I.J. Image, Graphics and Signal Processing, **2022, 5, 28-43**
Published Online on October 8, 2022 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2022.05.03

FeatureGAN: Combining GAN and Autoencoder for Pavement Crack Image Data Augmentations

Xinkai Zhang
School of computing and artificial intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China
E-mail: xkzhang@my.swjtu.edu.cn

Bo Peng
School of computing and artificial intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China
E-mail: bpeng@swjtu.edu.cn

Zaid Al-Huda
School of computing and artificial intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China
E-mail: zaidalhuda@swjtu.edu.cn

Donghai Zhai
School of computing and artificial intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China
E-mail: dhzhai@swjtu.edu.cn

Received: 15 May 2022; Revised: 14 June 2022; Accepted: 27 July 2022; Published: 08 October 2022

Abstract: In the pavement crack segmentation task, the accurate pixel-level labeling for the fully supervised training of deep neural networks Deep Neural Networks (DNN) is challenging. Although cracks often exhibit low-level image characters in terms of edges, there might be various high-level background information based on the complex pavement conditions. In practice, crack samples containing various semantic backgrounds are scarce. To overcome these problems, we propose a novel method for augmenting the training data for DNN based crack segmentation task. It employs the generative adversarial network Generative Adversarial Networks (GAN), which utilizes a crack-free image, a crack image, and a corresponding image mask to generate a new crack image. In combination with an auto-encoder, the proposed GAN can be used to train crack segmentation networks. By creating a manual mask, no additional crack images are required to be labeled, and data augmentation and annotation are achieved simultaneously. Our experiments are conducted on two public datasets using five segmentation models to verify the effectiveness of the proposed method. Experimental results demonstrate that the proposed method is effective for crack segmentation.

Index Terms: Pavement crack segmentation, auto-encoder, GAN, data augmentation, data annotation.

1. Introduction

Cracks are one of the most common road diseases on highways, and they need to be identified and repaired in time to prevent them from worsening and posing a greater risk to driving safety. Compared to manual inspection, using a deep learning model is more efficient [1,2]. The success of deep learning models depends on a large amount of training data. However, the current situation is that we lack training data, leaving the model in a few-shot learning environment. In the case of very few training samples, existing machine learning and deep learning models generally cannot achieve good performance, and models trained with small samples are prone to over-fitting to small samples and under-fitting to the target task. The method of augmenting data can fundamentally solve the few-sample problem, and data generation is one of them. The objective is to generate diverse and informative samples from a small number of samples. Currently, GAN (Generative Adversarial Networks) and VAE (Variational Autoencoder) are the most commonly used data generation techniques [3].

The most widely used GAN-based network structures are DCGAN (Deep Convolutional Generative Adversarial Networks), BAGAN (Balancing Generative Adversarial Networks), CGAN (Conditional Generative Adversarial Networks), ACGAN (Auxiliary Classifier Generative Adversarial Networks) and so on [4,5,6,7]. Training these methods typically requires a large amount of data. It is not possible to generate realistic images when the amount of data is small, so the generated images cannot be used for training. The crack segmentation task still requires manual labeling, even if the data generated is similar to the real data. It is very time-consuming and requires expertise in related fields to manually label pavement cracks. In contrast, our method does not require any annotation of the newly generated data.

Based on the labeled data, our method can generate corresponding crack images on the crack-free pavement according to the given label. Although the number of original images is relatively small, the generated crack images can contribute to improving the segmentation model. The generated pavement image and the given label make a pair of training data. Following the generation of the new data, we can directly use it without labeling it for training.

The proposed method is essentially divided into four steps: First, the use of an autoencoder, which generates high-dimensional features of real images. Its decoder is responsible for decoding the high-dimensional features generated by the FeatureGAN into images. Secondly, processing the pavement image with Gaussian noise. Thirdly, processing the pavement image with FeatureGAN. During the second step, the pavement image is processed with a Gaussian-noise patch and the remainder of the patch is crack-free - which makes the input to the generator. The output of the generator and the input of the discriminator are not images, but rather high-dimensional features. The fourth step is to create images. In our GAN, the final images are not directly generated, so a decoding procedure is required to generate images. In this step, the decoding part of the autoencoder is utilized. Our method converts a preprocessed image into a real crack image. The crack region is created by transforming the Gaussian noise region. Therefore, it is not required to label the generated image, as its mask is the one used during preprocessing. Moreover, the correspondence between the generated pavement image and its mask is more precise when compared with manual annotation. FeatureGAN alleviates the difficulty of fitting in the direct end-to-end image generation process. Compared with the end-to-end methods, our method provides more realistic images.

The main contributions of this paper are:

(1) We propose a dedicated GAN-based method for generating pavement crack images. The images generated by our method conform to human intuition and can improve the performance of segmentation models in the task of pavement crack segmentation. Our method does not require additional annotations on the generated images. Data augmentation and annotation are implemented simultaneously.

(2) We introduce an autoencoder in the training process of GAN, which improves the effectiveness compared with using GAN alone.

(3) Extensive experiments are performed to test the efficiency of the proposed method on the public databases. The popular segmentation methods are applied to validate the performance improvement by using the generated images.

In Section 2, we review previous studies. In Section 3, we introduce the details of the proposed method. In Section 4, we conduct comparative experiments on five different segmentation networks to measure the effectiveness of our method. In Section 5, we demonstrate the effectiveness of the main part of our approach through ablation studies. Finally, in Section 6 we conclude the paper.

2. Previous Studies

2.1. Few-shot learning

Machine learning tends to perform poorly when the amount of training data is rare [8,9]. The fundamental problem of few-sample learning is that the amount of target task data is not enough, and it is difficult to train a robust learning model. Therefore, augmenting the data can greatly help to solve the few sample problems [10]. Currently, widely used methods of data augmentation include traditional data augmentation methods (such as rotation, translation, scaling, etc.) [11,12]; splicing different foreground and background to generate diverse images [13,14,15]; using unlabeled or insufficiently labeled data [16,17]; image generation methods (e.g. GAN, VAE) [3,4,18], which generate diverse images or features.

The Mixup algorithm uses a linear interpolation of feature vectors to mix two samples, to achieve the purpose of expanding the training distribution [13]. The method adopted in Cutout randomly discards a square area of an image and adds such samples in training to improve the robustness and performance of the neural network [14]. However, Sangdoo Yun et al. proposed Cutmix which does not drop a certain area directly, since it may lead to information loss and low efficiency during training [15]. So in Cutmix, a different strategy from cutout is adopted: two samples are first blended by linear interpolation like Mixup, and then the removed regions are replaced with patches from the other image, rather than discarding directly. The Synthetic Minority Over-sampling Technique (SMOTE) algorithm use points that are close to the few samples in the feature space as new samples [19]. In semi or weakly supervised scenarios, data that is not fully annotated can also be used to train the model [16,17]. The most common image generation methods are based on GAN and VAE [3,18]. The idea of GAN and VAE is to generate new samples that are consistent with the distribution of real data by learning the distribution of images.

2.2. Generating images end-to-end with GAN

Goodfellow et al. proposed GAN in 2014. The structure is divided into two parts: the generator and the discriminator. Since the noise input into the generator is completely random, the generated images are also highly random. To control the category of the output image, CGAN, and ACGAN are proposed [5,6]. They control the categories of images produced by the generator by adding information representing the category to the input random noise. The GAN network structure has the problems of low training efficiency and instability. To this end, Alec Radford et al. proposed DCGAN, which combines GAN and CNN to improve the quality of GAN-generated images and the stability of training [4]. Martin Arjovsky et al. proposed WGAN, which further improves the stability of training, solves

the problem of mode collapse during GAN training and provides a reference value that can indicate the quality of the training process. Ishaan Gulrajani et al. proposed gradient penalty in WGAN-gp to improve the parameter truncation method in WGAN to satisfy 1-Lipschitz [20,21]. The above methods all start from random noise to generate images, and the generated images are uncertain and lack authenticity. To improve image quality, Ashish Shrivastava et al. proposed simGAN, which optimizes synthetic images to improve their authenticity [22]. In this paper, the proposed method transforms the task of fitting the pixel distribution of ordinary GANs into the task of fitting high-dimensional feature distributions, which can provide better detail and realism for the final generated images.

2.3. Augmentation based on deep features

It is difficult to obtain an effective image by completely using random noise as input, especially when the image has more details. Generation and conversion are easier when two images belong to the same image distribution or two close image distributions [23]. According to the principle of domain risk minimization, the SMOTE algorithm searches the sample space for samples in the area around the known small samples to obtain new samples [19]. Paul Upchurch et al. proposed that augmentation to images of the desired class can be achieved by performing high-level semantic transformation using simple linear interpolation of depth-wise convolutional features from pre-trained convolutions to convert one class of images to another. Antreas Antoniou et al. proposed DAGAN [24], which can increase the number of images in this category by inserting noise into the depth features of the image, and then making the image transform without changing its category. In the DAGAN framework, instead of taking category information and a noise vector as input, the generator is essentially an auto-encoder: it encodes an existing image, adds noise, and then decodes it. These methods are all based on the same distribution, using real data to generate new data. Yulin Wang et al. proposed to find a semantically meaningful direction in the feature space, and moving the depth feature of the image along this direction can change the original semantics of its attributes, thereby realizing semantic data augmentation [25]. This method does not generate training samples explicitly, but only augments the training set with deep features. Like most current data augmentation methods, these methods are easily applied to image classification tasks. Because they are all based on image category information, additional manual annotation is still unavoidable in the segmentation task to obtain the masks of the generated images. Our method uses masks to preprocess the image so that the generated image is obtained based on the same mask used in the preprocessing, and no additional manual annotation is required after the final image is generated.

3. The Proposed Method

According to the vicinal risk minimization, the conversion between two similar image distributions will be more acceptable [26]. Our proposed method exploits two distributions: images with cracks and images without cracks. We can think of the crack region dividing the pavement image into these two distributions. If we generate a crack image directly from a crack-free image, it is hard to put constraints on the location and width of the cracks. So to control these two factors, we need to find an intermediate state, which not only contains constraints on cracks but also approximates the distributions with and without cracks.

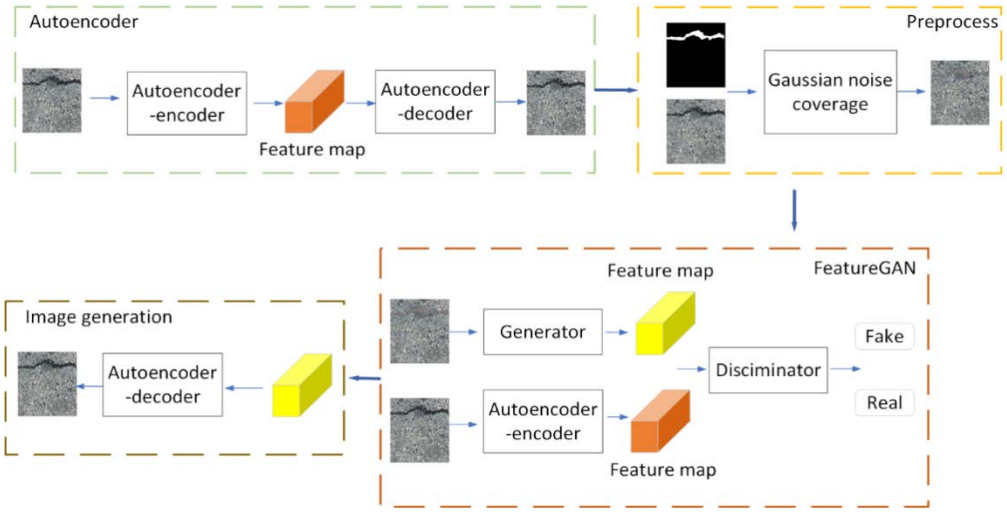

Fig.1. The overall structure of our method. It is mainly composed of four parts: a. Autoencoder, b. Image preprocessing, c. FeatureGAN network, d. Image generation.

For the general GAN structure, it is difficult to train. To make the GAN network easier to train, we divided the original GAN model into two parts: 1) The encoder of the discriminator and the decoder of the generator form an autoencoder. 2) The remaining parts of the generator and the discriminator form a new GAN model. The training is also divided into two parts: 1) Training the autoencoder. 2) Training the new GAN model.

In this section, we introduce the steps and details of our method. The overall relational structure of our network is shown in Fig.1., which consists of four parts: 1) image preprocessing, 2) training autoencoder, 3) training GAN network, and 4) image generation. The result of preprocessing is used as the input of the generator to generate high-dimensional features that are close to the true distribution. The encoder of autoencoder is used to encode real images into high-dimensional features. These two high-dimensional features are used as the input of the discriminator, respectively.

3.1. Autoencoder

An autoencoder includes two parts: encoder and decoder. Given the input $x \in \mathcal{X}$ and the feature space $h \in \mathcal{F}$, the autoencoder solves the mapping A (encoder) and B (decoder) to minimize the reconstruction error of the input feature [27]:

$$A, B = \arg\min_{A,B} \| x - B[A(x)] \|^1 \tag{1}$$

where A is the mapping from $\mathcal{X}$ to $\mathcal{F}$, B is the mapping of $\mathcal{F}$ to $\mathcal{X}$, $x \sim X$.

The structure of our autoencoder comes from CartoonGAN's generator [28]. We remove the Residual blocks, replace the 7*7 convolutional layers with three 3*3 convolutional layers, and replace InstanceNorm with BatchNorm. The structure of the autoencoder is shown in Fig.2. After an input cracked road image is encoded by the encoder, high-dimensional features of size $4 \times N \times \frac{H}{4} \times \frac{W}{4}$ are obtained, and then the reconstructed image is decoded by the decoder.

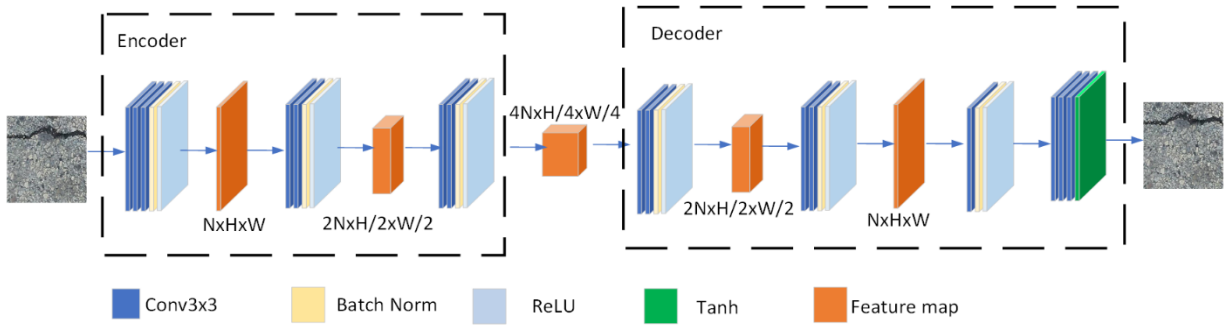

Fig.2. The structure of the autoencoder and the process of encoding and reconstructing road crack images using the autoencoder. The input is a real road image with cracks, which is encoded to obtain high-dimensional features, and then the high-dimensional features are decoded to obtain a reconstructed image.

3.2. Image preprocessing

Usually, when the number of original images is limited, the use of Gaussian noise as the input to generate the image will lead to the low quality of the generated image. Due to the high texture details in the real images, it is difficult for neural networks to well fit the difference between the Gaussian noise and the real images. Through preprocessing, a distribution close to the real image can be artificially obtained. The preprocessing process for cracked images used for network training is shown in Fig.3. According to the corresponding mask, the crack area of the original real image is covered with Gaussian noise, which leads to a synthetic image I_g.

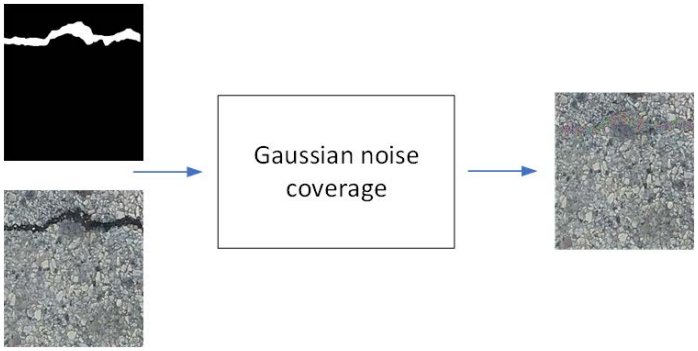

Fig.3. Preprocess real pavement images with cracks.

Based on the CRACK500 dataset, the general distribution of images is analyzed using the PCA method [29,30]. Due to limited computing resources, not all images can be used. Therefore, 100 original images are randomly selected from the crack images, Gaussian noise images, and I_g. The results figure of analyzing the images using the PCA are shown in Fig.4. As shown in Fig.4., the I_g is similar to the real image relative to the Gaussian noise.

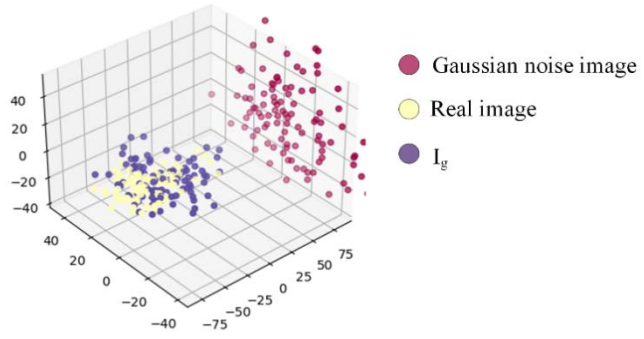

Fig.4. Data distribution plots of Gaussian noise images, real images, and I_g by using PCA. Red dots: Gaussian noise images. Yellow dots: real images. Purple dots: I_g.

3.3. FeatureGAN

FeatureGAN is used to map the I_g into high-dimensional features in the real feature space. The overall structure of the FeaturenGAN network is the same as the general GAN structure, including a generator and a discriminator. The difference is that the generator of FeatureGAN does not generate the final image, and it only outputs a high-dimensional feature. Correspondingly, the discriminator is used to judge the authenticity of high-dimensional features. As shown in Fig.5., we take the high-dimensional features generated by the generator and the encoder as the inputs of the discriminator, respectively, and then the discriminator outputs the respective discrimination results. The settings of some layers refer to CartoonGAN. For example, keeping the original activation layer unchanged, use ReLU and LeakyReLU, replacing the 7*7 convolutional layer with three 3*3 convolutional layers, adding Residual blocks to the discriminator, and adding Convolutional Block Attention Module (CBAM) to some Residual blocks [31].

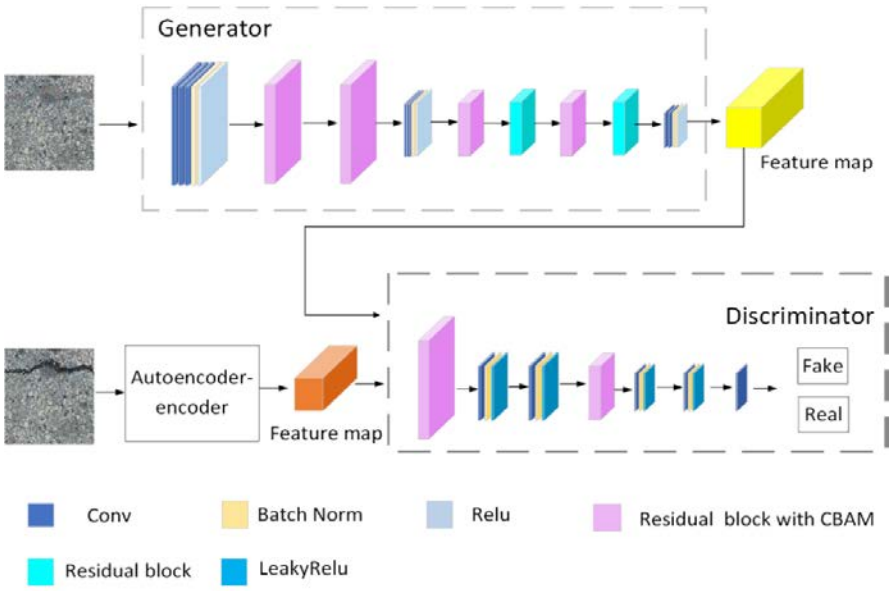

Fig.5. The structure of the GAN framework. The input for the generator is I_g and the output is high-dimensional features. The discriminator requires two inputs: high-dimensional features of the generator outputs, and the high-dimensional features of real images, which are obtained from the output of the encoder of the autoencoder.

The loss function of our GAN network refers to the loss function of WGAN-gp. The loss function used by WGAN-gp is defined in Eq. (2):

$$L = \min_{G} \max_{D} E_{\tilde{x}\sim P_g}[D(\tilde{x})] - E_{x\sim P_r}[D(x)] + \lambda_1 E_{\hat{x}\sim P_{\hat{x}}}[(\| \nabla_{\hat{x}} D(\hat{x}) \|_2 - 1)^2] \quad (2)$$

where G represents the generator and D represents the discriminator. In WGAN-gp, P_g represents the distribution of image data generated by the generator, P_r represents the distribution of real image data, $P_{\hat{x}}$ represents the distribution obtained by sampling from P_r and P_g once, and then doing random sampling again on the line connecting the two points. The default value of λ_1 is 10.

The generator is not directly used to generate images in our method. Image x in the original loss function is replaced with its corresponding high-dimensional features. In addition, the input I_g and the real image is paired. To make the generator fit the distribution of I_g to the distribution of real image faster, we add the content loss to the loss function. Therefore, the loss function is defined in Eq. (7):

$$L_{(G,D)} = \min_{G} \max_{D} E_{\tilde{h}\sim P_g}\left[D(\tilde{h})\right] - E_{h\sim P_r}[D(h)] + \lambda_2 E_{\hat{h}\sim P_{\hat{h}}}\left[\left(\| \nabla_{\hat{h}} D(\hat{h}) \|_2 - 1\right)^2\right] \quad (3)$$

$$\hat{h} = G(\hat{x}), \hat{x} \sim P_{\hat{x}} \quad (4)$$

$$h = A(x), x \sim P_x \quad (5)$$

$$L_{con} = \| \tilde{h} - h \|_2 \quad (6)$$

$$L = L_{(G,D)} + L_{con} = \min_{G} \max_{D} E_{\tilde{h}\sim P_g}\left[D(\tilde{h})\right] - E_{h\sim P_r}[D(h)] + \lambda_2 E_{\hat{h}\sim P_{\hat{h}}}\left[\left(\| \nabla_{\hat{h}} D(\hat{h}) \|_2 - 1\right)^2\right] + \| \tilde{h} - h \|_2 \quad (7)$$

where G represents the generator, D represents the discriminator, and A represents the encoder in the autoencoder. The image distribution obtained after preprocessing is denoted by $P_{\tilde{x}}$. P_x is the real image distribution, P_g is the high-dimensional feature distribution output by the generator, and P_r is the high-dimensional feature distribution corresponding to the real image. Similar to WGAN-gp, $P_{\hat{h}}$ represents the distribution obtained by sampling from P_g and P_r each, and then doing random sampling again on the line connecting the two points. The value of λ_2 used in the experiments is 0.1.

To generate cracks on crack-free images, except for adding Gaussian noise to the crack area, we don't need to modify other areas - because they contain the information of real pavement. To make the generator focus on the regions with Gaussian noise, we add an attention mechanism - CBAM into the network. The residual block structure used in the generator is shown in Fig.6. There are two 3*3 convolutional layers in the block, and this module does not change the shape of the input feature map. The structure of the attention mechanism in the residual block is shown in Fig. 7. The Attention layer contains a Convolutional Block Attention Module. There are two parts: the Channel Attention Module (CAM) and the Spatial Attention Module (SAM) [31]. While CBAM is a mixed attention mechanism module, which combines the spatial and the channel attention. Compared with the attention mechanism module that only focuses on one side, it takes both sides into account and achieves better results. In CAM, the input feature maps are sent into the shared convolutional layer after max pooling and average pooling, respectively. Then the feature maps from the shared convolutional layer are added. They are activated with sigmoid and multiplied with the input features. The output of the CAM is used as the input of the SAM. SAM performs mean pooling and maximum pooling respectively on the feature map in the channel dimension. Then we take the cancat and convolution operations. Finally, it is activated through a Sigmoid function and multiplied with the channel attention feature for the spatial attention feature map.

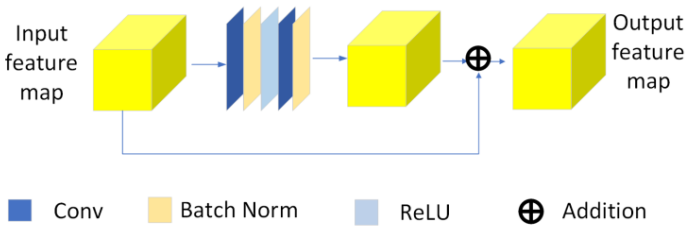

Fig.6. The structure of the residual block.

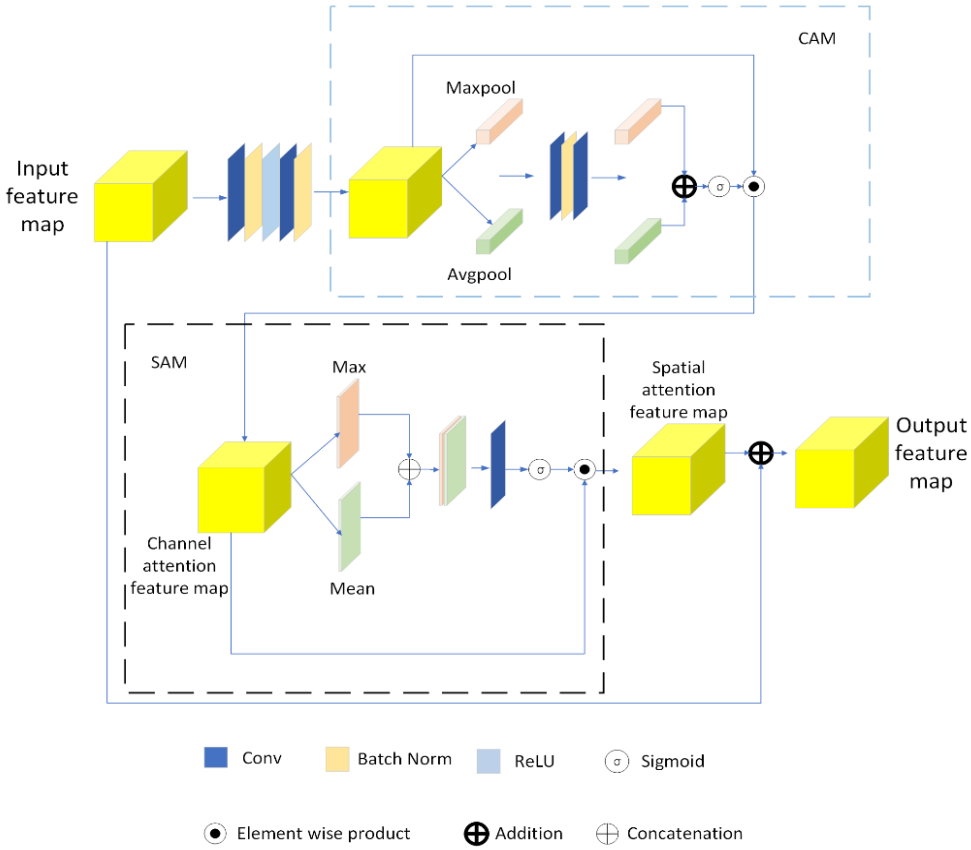

Fig.7. The structure after adding CBAM to the residual block. CAM refers to Channel Attention Module, SAM refers to Spatial Attention Module. The feature map will not change its original shape after going through CAM and SAM.

3.4. Image generation

Since the generator produces a high-dimensional feature that is close to the real distribution image, we need to design a decoder for the autoencoder to output the high-dimensional feature as the desired image. So far, we have completed the entire process of generating images. The process of generating cracked images using crack-free images and existing masks is shown in Fig.8. Firstly, we select a crack-free image and a mask image to obtain the corresponding I_g. Then, I_g is send into the trained generator to obtain high-dimensional features, which are finally decoded by the decoder of the autoencoder and mapped to the pixel space to obtain a cracke image. The crack region corresponds to the crack objects in the mask image. In this way, we can obtain a set of images that can be used for segmentation training without additional annotations to the generated images.

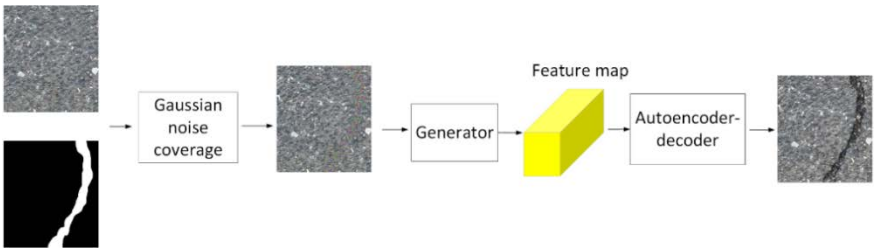

Fig.8. The process of generating crack images based on crack-free images and existing masks. The generated image and the mask image used in preprocessing can be used as training data for semantic segmentation.

4. Experiments

In this section, we will experiment with our method on two datasets, CRACK500 and GAPS384, and evaluate on five different segmentation models: FCN, Segnet, Enet, DeepCrack, and U^2-Net† [32,33,34,35,36].

4.1. Datasets

• CRACK500 is originally collected by the authors of Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection [30]. We crop each image without repetition to obtain 1061 images without cracks and 3185 training images with cracks. Due to limited computational resources, we randomly selected 10% of the images

containing cracks. That is, we use 318 cracke images to train our networks. The cropped validation set has 2461 images, and we randomly select 247 images as the validation set for evaluating the segmentation model.

• GAPS384 consists of 384 images selected from the GAPS dataset by the authors of Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection, each image size is 1920*1080 [30,37]. Due to limited computing resources, we crop each image into 12 non-repetitive regions to obtain 4,608 images and remove some images with most of the regions being manhole covers. Among them, there are 1,259 images with cracks in the training set, and the test set and validation set are combined. And there are 102 cracke images in the validation set. There are 3221 images without cracks.

4.2. Networks

To test the effectiveness of the image data generated by our method, we selected five segmentation network models for experiments. In the context of our experiments, we do not care about the original performance of these segmentation models on the dataset but focus on whether the relative metrics of the segmentation models can be improved after adding our data into the original training set.

• FCN is used for image semantic segmentation, extending image-level classification to pixel-level classification, and can use supervised training to predict the category of each pixel. It changes the final FC layer of VGG to a conv layer, where the up-sampling layer enables the network to perform pixel-level classification. We are using FCN-8S.

• Segnet consists of an encoding network and a corresponding decoding network. The encoder does not use a fully connected layer and performs convolution like FCN, so it is a lightweight network with fewer parameters[38].

• Enet is a real-time semantic segmentation network structure that uses an asymmetric Encoder-Decoder structure to reduce the amount of parameters. The model architecture consists of an initial block and five bottlenecks. The first three bottlenecks are used to encode the input image and the other two are used to decode the input image.

• Zou et al. built a DeepCrack network for ground crack detection. In this model, the multi-scale depth-wise convolutional features learned in the hierarchical convolution stage were fused to capture fine crack structure.

•U^2-Net$t^{\dagger}$ is a relatively smaller version, with fewer filters than the ordinary U^2-Net network [36]. U^2-Net is a network structure based on Unet. The author refers to FPN and Unet [39,40]. It has achieved good results for segmenting the objects and also has good real-time performance.

4.3. Evaluate Metric

To evaluate the segmentation effect, we adopt two commonly used segmentation evaluation metrics: IOU (Intersection over Union) and F-measure [41]. IOU is the ratio of the coincident area of the actual area and the predicted area to the overall area of the two areas (in Eq. (8)). The maximum value is 1 and the minimum value is 0. The F-measure is based on the harmonic mean of Recall and Precision [42, 43] (in Eq.(11)).

$$\mathrm{IOU} = \frac{\mathrm{TP}}{\mathrm{FN+FP+TP}} \tag{8}$$

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \tag{9}$$

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \tag{10}$$

$$\mathrm{F-mearsure} = 2 \times \frac{\mathrm{Precision \times Recall}}{\mathrm{Precision+Recall}} \tag{11}$$

where TP, FN, and FP represent True Positives, False Negative, and False Positive, respectively.

4.4. Results

We set up three training sets: original real training set, original real training set with image data augmented by traditional methods, and original real training set with image data augmented by our method. They are denoted as R, R+C, and R+ours, respectively. Traditional enhancement methods refer to color transformations and rotations. The parameters of the color transformation method are set as: the random selection range of brightness, contrast, and saturation are in [0.5, 1.5], the random selection range of hue is in [-0.1, 0.1], and the rotation angle is 90, 180, 270, and 360.

4.4.1. Results on CRACK500 Dataset

Experiments on the CRACK500 dataset. The real training set we used has 318 images and the validation set has 247 images. 701 images are augmented for it using the traditional method and our method, respectively. Three training sets were obtained: R: 318 images, R+C: 1019 images, and R+ours: 1019 images. These three training sets are used for training separately on FCN, Segnet, Enet, DeepCrack, U^2-Net$t^{\dagger}$. In the training process of each network, except for the different training sets, other settings are the same, such as training times, network hyperparameters, learning rate, etc. After the training is completed, the indicators are tested on the same validation set, that is, the real image validation set. The test results of the indicators are shown in Table 1. and Table 2. Compared with using only real images as the

training set, using our method can increase the IOU of FCN, Segnet, Enet, DeepCrack, and U^2-Net† segmentation by 3.24%, 4.83%, 3.43%, 4.95%, and 2.07%, respectively. F-measure increased by 2.95%, 4.59, 3.02%, 4.58%, and 1.91% respectively. Compared with traditional data augmentation methods, our method also brings more improvement in metrics. Compared with training on R+C, training on R+Ours can improve the IOU of FCN, Segnet, Enet, DeepCrack, U^2-Net†segmentation by 1.98%, 1.22%, 1.01%, 1.13%, 1.08%, F-measure increased by 1.87%, 1.27%, 1.01%, 0.47%, and 0.98% respectively.

Table 1. IOU performance achieved on CRACK500 by five semantic segmentation methods using R, R+C, and R+Ours respectively.

Segmentation architectures	R	R+C	R+Ours
FCN	0.4555	0.4681	**0.4879**
Segnet	0.4673	0.5034	**0.5156**
Enet	0.4727	0.4969	**0.5070**
DeepCrack	0.4360	0.4742	**0.4855**
U^2-Net†	0.5158	0.5257	**0.5365**

Table 2. F-measure performance achieved on CRACK500 by five semantic segmentation methods using R, R+C, and R+Ours respectively.

Segmentation architectures	R	R+C	R+Ours
FCN	0.5994	0.6102	**0.6289**
Segnet	0.6092	0.6424	**0.6551**
Enet	0.6144	0.6365	**0.6466**
DeepCrack	0.5733	0.6144	**0.6191**
U^2-Net†	0.6545	0.6638	**0.6736**

4.4.2 Results on GAPS384 Dataset

Experiments are performed on the GAPS384 dataset. The real training set we used contains 1259 images and the validation set contains 102 images. 1307 images are augmented using traditional methods and our method, respectively. Three training sets are set as: R: 1259 images, R+C: 2566 images, and R+ours: 2566 images. These three train sets are used to train FCN, Segnet, Enet, DeepCrack, U^2-Net† respectively. We keep the environment and parameter settings for each training of the same network unchanged, and then the metrics are evaluated on the same validation set. The test results of the indicators are shown in Table 3. and Table 4. Since the cracks in the images in CRACK500 are wider and more obvious, while the cracks in the images in GAPS384 are thin and dimly lit, the segmentation results on the GAPS384 dataset are worse than those on CRACK500. Compared with the case of using only real images as the training set, using our method can increase the IOU results of FCN, Segnet, Enet, DeepCrack, U^2-Net† by 10.37%, 4.68%, 4.77%, 14.92%, 4.11%, and F- measure is increased by 12.45%, 3.73%, 5.63%, 19.98%, and 4.44% respectively. Besides, our method is more efficient than traditional methods. Compared with traditional methods, the IOU of FCN, Segnet, Enet, DeepCrack, and U^2-Net† are increased by 1.42%, 5.16%, 3.17%, 6.66%, and 1.39%, and F-measure by 1.42%, 4.66%, 3.73%, 8.51%, and 1.96%.

Table 3. IOU results achieved on GAPS384 by five semantic segmentation methods using R, R+C, and R+Ours, respectively.

Segmentation architectures	R	R+C	R+Ours
FCN	0.2309	0.3204	**0.3346**
Segnet	0.2910	0.2862	**0.3378**
Enet	0.3086	0.3246	**0.3563**
DeepCrack	0.0932	0.1758	**0.2424**
U^2-Net†	0.3521	0.3793	**0.3932**

Table 4. F-measure results achieved on GAPS384 by five semantic segmentation methods using R, R+C, and R+Ours, respectively.

Segmentation architectures	R	R+C	R+Ours
FCN	0.3566	0.4669	**0.4811**
Segnet	0.4206	0.4113	**0.4579**
Enet	0.4414	0.4604	**0.4977**
DeepCrack	0.1536	0.2683	**0.3534**
U^2-Net†	0.4908	0.5156	**0.5352**

From the results in Table 1, Table 2, Table 3, and Table 4, we can see that after adding the enhanced data, the effects of the five segmentation models are improved to varying degrees, and our method is better than the traditional method. Since there is a single type of road in GAPS384, while there are many types of roads in the CRACK500 dataset, we observe that the results obtained from the experiments on the GAPS384 dataset are improved more.

4.4.3 Qualitative results

In this section, we will show the segmentation results produced by the DeepCrack segmentation network. In the same environment, we use the three training sets of R, R+C, and R+Ours to train DeepCrack, and then get the results of segmentation on the validation set. The results of the experiments on the CRACK500 dataset are shown in Fig.9. As can be seen from Fig.9., the network trained with the images generated by our method can have a better effect on the finer crack area, and the segmentation details are more accurate. The results of the experiments on the GAPS384 dataset are shown in Fig.10. It can be seen from Fig.10. that if data augmentation is not used, the trained network segmentation effect is very poor, and it is almost impossible to recognize such very thin cracks. Both the traditional data augmentation method and our data augmentation method can improve the segmentation performance of the network. The network trained with R+Ours can identify more crack regions.

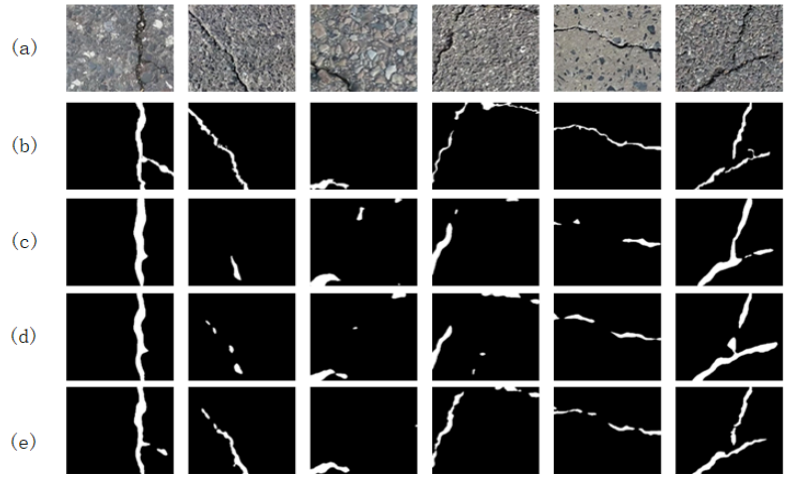

Fig.9. Visual effects of experiments on the CRACK500 dataset using DeepCrack. (a) original image, (b) ground truth, (c) segmentation result of the network trained with R (d) segmentation result of the network trained with R+C, and (e) segmentation result of the network trained with R+Ours.

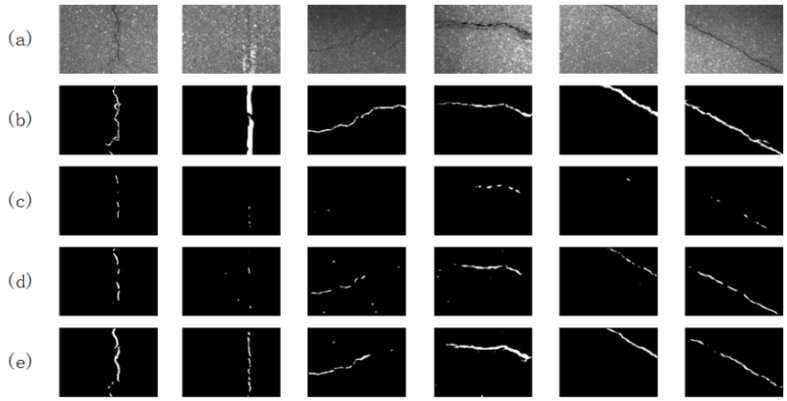

Fig.10. Visual effects of experiments on the GAPS384 dataset using DeepCrack. (a) original image, (b) ground truth, (c) segmentation result of the network trained with R, (d) segmentation result of the network trained with R+C, and (e) segmentation result of the network trained with R+Ours.

5. Ablation Study

In this section, we perform experiments on the CRACK500 dataset to verify the effects of each part of the proposed method, which include (1) The role of generating cracks on crack-free images, (2) The using of disassembled autoencoder from GAN to assist training, (3) The using of attention mechanism, (4)The role of using GAN after preprocessing.

5.1. The effects of creating cracks on crack-free images

We take the real crack and crack-free images together as a training set, denoted as R+R^*, to train the segmentation network. The ground truth of the crack-free image is an all-black image. The joined images are from the 701 crack-free images used during the preprocessing in Section 4.4.1. FCN, Segnet, ENET, DeepCrack, U^2-Net$t^\dagger$ are trained separately,

and then IOU and F-measure metrics are tested in the validation set. The parameter settings of the segmentation network and the validation set are the same as those used in Section 4.4.1. The test indicators are shown in Table 5. and Table 6. By adding crack-free images, it produces a negative effect on the segmentation effect of FCN, ENET, and U^2-Net†. The IOU results of Segnet and Deepcrack segmentation are increased by 1.72% and 1.04%, respectively, and the F-measure is increased by 1.18% and 0.91%, respectively. They are not as good as the traditional method and our method.

Table 5. IOU performance achieved on CRACK500 by five semantic segmentation mthods using R, R+R^*, R+C, and R+Ours respectively.

Segmentation architectures	R	R+R^*	R+C	R+Ours
FCN	0.4555	0.4386	0.4681	**0.4879**
Segnet	0.4673	0.4845	0.5034	**0.5156**
Enet	0.4727	0.4450	0.4969	**0.5070**
DeepCrack	0.4360	0.4464	0.4742	**0.4855**
U^2-Net†	0.5158	0.4654	0.5257	**0.5365**

Table 6. F-measure performance achieved on CRACK500 by five semantic segmentation methods using R, R+R^*, R+C, and R+Ours respectively.

Segmentation architectures	R	R+R^*	R+C	R+Ours
FCN	0.5994	0.5811	0.6102	**0.6289**
Segnet	0.6092	0.6210	0.6424	**0.6551**
Enet	0.6144	0.5877	0.6365	**0.6466**
DeepCrack	0.5733	0.5824	0.6144	**0.6191**
U^2-Net†	0.6545	0.5987	0.6638	**0.6736**

5.2. Use the disassembled autoencoder from GAN to assist in training

We first add the decoder from our autoencoder to the generator, and the encoder to the discriminator, which makes a GAN network in the general form. The input to the generator is I_g and the output is the generated crack image. The input to the discriminator is the real crack image or the generated crack image. The loss function adopts the loss function of the original WGAN-gp. Gradient penalty weight λ_1=0.1. Both the generator and discriminator learning rates are set to be 2e-4, and the batch size is set to be 8. The images obtained after training for 1500 epochs are shown in Figure 11(b). The GAN network is kept under the same training parameters, and we disassemble the autoencoder from the original network to assist the training. The results are shown in Fig.11.(c). It can be seen that the images generated by our method produce better visual effects. This is the reason why we use the generator of the proposed method for generating feature map instead of the crack image.

Fig.11. Results obtained using the general GAN structure and our GAN structure training. (a) I_g, (b) results generated by training on general GAN architecture, (c) results generated by training on our GAN architecture.

In this general GAN form, 701 images are generated and added to the real image, denoted as R+G. R+G is used to train FCN, Segnet, ENET, DeepCrack, and U^2-Net† respectively. The training and validation strategies are the same as in Section 4.4.1. The experimental results are shown in Table 7. and Table 8. After the images obtained using the general GAN structure are added to the training, the segmentation result shows a negative effect in FCN, Segnet, ENet, and DeepCrack. The improvement of IOU and F-measure for U^2-Net† segmentation is only 0.31% and 0.49%.

Table 7. IOU performance achieved on CRACK500 by five semantic segmentation methods using R, R+G, R+C, and R+Ours respectively.

Segmentation architectures	R	R+G	R+C	R+Ours
FCN	0.4555	0.4490	0.4681	**0.4879**
Segnet	0.4673	0.4288	0.5034	**0.5156**
Enet	0.4727	0.4246	0.4969	**0.5070**
DeepCrack	0.4360	0.4353	0.4742	**0.4855**
U^2-Net†	0.5158	0.5189	0.5257	**0.5365**

Table 8. F-measure performance achieved on CRACK500 by five semantic segmentation methods using R, R+G, R+C, and R+Ours respectively.

Segmentation architectures	R	R+G	R+C	R+Ours
FCN	0.5994	0.5911	0.6102	**0.6289**
Segnet	0.6092	0.5729	0.6424	**0.6551**
Enet	0.6144	0.5676	0.6365	**0.6466**
DeepCrack	0.5733	0.5720	0.6144	**0.6191**
U^2-Net†	0.6545	0.6594	0.6638	**0.6736**

5.3. Using the attention mechanism

We train the network with all attention layers removed and retain attention layers under the same conditions. Gradient penalty weight λ_2=0.1. Both the generator and discriminator learning rates are set to be 2e-4, and the batch size is set to 8. The visualization of the attention map and its effects are shown in Fig.12. In experiments, it can be found that after adding the attention mechanism, the crack area of the generated image contains more details. The content of the crack region of the image generated without the attention mechanism is relatively plain.

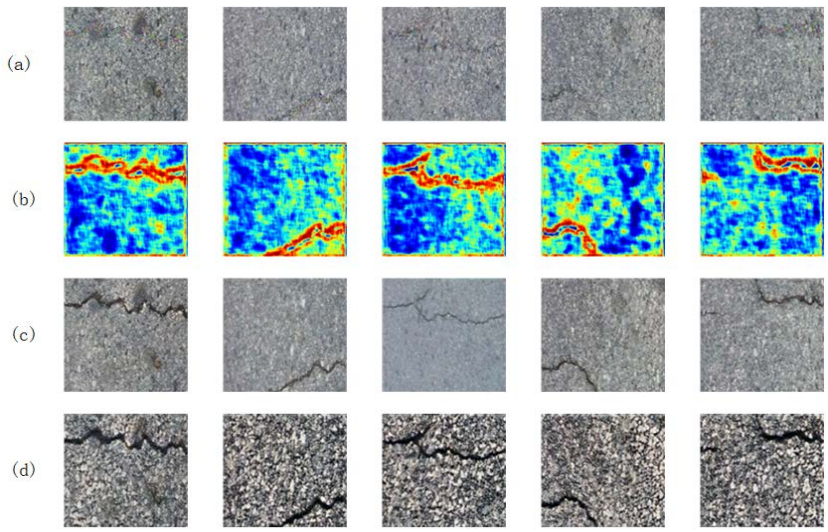

Fig.12. Visualization of the attention maps and their effects: (a) images obtained by preprocessing (b) the attention maps of the generator (c) the results obtained by adding CBAM (d) the results obtained without CBAM.

The 701 cracked pavement images generated by the network without the attention layer and the real images are used as the training set, denoted as R+W. FCN, Segnet, Enet, DeepCrack, U^2-Net† are trained on this training set. Then we test IOU and F- measure. The I_g, the training, and the validation processes are the same as those used in Section 4.4.1. The verified index results are shown in Table 9. and Table 10. Compared with the original training set, the network-enhanced data without using the attention layer increases the IOU of FCN, Segnet, DeepCrack, and U^2-Net† segmentation results by 1.19%, 2.87%, 2.36%, 0.12%, and F-Measure increased by 0.99%, 2.60%, 2.44%, and 0.07% respectively. The improvement effect is not as good as the traditional augmentation method and the network that retains the attention layer. And the IOU and F-Measure of the Enet segmentation results are reduced by 1.40% and 1.33%, respectively.

Table 9. IOU performance achieved on CRACK500 by five semantic segmentation methods using R, R+W, R+C, and R+Ours respectively.

Segmentation architectures	R	R+W	R+C	R+Ours
FCN	0.4555	0.4674	0.4681	**0.4879**
Segnet	0.4673	0.4960	0.5034	**0.5156**
Enet	0.4727	0.4587	0.4969	**0.5070**
DeepCrack	0.4360	0.4596	0.4742	**0.4855**
U^2-Net†	0.5158	0.5170	0.5257	**0.5365**

Table 10. F-measure performance achieved on CRACK500 by five semantic segmentation methods using R, R+W, R+C, and R+Ours respectively.

Segmentation architectures	R	R+W	R+C	R+Ours
FCN	0.5994	0.6093	0.6102	**0.6289**
Segnet	0.6092	0.6352	0.6424	**0.6551**
Enet	0.6144	0.6011	0.6365	**0.6466**
DeepCrack	0.5733	0.5977	0.6144	**0.6191**
U^2-Net†	0.6545	0.6552	0.6638	**0.6736**

5.4. The role of using GAN after preprocessing

We take I_g used in Section 4.4.1 as extra training samples. The mask corresponding to each image is the corresponding mask in the preprocessing. Now, the entire training set is denoted as R+I_g. We use it to train FCN, Segnet, ENET, DeepCrack, and U^2-Net† respectively. Except for the different training set, other settings are the same as Section 4.4.1. The test results are shown in the Table 11. and Table 12. The results show that adding I_g directly to the training set does not bring about a significant improvement. Conversely, the metrics for Segnet, Enet, and U^2-Net†declined. In particular, the IOU of U^2-Net† dropped to 2.26%, and F-measure dropped to 4.32%. This suggests that using GAN after preprocessing is beneficial.

Table 11. IOU performance achieved on CRACK500 by five semantic segmentation methods using R, R+I_g., R+C, and R+Ours respectively.

Segmentation architectures	R	R+I_g.	R+C	R+Ours
FCN	0.4555	0.4571	0.4681	**0.4879**
Segnet	0.4673	0.4977	0.5034	**0.5156**
Enet	0.4727	0.4118	0.4969	**0.5070**
DeepCrack	0.4360	0.4506	0.4742	**0.4855**
U^2-Net†	0.5158	0.0226	0.5257	**0.5365**

Table 12. F-measure performance achieved on CRACK500 by five semantic segmentation methods using R, R+I_g., R+C, and R+Ours respectively.

Segmentation architectures	R	R+I_g.	R+C	R+Ours
FCN	0.5994	0.5997	0.6102	**0.6289**
Segnet	0.6092	0.6385	0.6424	**0.6551**
Enet	0.6144	0.5568	0.6365	**0.6466**
DeepCrack	0.5733	0.5808	0.6144	**0.6191**
U^2-Net†	0.6545	0.0432	0.6638	**0.6736**

6. Conclusion

In this paper, we proposed a new data augmentation method applied to the task of road crack segmentation. Compared to generating cracked images from disordered noise, it is easier to obtain cracked images based on non-cracked images. The proposed method generates more realistic images under the same conditions by adjusting the general GAN structure to the proposed structure for training. With the inclusion of an attention mechanism, the proposed augmentation method achieved better results. The semantic segmentation results of five effective segmentation models are trained and evaluated on CRACK500 and GAPS384 to prove the effectiveness the proposed method.

Experimental results indicate that adding our generated images to the training set can enhance the performance of crack segmentation on both datasets.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61772435), Sichuan Highway Science and Technology Project (No. 2019-01) and the Fundamental Research Funds for the Central Universities (No. 2682021ZTPY069).

References

[1] T. Cao, J. Hu, and S. Liu, “Enhanced edge detection for 3d crack segmentation and depth measurement with laser data,” International Journal of Pattern Recognition and Artificial Intelligence, p. 2255006, 2022.
[2] S. A. H. Tabatabaei, A. Delforouzi, M. H. Khan, T. Wesener, and M. Grzegorzek, “Automatic detection of the cracks on the concrete railway sleepers,” International Journal of Pattern Recognition and Artificial Intelligence, vol. 33, no. 09, p. 1955010, 2019.
[3] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” arXiv preprint arXiv:1312.6114, 2013.
[4] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” arXiv preprint arXiv:1511.06434, 2015.
[5] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” arXiv preprint arXiv:1411.1784, 2014.
[6] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” in International conference on machine learning. 1em plus 0.5em minus 0.4em PMLR, 2017, pp. 2642–2651.
[7] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, “Bagan: Data augmentation with balancing gan,” arXiv preprint arXiv:1803.09655, 2018.
[8] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Generalizing from a few examples: A survey on few-shot learning,” ACM computing surveys (csur), vol. 53, no. 3, pp. 1–34, 2020.
[9] Biserka Petrovska, Igor Stojanovic, Tatjana Atanasova-Pacemska,"Classification of Small Sets of Images with Pre-trained Neural Networks", International Journal of Engineering and Manufacturing, Vol.8, No.4, pp.40-55, 2018.
[10] J. Bennilo Fernandes, Kasiprasad Mannepalli, " Enhanced Deep Hierarchal GRU & BILSTM using Data Augmentation and Spatial Features for Tamil Emotional Speech Recognition", International Journal of Modern Education and Computer Science, Vol.14, No.3, pp. 45-63, 2022.
[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in neural information processing systems, vol. 25, 2012.
[12] I. Sato, H. Nishimura, and K. Yokoi, “Apac: Augmented pattern classification with neural networks,” arXiv preprint arXiv:1505.03229, 2015.
[13] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” arXiv preprint arXiv:1710.09412, 2017.
[14] T. DeVries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” arXiv preprint arXiv:1708.04552, 2017.
[15] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, “Cutmix: Regularization strategy to train strong classifiers with localizable features,” in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 6023–6032.
[16] T. Remez, J. Huang, and M. Brown, “Learning to segment via cut-and-paste,” in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 37–52.
[17] G. Li, J. Wan, S. He, Q. Liu, and B. Ma, “Semi-supervised semantic segmentation using adversarial learning for pavement crack detection,” IEEE Access, vol. 8, pp. 51 446–51 459, 2020.
[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Advances in neural information processing systems, vol. 27, 2014.
[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” Journal of artificial intelligence research, vol. 16, pp. 321–357, 2002.
[20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” Advances in neural information processing systems, vol. 30, 2017.
[21] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in International conference on machine learning. 1em plus 0.5em minus 0.4em PMLR, 2017, pp. 214–223.
[22] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, “Learning from simulated and unsupervised images through adversarial training,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2107–2116.
[23] Thuy Nguyen Thi Thu, Lich Nghiem Thi, Nguyen Thu Thuy, Toan Nghiem Thi, Nguyen Chi Trung, "Improving Classification by Using MASI Algorithm for Resampling Imbalanced Dataset", International Journal of Intelligent Systems and Applications, Vol.11, No.10, pp.33-41, 2019.
[24] F. H. K. d. S. Tanaka and C. Aranha, “Data augmentation using gans,” arXiv preprint arXiv:1904.09135, 2019.
[25] Y. Wang, G. Huang, S. Song, X. Pan, Y. Xia, and C. Wu, “Regularizing deep networks with semantic data augmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.
[26] O. Chapelle, J. Weston, L. Bottou, and V. Vapnik, “Vicinal risk minimization,” Advances in neural information processing systems, vol. 13, 2000.
[27] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, “Deep learning (vol. 1) cambridge,” 2016.

[28] Y. Chen, Y.-K. Lai, and Y.-J. Liu, "Cartoongan: Generative adversarial networks for photo cartoonization," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9465–9474.
[29] H. Abdi and L. J. Williams, "Principal component analysis," Wiley interdisciplinary reviews: computational statistics, vol. 2, no. 4, pp. 433–459, 2010.
[30] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, and H. Ling, "Feature pyramid and hierarchical boosting network for pavement crack detection," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 4, pp. 1525–1535, 2019.
[31] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 3–19.
[32] 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.
[33] 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, no. 12, pp. 2481–2495, 2017.
[34] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, 2016.
[35] 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, no. 3, pp. 1498–1512, 2018.
[36] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, and M. Jagersand, "U2-net: Going deeper with nested u-structure for salient object detection," Pattern Recognition, vol. 106, p. 107404, 2020.
[37] M. Eisenbach, R. Stricker, D. Seichter, K. Amende, K. Debes, M. Sesselmann, D. Ebersbach, U. Stoeckert, and H.-M. Gross, "How to get pavement distress detection ready for deep learning? a systematic approach," in 2017 international joint conference on neural networks (IJCNN). 1em plus 0.5em minus 0.4em IEEE, 2017, pp. 2039–2047.
[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.
[39] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2117–2125.
[40] 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. 1em plus 0.5em minus 0.4em Springer, 2015, pp. 234–241.
[41] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," International journal of computer vision, vol. 88, no. 2, pp. 303–338, 2010.
[42] R. Margolin, L. Zelnik-Manor, and A. Tal, "How to evaluate foreground maps?" in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 248–255.
[43] N. Liu, J. Han, and M.-H. Yang, "Picanet: Learning pixel-wise contextual attention for saliency detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3089–3098.

Authors' Profiles

Xinkai Zhang received his B.S. degree in software engineering from Taiyuan University of Science and Technology, Taiyuan, Shanxi, China in 2019. Currently, he is pursuing his master degree in School of computing and artificial intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China since 2020. His research interests include data augmentation, image segmentation, and computer graphics.

Bo Peng is an Associate Professor in the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, China. She received the M.S. degree from the Department of Computer Science, University of Western Ontario (UWO) in 2008 and the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University in 2012. From Aug. 2011 to Jan. 2012, she worked as a Research Assistant in the Department of Computing, The Hong Kong Polytechnic University. Her research interests include image segmentation, segmentation quality evaluation, and pattern recognition. She is a member of IEEE.

Zaid Al-Huda received his master degree in artificial intelligence from University of, Hyderabad, India in 2015, and the Ph.D. degree in computer science and technology from School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, Sichuan, China, in July 2021. Currently, he is a Postdoctoral Research Fellow with School of Computing and Artificial Intelligent, Southwest Jiaotong University, Chengdu, Sichuan, China since August 2021. His research interests include computer vision, deep learning applications, image segmentation, segmentation quality evaluation, and pattern recognition.

Donghai Zhai received his B.E. degree in hydrogeology and engineering geology, M.S. degree in computer science and technology, and Ph.D. degree in traffic information engineering and control from Southwest Jiaotong University, China, in 1997, 2000, and 2003 respectively. From 2003 to 2005 he was employed at IBM China Research Laboratory performing system analyses and architecture design. Since 2006 he has been associated with Southwest Jiaotong University in the department of software engineering. He is also a visiting scholar at Louisiana State University (LSU), Baton Rouge, USA. His research interests include autonomous driving, digital image processing, computer vision, and pattern recognition.

How to cite this paper: Xinkai Zhang, Bo Peng, Zaid Al-Huda, Donghai Zhai, " FeatureGAN: Combining GAN and Autoencoder for Pavement Crack Image Data Augmentations", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.14, No.5, pp. 28-43, 2022. DOI:10.5815/ijigsp.2022.05.03