for CrackForest and Illinois Bridge data-set, the BDE is lowest in ANet-FSM architecture. This represents that pruning feature space significantly enhances network performance. For Illinois data-set, the BDE of ANet-FSM is seven times lower than InspectionNet and twelve times lower than SegNet. For CrackForest data-set, the BDE of ANet-FSM is 1.8 times lower than InspectionNet and 2.5 times lower than SegNet. These two data-sets significantly represents the effect gradient vanishing problem in complex architectures like SegNet and InspectionNet.

ANet-FSM architecture achieves the lowest GCE in CrackForest and Illinois Bridge data-set. However, for the Crack260 data-set, it achieves the second-best result among all the methods. On the other hand, ANet-FSM architecture achieves best VI for the Illinois Bridge data-set. For CrackForest data-set InspectionNet achieves the best VI. SegNet achieves the best VI for the Crack260 data-set.

The rand index metric in the Berkley segmentation benchmark represents the previously analyzed measure accuracy. Since, accuracy is dependent on TP, FP, FN and TN for measurement, it was not incorporated for evaluation in table 9 and 8. Apart from this, the region uniformity measure is widely used in semantic segmentation architecture evaluation. Region uniformity is more appropriate for image segmentation problems such as scene parsing and medical image analysis. These segmentation problems identify a region containing a substantial amount of pixels. Unlike these regions, crack width length can be of one pixel to several pixels. The crack area encompasses a very small number of pixels (usually five to ten pixels approximately). For this reason, VI and GCE measures are not directly applicable to the crack segmentation problem also. As a result, the GCE present in table 8 is higher than usual and the VI measure shows different behavior for each individual data-set.

Table 9. Comparison ANet-FSM architecture with crack detection architectures. The ANet-FSM architecture was trained and tested on the data-set prepared by DeepCrack[43].

Method	Prec	Rec	F1	mIoU	Proceesing time
DeepCrack[43]	86.10%	86.90%	86.50%	80.20%	109 ms
SDDNet[44]	87.10%	87.00%	87.00%	87.90%	13.54ms
ANet-FSM	98.2%	78.2%	87.1%	79.0%	1.14ms

We have also compared the performance of ANet-FSM architecture with state-of-the-art crack detection architectures such as DeepCrack [43] and SDDNet [44]. The results of deep crack and SDDNet were extracted from the experiments reported in [43]. The ANet-FSM architecture was trained and tested using the data-set [43] used for the experiment in [44]. For comparison metrics, we have taken into consideration the mean intersection over union (mIou) [44], precision, recall, F1 score and the processing time. The results are shown in table 9. SDDNet architecture achieves the highest F1 score and mIou among all the methods. However, the ANet-FSM architecture achieves the highest precision rate and an F1 score close to SDDNet. This phenomenon represents the effect of gradient vanishing in lowest in ANet-FSM among all the methods. Moreover, the processing time of ANet-FSM is 1.14 ms per image, whereas SDDNet has 13.04 ms per image. The processing speed of ANet-FSM architecture is thirteen-time smaller than the SDDNet architecture. Considering this significant fast processing time of ANet-FSM architecture, the smaller mIou score is reasonable. Apart from this, this processing time also depicts ANet-FSM architecture is nominally affected by gradient stability problem.

3.2.3. Qualitative Comparisons

The qualitative comparison of several crack identification networks is performed in this section. We first show the results of ANet-FSM architecture in figure 12. Then we compare our results with the image classification method (Gibbs [3]). Finally, the results are compared with the state-of-the-art encoder-decoder architectures such as SegNet [40], SegNet-SO [42] and InspectionNet [41]. We color-coded the true positive pixels (correctly detected crack pixels) with red, FP's (missed crack pixels) with blue, and FN's (pixels incorrectly labeled as crack) with green, respectively. The TN pixels (correctly labeled non-crack) are represented with their original texture.

We have represented the results of ANet-FSM architecture on three thresholding values such as high (15%), low(10%) and optimal (14%) in figure 10. If a low threshold is applied, the network becomes more sensitive towards the environmental non-uniformity and gradient vanishing problem. On the other hand the network becomes more specific to correct classification. As a result, the number of falsely identified (blue colors) and correctly classified pixels (red colors) of the network significantly increases (shown in row 2(a),(b),(c) of figure 10). When a high threshold is applied (shown in row 3(a),(b),(c) of figure 10) the sensitivity towards noise is alleviated but the specificity of correct classification also decreases. Application of an optimal threshold not only reduces the false classification rate but also

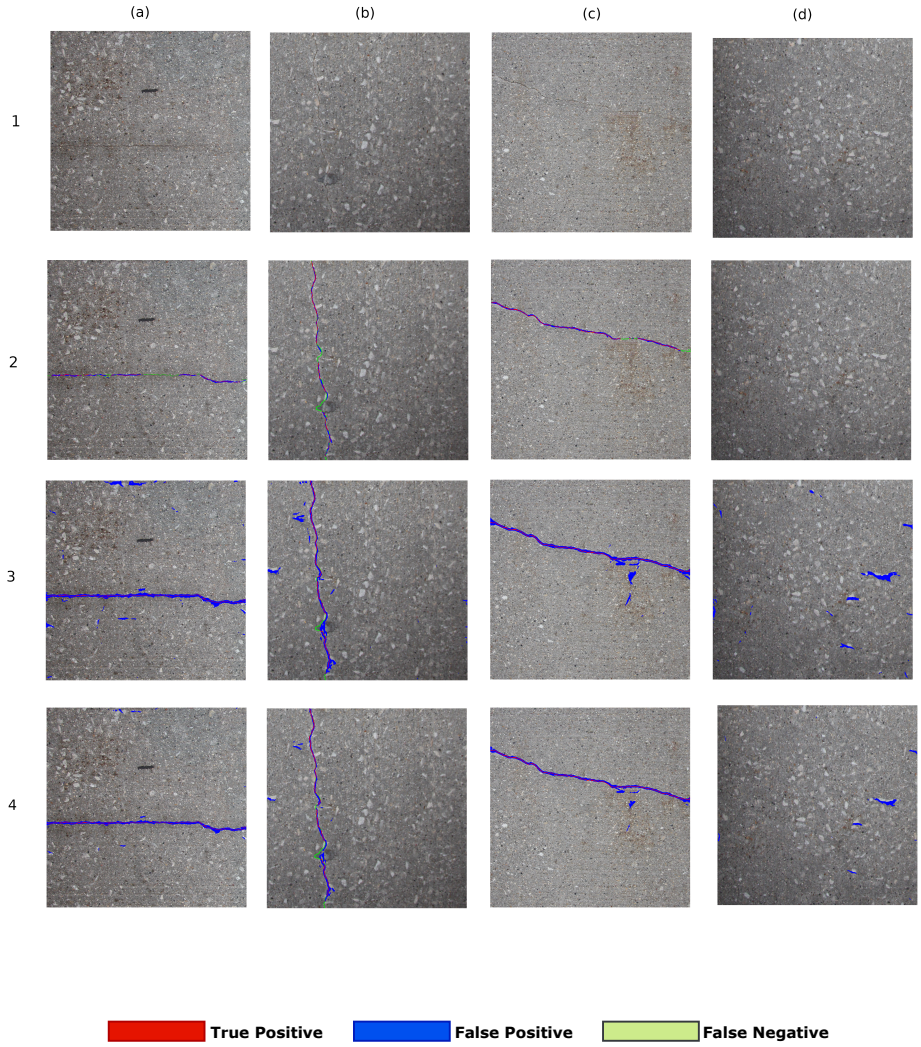

Figure 10. Comparison of different thresholds on the performance of ANet-FSM on four sample images. Columns are the tested images: (a) contains clear horizontal crack, (b) contains clear vertical crack, (c) contains a crack at arbitrary orientation, (d) non-cracked concrete image. Row 1- Original image; Row 2- ANet-FSM (hi); Row 3- ANet-FSM (low); and Row 4- ANet-FSM (opt). Results will be seen clearer when zoom-in.

increases correct classifications significantly. This thresholding obtains a balance between specificity and sensitivity as shown in figure 10.

We compared the results of the Gibbs [3] architecture with ANet-FSM architecture in figure 11. Gibbs architecture divides the original image into smaller sub-blocks of size 256×256 and classifies them as crack and non-crack. The non-crack blocks are marked as black pixels in figure 11 and crack blocks are represented with their original texture. The results represent that, Gibbs architecture falsely identifies many crack blocks as well as fails to identify the exact location of cracks. On the other hand, the ANet-FSM architecture localize and identify crack location more precisely than the Gibbs architecture. For example, in image 2(a) Gibbs method falsely identifies four 256×256 blocks as crack blocks. The ANet-FSM architecture in figure 3(a) represents the exact crack location as well as misidentifies a very less number of crack pixels in comparison to Gibbs architecture.

The result of different encoder-decoder networks such as SegNet, SegNet-SO and InspectionNet and the proposed ANet-FSM architecture is shown in figure 12.

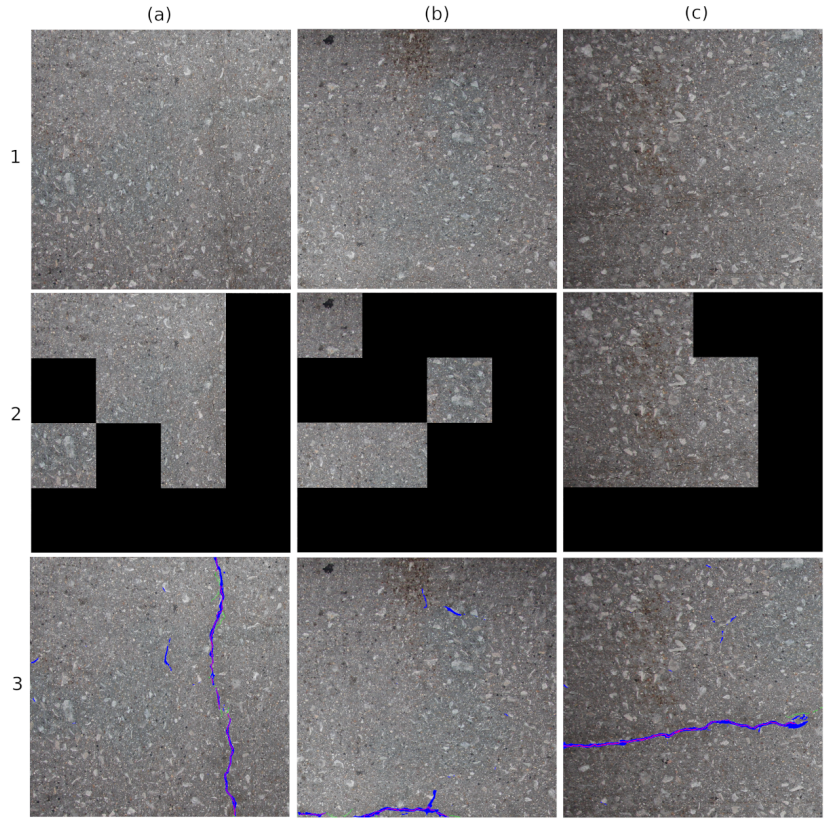

Figure 11. Comparison of image classification architectures with encoder-decoder architecture on three sample images. Columns are the tested images: (a) contains clear vertical crack, (b) contains small crack on bottom, (c) contains a crack at arbitrary orientation. Row 1- Original image; Row 2- Gibbs architecture [3]; Row 3- ANet-FSM architecture.

The results of 2(a), (c) and (d) in figure 12 show that the SegNet architecture has more falsely classified pixels (blue colored) than the remaining networks. The excessive number of feature space (due to maximal network complexity), as well as the vanishing gradients, contribute to this false classification. The SegNet-SO architecture in 3(b) moderately removes the false classification present in image 2(b). The results in 3(a), (c) and (d) represent an considerable amount of falsely classified pixels, specifically in 3(a) where no crack pixels are present originally. On the other hand, the InspectionNet architecture represented in image 4(a), (b), (c) and (d) shows a performance improvement in comparison to SegNet and SegNet-SO. The results in 4(a) and 4(d) are less affected by the FP rate than SegNet and SegNet-SO. However, the false identification rate increases more than SegNet-SO architecture when a significant number of crack pixels are present as shown in image 3(c). As a result, it can be interpreted that InspectionNet is highly unstable as well as affected by the environmental non-uniformity (lighting and shading). On the other hand, the effect of FPs is significantly low in ANet-FSM architecture in comparison to the results in row-2, row-3 and row-4. It has almost no false identification (blue pixels) in figure 5(a). There exists a small amount of falsely identified pixels in figure 5(d), which is considerably lower than the previous networks. Moreover, this network alleviates this false classification without affecting the correct classification rate (represented as red pixels in 5(b) and (c)), which is the effect of feature silencing. The ANet-FSM architecture not only improves the accuracy of crack identification but also eliminates the effect of false identification significantly with the FSM. Therefore, it can be concluded, ANet-FSM architecture is less effected by the gradient vanishing problem in comparison to the encoder-decoder architectures in [40–42]. Based on the percentage of correctly identified cracks pixels identified, it can be concluded that ANet-FSM provides performance that is an improvement on

Peer-reviewed version available at Sensors 2020, 20, 4403; doi:10.3390/s20164403

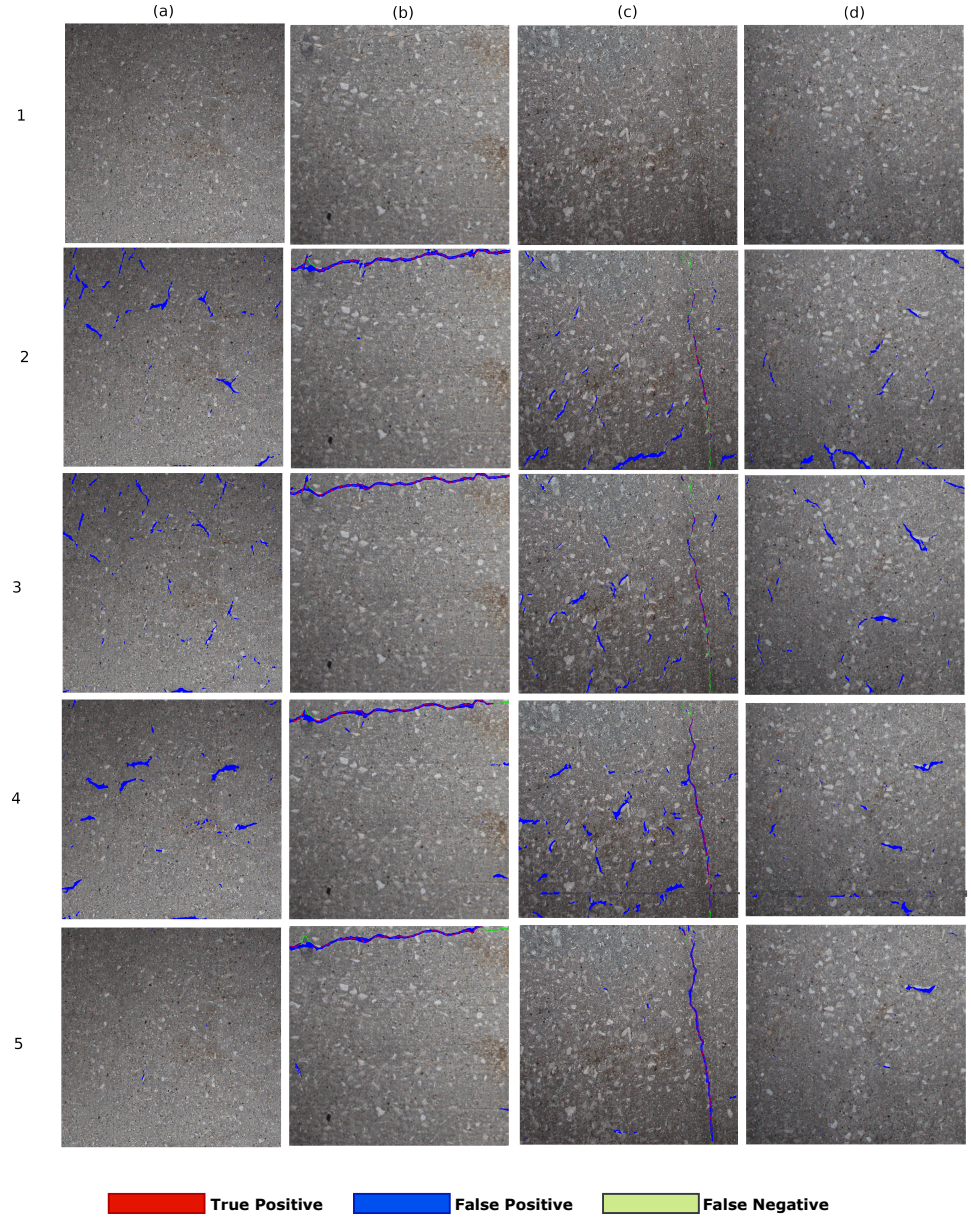

Figure 12. Comparison of different encoder-decoder based architectures on four sample images. Columns are the tested images: (a) contains non-cracked concrete image, (b) contains clear horizontal crack on top, (c) contains clear vertical crack, (d) contains non-crack concrete image. Row 1- Original image; Row 2- SegNet [40]; Row 3- SegNet-SO [42]; Row 4- InspectionNet; Row 5- ANet-FSM architecture.

the state-of-the-art encoder decoder network architectures designed for crack detection in the recent past.

4. Conclusions and Future Work

A deep convolutional neural network for concrete crack classification is presented in this work. A framework for eliminating the effect of gradient vanishing problem for class imbalanced data-set is presented in this work. Silencing unnecessary feature space, enhances the precision of our framework as well as reduce the effect of gradient vanishing problem. Moreover, the sensitivity of the architecture to environmental non-uniformity is reduced by the FSM module. As a result, our architecture is more precise than the crack detection methods present in the literature. The experimental results in this

study also represent that an enormous amount of unnecessary computations is performed in deep architectures. Elimination of these computations enhance the speed and processing of deep networks remarkably, which is important for real-time deployment of this architectures. The FSM module selects feature based on some threshold in the proposed framework. In the future, we want to develop an optimization framework for feature silencing in class imbalanced data-sets. Another future direction of this work would be exploring the area of other concrete distress identification such as spalling detection using an optimized feature silencing module.

Author Contributions: Conception and design (H.M.L. and A.T.); analysis and interpretation (U.H.B., H.M.L., and A.T.); writing the article (U.H.B.); critical revision (A.T. and H.M.L.); final approval of the article (A.T., H.M.L., and U.H.B.); data collection (H.M.L. and U.H.B.); literature search (H.M.L., A.T., and U.H.B.)

Funding: This work is supported by the U.S. National Science Foundation (NSF) under grants NSF-CAREER: 1846513 and NSF-PFI-TT: 1919127, and the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (USDOT/OST-R) under Grant No. 69A3551747126 through INSPIRE University Transportation Center, the Vingroup Innovation Foundation (VINIF) in project code VINIF.2020.NCUD.DA094, and the Japan NineSigma through the Penta-Ocean Construction Ltd. Co. under Agreement No. SP-1800087. The views, opinions, findings and conclusions reflected in this publication are solely those of the authors and do not represent the official policy or position of the NSF, the USDOT/OST-R and any other entities.

Conflicts of Interest: Authors do not have any conflicts of interest in this study

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Networks
FSM	Feature Silencing Module
ANet	Attention Network
DoG	Difference of Gaussians
ANN	Artificial Neural Networks
SVM	Support Vector Machines
MLP	Multi Layer Perceptron
BN	Batch Normalization
ReLU	Rectified Linear Unit
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
SO	Side Outputs
BDE	Boundary Displacement Error
VI	Variation in Information
GCE	Global Consistency Error

References

1. La, H.M.; Lim, R.S.; Basily, B.B.; Gucunski, N.; Yi, J.; Maher, A.; Romero, F.A.; Parvardeh, H. Mechatronic Systems Design for an Autonomous Robotic System for High-Efficiency Bridge Deck Inspection and Evaluation. *IEEE/ASME Transactions on Mechatronics* **2013**, *18*, 1655–1664. doi:10.1109/TMECH.2013.2279751.
2. La, H.M.; Gucunski, N.; Dana, K.; Kee, S.H. Development of an autonomous bridge deck inspection robotic system. *Journal of Field Robotics* **2017**, *34*, 1489–1504, [https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21725]. doi:10.1002/rob.21725.
3. Gibb, S.; La, H.M.; Le, T.; Nguyen, L.; Schmid, R.; Pham, H. Nondestructive evaluation sensor fusion with autonomous robotic system for civil infrastructure inspection. *Journal of Field Robotics* **2018**, *35*, 988–1004.
4. Sy, N.; Avila, M.; Begot, S.; Bardet, J.C. Detection of defects in road surface by a vision system. Electrotechnical Conference, 2008. MELECON 2008. The 14th IEEE Mediterranean. IEEE, 2008, pp. 847–851.

5. Li, Q.; Liu, X. Novel approach to pavement image segmentation based on neighboring difference histogram method. Image and Signal Processing, 2008. CISP'08. Congress on. IEEE, 2008, Vol. 2, pp. 792–796.
6. Oliveira, H.; Correia, P.L. Automatic road crack segmentation using entropy and image dynamic thresholding. Signal Processing Conference, 2009 17th European. IEEE, 2009, pp. 622–626.
7. Dinh, T.H.; Ha, Q.P.; La, H.M. Computer vision-based method for concrete crack detection. 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), 2016, pp. 1–6. doi:10.1109/ICARCV.2016.7838682.
8. Tavakkoli, A.; Nicolescu, M.; Bebis, G. Automatic robust background modeling using multivariate non-parametric kernel density estimation for visual surveillance. International Symposium on Visual Computing. Springer, 2005, pp. 363–370.
9. Sun, Y.; Salari, E.; Chou, E. Automated pavement distress detection using advanced image processing techniques. Electro/Information Technology, 2009. Eit'09. IEEE International Conference on. IEEE, 2009, pp. 373–377.
10. Landstrom, A.; Thurley, M.J. Morphology-based crack detection for steel slabs. *IEEE Journal of selected topics in signal processing* **2012**, *6*, 866–875.
11. Bai, X.; Zhou, F.; Xue, B. Multiple linear feature detection based on multiple-structuring-element center-surround top-hat transform. *Applied optics* **2012**, *51*, 5201–5211.
12. Elbehiery, H.; Hefnawy, A.; Elewa, M. Surface defects detection for ceramic tiles using image processing and morphological techniques, 2005.
13. Tanaka, N.; Uematsu, K. A Crack Detection Method in Road Surface Images Using Morphology. *MVA* **1998**, *98*, 17–19.
14. Maode, Y.; Shaobo, B.; Kun, X.; Yuyao, H. Pavement crack detection and analysis for high-grade highway. Electronic Measurement and Instruments, 2007. ICEMI'07. 8th International Conference on. IEEE, 2007, pp. 4–548.
15. Zhao, H.; Qin, G.; Wang, X. Improvement of canny algorithm based on pavement edge detection. 2010 3rd International Congress on Image and Signal Processing. IEEE, 2010, Vol. 2, pp. 964–967.
16. Lim, R.S.; La, H.M.; Shan, Z.; Sheng, W. Developing a crack inspection robot for bridge maintenance. 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 6288–6293. doi:10.1109/ICRA.2011.5980131.
17. Prasanna, P.; Dana, K.; Gucunski, N.; Basily, B. Computer-vision based crack detection and analysis. Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2012. International Society for Optics and Photonics, 2012, Vol. 8345, p. 834542.
18. Lim, R.S.; La, H.M.; Sheng, W. A robotic crack inspection and mapping system for bridge deck maintenance. *IEEE Transactions on Automation Science and Engineering* **2014**, *11*, 367–378.
19. Tavakkoli, A.; Nicolescu, M.; Bebis, G. An adaptive recursive learning technique for robust foreground object detection. the International Workshop on Statistical Methods in Multi-image and Video Processing (in conjunction with ECCV06, 2006, pp. 1–6.
20. Tavakkoli, A.; Ambardekar, A.; Nicolescu, M.; Louis, S. A genetic approach to training support vector data descriptors for background modeling in video data. International Symposium on Visual Computing. Springer, 2007, pp. 318–327.
21. Cha, Y.J.; Choi, W.; Büyüköztürk, O. Deep learning-based crack damage detection using convolutional neural networks. *Computer-Aided Civil and Infrastructure Engineering* **2017**, *32*, 361–378.
22. Gavilán, M.; Balcones, D.; Marcos, O.; Llorca, D.F.; Sotelo, M.A.; Parra, I.; Ocaña, M.; Aliseda, P.; Yarza, P.; Amírola, A. Adaptive road crack detection system by pavement classification. *Sensors* **2011**, *11*, 9628–9657.
23. Bray, J.; Verma, B.; Li, X.; He, W. A neural network based technique for automatic classification of road cracks. Neural Networks, 2006. IJCNN'06. International Joint Conference on. IEEE, 2006, pp. 907–912.
24. Wu, L.; Mokhtari, S.; Nazef, A.; Nam, B.; Yun, H.B. Improvement of crack-detection accuracy using a novel crack defragmentation technique in image-based road assessment. *Journal of Computing in Civil Engineering* **2014**, *30*, 04014118.
25. O'Byrne, M.; Ghosh, B.; Schoefs, F.; Pakrashi, V. Regionally enhanced multiphase segmentation technique for damaged surfaces. *Computer-Aided Civil and Infrastructure Engineering* **2014**, *29*, 644–658.
26. Banharnsakun, A. Hybrid ABC-ANN for pavement surface distress detection and classification. *International Journal of Machine Learning and Cybernetics* **2017**, *8*, 699–710.

27. Moon, H.; Kim, J.; others. Intelligent crack detecting algorithm on the concrete crack image using neural network. *Proceedings of the 28th ISARC* **2011**, pp. 1461–1467.
28. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *nature* **2015**, *521*, 436.
29. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
30. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 4278–4284.
31. Billah, U.H.; La, H.M.; Tavakkoli, A.; Gucunski, N. Classification of Concrete Crack using Deep Residual Network. 9th International Conference on Structural Health Monitoring of Intelligent Infrastructure (SHMII-9), 2019, pp. 1–6.
32. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 2012, pp. 1097–1105.
33. Kim, B.; Cho, S. Automated vision-based detection of cracks on concrete surfaces using a deep learning technique. *Sensors* **2018**, *18*, 3452.
34. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* **2014**.
35. Gibb, S.; La, H.M.; Louis, S. A Genetic Algorithm for Convolutional Network Structure Optimization for Concrete Crack Detection. 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1–8. doi:10.1109/CEC.2018.8477790.
36. Dung, C.V.; others. Autonomous concrete crack detection using deep fully convolutional neural network. *Automation in Construction* **2019**, *99*, 52–58.
37. Dai, J.; Li, Y.; He, K.; Sun, J. R-fcn: Object detection via region-based fully convolutional networks. Advances in neural information processing systems, 2016, pp. 379–387.
38. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.
39. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. International Conference on Medical image computing and computer-assisted intervention. Springer, 2015, pp. 234–241.
40. Badrinarayanan, V.; Handa, A.; Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293* **2015**.
41. Yang, L.; Li, B.; Li, W.; Jiang, B.; Xiao, J. Semantic Metric 3D Reconstruction for Concrete Inspection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 1543–1551.
42. Billah, U.H.; Tavakkoli, A.; La, H.M. Concrete Crack Pixel Classification using an Encoder Decoder Based Deep Learning Architecture. International Symposium on Visual Computing. Springer, 2019, pp. 593–604.
43. Zou, Q.; Zhang, Z.; Li, Q.; Qi, X.; Wang, Q.; Wang, S. Deepcrack: Learning hierarchical convolutional features for crack detection. *IEEE Transactions on Image Processing* **2018**, *28*, 1498–1512.
44. Choi, W.; Cha, Y.J. SDDNet: Real-time Crack Segmentation. *IEEE Transactions on Industrial Electronics* **2019**.
45. Bang, S.; Park, S.; Kim, H.; Kim, H. Encoder–decoder network for pixel-level road crack detection in black-box images. *Computer-Aided Civil and Infrastructure Engineering* **2019**, *34*, 713–727.
46. Zhang, J.; Lu, C.; Wang, J.; Wang, L.; Yue, X.G. Concrete cracks detection based on FCN with dilated convolution. *Applied Sciences* **2019**, *9*, 2686.
47. Li, H.; Song, D.; Liu, Y.; Li, B. Automatic pavement crack detection by multi-scale image fusion. *IEEE Transactions on Intelligent Transportation Systems* **2018**, *20*, 2025–2036.
48. Zou, Q.; Zhang, Z.; Li, Q.; Qi, X.; Wang, Q.; Wang, S. Deepcrack: Learning hierarchical convolutional features for crack detection. *IEEE Transactions on Image Processing* **2018**, *28*, 1498–1512.
49. Gibb, S.; Le, T.; La, H.M.; Schmid, R.; Berendsen, T. A multi-functional inspection robot for civil infrastructure evaluation and maintenance. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 2672–2677. doi:10.1109/IROS.2017.8206091.
50. Shi, Y.; Cui, L.; Qi, Z.; Meng, F.; Chen, Z. Automatic road crack detection using random structured forests. *IEEE Transactions on Intelligent Transportation Systems* **2016**, *17*, 3434–3445.
51. Otsu, N. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics* **1979**, *9*, 62–66.

52. Arbelaez, P.; Maire, M.; Fowlkes, C.; Malik, J. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence* **2010**, *33*, 898–916.

Sample Availability: Samples of the code and results are available from the authors.