Journal of Computer Science and Artificial Intelligence
ISSN: 3078-8242 (Print), ISSN: 3078-2309 (Online) | Vol. 3, No. 1, 2025

Research on Road Damage Recognition Based on VanillaNet Neural Network

Han Wang *, Min Li, Xuyang Gu, Xiang Liu, Boyu Yang, Kai Chen

School of Mechanical and Automotive Engineering, Jinken College of Technology, Nanjing, China

* **Corresponding author**: sangam_333@163.com

Abstract: This paper presents a highly efficient and accurate model for identifying road conditions, specifically distinguishing between normal roads and potholes in images. Given the challenges posed by potholes, such as their prevalence in shaded areas, cluttered backgrounds, difficulty in annotation, and diverse shapes, we initiate the process by normalizing the dataset. We select the VanillaNet neural network as our foundational training architecture due to its elegant and simplistic design, which maintains exceptional performance in computer vision tasks. VanillaNet commences with several layers incorporating nonlinear activation functions, gradually eliminating these layers as training progresses, facilitating easy integration while preserving inference speed. This network surpasses contemporary models in both efficiency and accuracy, making it an ideal choice for our analysis. The quantitative results demonstrate promising outcomes, laying a solid foundation for future research endeavors. Subsequently, we train the developed VanillaNet model, employing deep training techniques to enhance its performance. The model is comprehensively evaluated based on accuracy, speed, and generalization capabilities to ensure it meets our requirements. In the object detection experiments, the model's performance is assessed using four key metrics.

Keywords: VanillaNet Neural Network, Nonlinear Activation Functions, Deep Training Techniques.

1. Introduction

Road maintenance and safety are critical concerns in urban infrastructure management. The timely detection and repair of road damages, particularly potholes, are essential to prevent accidents and ensure smooth traffic flow. However, identifying potholes in images is a challenging task due to their varied appearances, often being in shaded or cluttered areas, and the lack of consistent annotation standards. Traditional methods of road inspection are often time-consuming and labor-intensive, necessitating the development of automated systems for efficient road damage detection.

In recent years, deep learning techniques, particularly convolutional neural networks (CNNs), have shown remarkable success in various computer vision tasks, including object detection and image classification. Among the plethora of neural network architectures available, VanillaNet stands out for its simplicity and efficiency. VanillaNet's design philosophy emphasizes elegance and minimalism while maintaining high performance, making it an attractive option for tasks requiring both speed and accuracy.

Huang Meng and colleagues [1] presented Van-YOLOv8, a lightweight network based on yolov8, specifically designed for the detection of markers on recycled chips. To counteract the adverse effects of small, low-resolution markers on the precision of deep learning models, an approach to enhance resolution was initially employed. Tao Zhiyong et al. [2] put forth the dynamic snake convolutional Neck (dsconvn-Neck) to capture detailed fine-grained features, thereby enhancing the detection of minor target defects. Subsequently, they utilized the C2f Efficient Multiscale Attention (C2MA) module to minimize background noise and bolster resistance against background interference. Zhou Kang et al. [3] developed a technique grounded in the you-only-look-one version-10 (YOLOv10s) framework for the automated detection of glass curtain wall damage in high-rise structures. Tailored improvements were implemented in the backbone, neck, detection layer, and loss function to tackle issues related to imprecise damage localization and the detection of minor damage in glass curtain walls. Xu Shizhou et al. [4] introduced an enhanced Yolov5 vehicle target detection model, termed YOLO-HV, which integrates Vision Transformer (VIT) as its backbone. This model seeks to address the limitations of conventional CNN networks in integrating contextual information, thereby improving multi-scale target recognition performance. Liu Jing and Ren Qifeng [5] formulated a novel theorem concerning timing stability and provided an estimation of the stabilization time. Drawing from this theorem and employing Lyapunov's direct method, they derived two distinct sufficiency criteria to guarantee that neural networks, under Hölder's continuous conditions, achieve synchronization within a fixed time frame based on 1-paradigm and 2-paradigm, respectively.

This paper aims to leverage the capabilities of VanillaNet to develop a robust model for road damage recognition, specifically focusing on distinguishing between normal road surfaces and potholes. By normalizing the dataset and employing deep training techniques, we seek to enhance the model's performance, ensuring it meets the stringent requirements of real-world applications. The proposed model is evaluated based on multiple criteria, including accuracy, speed, and generalization ability, to provide a comprehensive assessment of its effectiveness.

2. Neural Network Modeling

2.1. Introduction to Neural Networks

Backpropagation (BP) neural network has good self-learning, self-adaptation and generalization ability, but it is easy to fall into local minima and has a slow convergence speed. Therefore, this paper proposes a method to optimize the BP algorithm based on Genetic Algorithm (GA) to

improve the training speed of BP and to overcome the disadvantage that BP is easy to fall into local minima. The UCI dataset is used here for experimental analysis, and the experimental results show that, compared with the BP algorithm and the method of learning connection weights using only GA, the method in this paper combines GA and BP to train neural networks better; does not easily fall into local minima; the trained network has better generalization ability; and has good stability performance.

First, it is very important to choose the right neural network architecture. The multilayer perceptron (MLP) architecture is the neural network topology of choice for many researchers, since it is the oldest neural network architecture and is compatible with all training software. However, despite the widespread use of MLPs in many applications, subsequent sections of this paper will show that their functional performance tends to be poor compared to other topologies, such as the Bridged Multilayer Perceptron (BMLP). The BMLP allows for cross-layer connectivity, which can increase the flexibility of the network.

Second, sizing the neural network is also critical. Choosing a neural network that is too large may lead to difficulties in training convergence, while choosing a neural network that is too small may limit its interpolation ability and its ability to handle new patterns. In the section "Using Minimum Network Size", we will show how to choose the appropriate network size to ensure effective training.

Finally, choosing the appropriate learning algorithm is also crucial. The Error Backpropagation (EBP) algorithm is one of the most popular learning algorithms, but it suffers from a few problems in terms of training speed and quality of results. For example, the training process of EBP may require 100 to 1,000 times more iterations than other more advanced algorithms (e.g., Levenberg-Marquardt or neuron-to-neuron algorithms). This not only reduces the training speed, but often leads the algorithm to fall into a local optimum without finding the optimal neural network solution.

2.2. Neural Architecture

This section shows in detail the architecture of VanillaNet, which takes 6 layers as an example. For the backbone, we use a 4×4×3×C convolutional layer with a step size of 4 to map an image with 3 channels to a feature with C channels. In stages 1, 2 and 3, a maximum pooling layer with a step size of 2 is used to reduce the size and feature map and the number of channels is increased by 2. In stage 4, we do not increase the number of channels as it follows the average pooling layer. The last layer is the fully connected layer used to output the classification results. The kernel size of each convolutional layer is 1×1 because our goal is to use the minimum computational cost for each layer while preserving the feature map information. Activation functions are applied after each 1×1 convolutional layer. Batch normalization is also added after each layer to simplify the training process of the network.

2.3. Neural Network Training

In this paper, we propose a VanillaNet neural network based road pothole detection model, which is built in a 6-layer VanillaNet architecture. For the backbone, we use a 4×4×3×C convolutional layer with a step size of 4 to map an image with 3 channels to a feature with C channels. In stages 1, 2 and 3, a maximum pooling layer with a step size of 2 is used to reduce the size and feature map and the number of channels is increased by 2. In stage 4, we do not increase the number of channels as it follows the average pooling layer. The last layer is the fully connected layer used to output the classification results. The kernel size of each convolutional layer is 1 × 1 because our goal is to use the minimum computational cost for each layer while preserving the feature map information. Activation functions are applied after each 1 × 1 convolutional layer. Batch normalization is also added after each layer to simplify the training process of the network.

1) Deep Training Strategy

The main idea of the deep training strategy is to train two convolutional layers at the beginning of the training with the activation function instead of a single convolutional layer. The activation function is gradually reduced to identity mapping as the training period increases. At the end of the training, the two convolutions can be easily merged into a single convolution to reduce the inference time. For the activation function A(x) (which can be a common function such as Tanh), we combine it with the constant mapping, which can be formulated as:

$$A'(x) = (1 - \lambda)A(x) + \lambda x \quad (1)$$

Where λ is the nonlinear parameter of the activation function A′ (x) used for balancing corrections.

Denote the number of current iteration rounds and the number of deep training iteration rounds as e and E, respectively. We set λ = e/E. Thus, at the beginning of training (e = 0), A′(x)=A(x), which indicates that the network has strong nonlinearity. When training converges, we have A′(x)=x, which indicates that there is no activation function between the two convolutional layers.

We first convert each batch normalization layer and its previous convolution into a single convolution. We denote $W \in \mathbb{R}^{C_{out}\times(C_{in}\times k\times k)}$, $B \in \mathbb{R}^{C_{out}}$ as the weight and bias matrix of a convolutional kernel with Cin input channels, Cout output channels, and a kernel size of k. The scale, bias, mean and variance in batch normalization are denoted as $\gamma, \beta, \mu, \sigma \in \mathbb{R}^{C_{out}}$ The combined weight and bias matrices are:

$$W_i' = \frac{\gamma_i}{\sigma_i} W_i, B_i' = \frac{(B_i - \mu_i)\gamma_i}{\sigma_i} + \beta_i \quad (2)$$

Where subscript i ∈ {1, 2, …, Cout} denotes the value in the ith output channel.

After merging the convolution with the batch normalization, we start merging the two 1×1 convolutions. Denoting $x \in \mathbb{R}^{C_{in}\times H\times W}$ and $y \in \mathbb{R}^{C_{out}\times H'\times W'}$ as input and output features, the convolution can be formulated as:

$$y = W * x = W * im2col(x) = W * X \quad (3)$$

Where * denotes the convolution operation, - denotes matrix multiplication, and $X \in \mathbb{R}^{(C_{in}\times 1\times 1)\times(H'\times W')}$ is derived from the im2col operation for converting the input to a matrix corresponding to the shape of the kernel. Fortunately, for the 1 × 1 convolution, we find that the im2col operation turns out to be a simple shaping operation, since there is no need to have overlapping sliding kernels.

Therefore, the weight matrices of the two convolutional layers are denoted as W1 and W2, and the two convolutional formulas without activation functions are formulated as:

$$y = W^1 * (W^2 * x) = W^1 * W^2 * im2col(x) = (W^1 * W^2) * X \quad (4)$$

Thus, 1×1 convolution can be merged without increasing the speed of inference.

2) Activation functions

In fact, there are two ways to improve the nonlinearity of a neural network: stacking the nonlinear activation layers or increasing the nonlinearity of each activation layer, while the tendency of existing networks is to choose the former, which leads to high latency when there is an excess of parallel computing power. A straightforward idea to improve the nonlinearity of activation layers is stacking. Serial stacking of activation functions is the core idea of deep networks. Instead, we turn to stacking activation functions simultaneously. Denote the single activation function for input x in a neural network as a(x), which can be a commonly used function such as ReLU and Tanh. the simultaneous stacking of A(x) can be formulated as:

$$A_s(x) = \sum_{i=1}^{n} a_i \, A(x + b_i) \tag{5}$$

Where n denotes the number of stacked activation functions, and ai, bi are the scale and deviation of each activation to avoid simple accumulation. The nonlinearity of the activation functions can be greatly enhanced by stacking them simultaneously. Equation can be viewed mathematically as a series, which is an operation that adds many quantities together.

To further enrich the approximation ability of the series, we enable the series-based function to learn global information by changing the inputs of its neighbors. Specifically, given an input feature $x \in \mathbb{R}^{C\times H\times W}$, where H, W, and C are its width, height, and number of channels, the activation function equation is:

$$A_s(x_{h,w,c}) = \sum_{i,j\in(-n,n)} a_{i,j,c} A(x_{i+h,j+w,c} + b_c) \tag{6}$$

Where $h \in \{1, 2, \ldots, H\}$, $w \in \{1, 3, 2, \ldots\ldots, W\}$ and $c \in \{1, 2, .., C\}$. It is easy to see that the level-based activation function As(x) degenerates to a pure activation function A(x) when $n = 0$, which implies that the proposed method can be regarded as a general extension of existing activation functions. We use ReLU as the basic activation function to construct our sequences because it is effective for inference in GPUs.

We further analyze the computational complexity of the proposed activation function compared to the corresponding convolutional layer. For a convolutional layer with K kernel sizes, Cin input channel and Cout output channel, the computational complexity is:

$$O(CONV) = H \times W \times C_{in} \times C_{out} \times k^2 \tag{7}$$

And the computational cost of its tandem activation layer is:

$$O(SA) = H \times W \times C_{in} \times n^2 \tag{8}$$

So we get:

$$\frac{O(CONV)}{O(SA)} = \frac{H\times W\times C_{in}\times C_{out}\times k^2}{H\times W\times C_{in}\times n^2} = \frac{C_{out}\times k^2}{n^2} \tag{9}$$

As an example, the fourth stage in VanillaNet-B, where Cout = 2048, k = 1 and n = 7, has a ratio of about 84. In conclusion, the computational cost of the proposed activation function is still much lower than that of convolutional layers. More experimental complexity analysis will be shown in the next section.

3. Model Training

In this section, the VanillaNet neural network model (see Fig. 1) is created as in the previous section, and the performance of the model is enhanced using deep training techniques in VanillaNet, and then, the model is thoroughly evaluated based on its accuracy, speed, and generalization ability to ensure that it meets our needs. In the target detection experiments, this paper evaluates the performance of the model in terms of four metrics: mAP (mean Average Precision), FPS (Frames Per Second), and the number of parameters (Params) and computations (GFLOPs) of the model. mAP refers to the average precision of the model for all the targets it detects in the test set. The higher the value of the index, the better the model's detection effect on different categories of targets. FPS reflects the number of images processed by the model per unit time, the higher the value of FPS, the better the real-time performance of the model. The number of parameters of the model is used to evaluate the complexity of the model, the smaller the number of parameters is, the lighter the model is. GFLOPs is used to evaluate the computational efficiency of the model, the lower the value of GFLOPs is, the higher the computational efficiency of the model is.

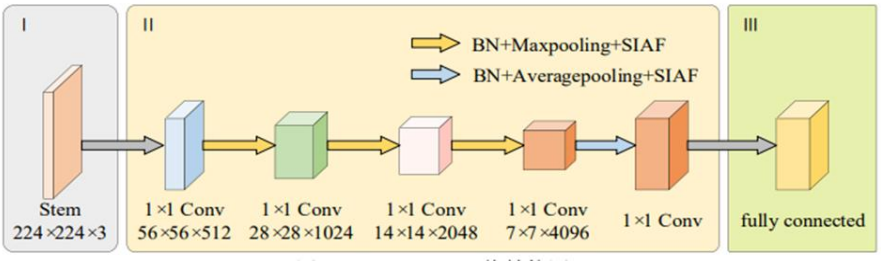

Figure 1. VanillaNet Network Architecture Diagram

The result loss value of loss function is mainly used to describe the deviation between the prediction result and the real result, which can effectively judge the accuracy of a model, so the loss value is mainly used in the experimental process to reflect the training process of the model. In this experiment, the calculation of confidence loss is mainly changed and combined with IoU and GIoU respectively, such a practice is to explore the effect of the change of IoU on the loss function. During the experiment, although the calculation of the loss function is changed, it is still possible to choose to use the Loss value as a criterion. As can be seen in Fig. 2, the training loss and validation loss of the model gradually decrease as the number of iteration rounds increases, and the validation loss is smaller, both of which end up smoothing out below 0.2. Fig. 3 shows that with the increase in the number of iteration rounds, the training analysis accuracy of the model first increases and then gradually smooth, and the model validation analysis accuracy is also gradually smooth at the end, and is greater than the training analysis accuracy.

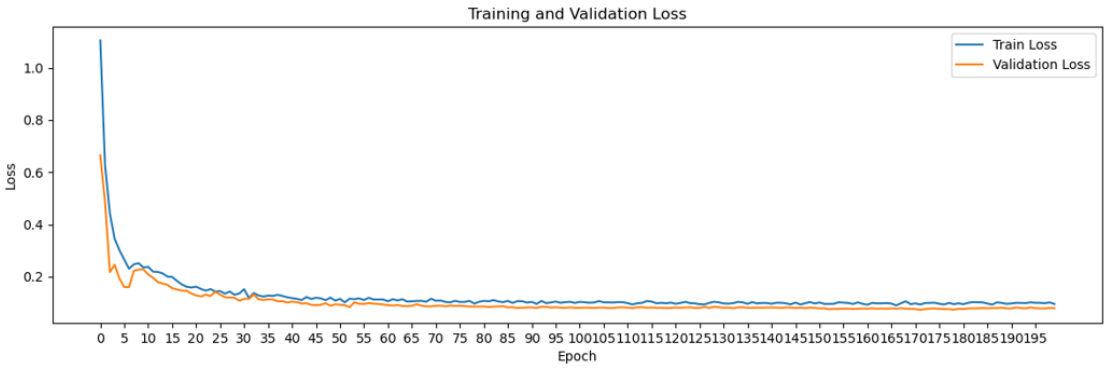

Figure 2. VanillaNet network analysis loss

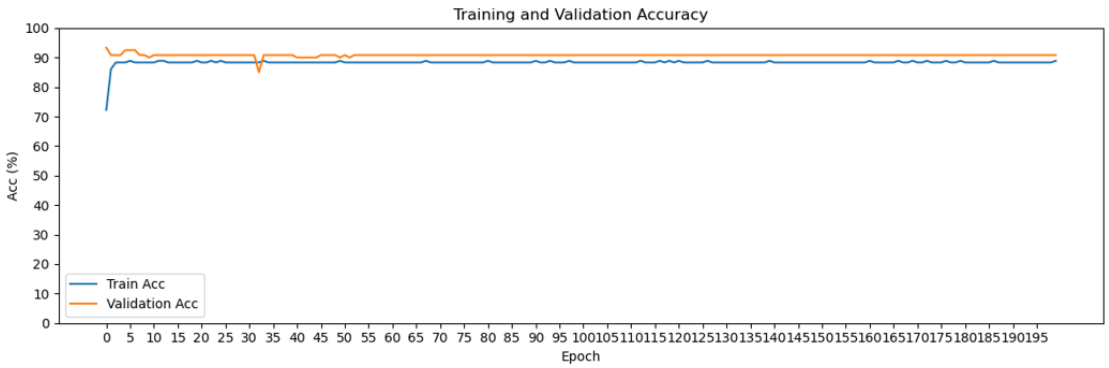

Figure 3. VanillaNet network analysis accuracy

4. Conclusion

In this study, we have successfully developed a high-performance model for road damage recognition using the VanillaNet neural network. By addressing the challenges associated with pothole detection, such as their diverse shapes, shaded locations, and cluttered backgrounds, we have demonstrated the effectiveness of our approach. The normalization of the dataset and the application of deep training techniques have significantly enhanced the model's performance, enabling it to achieve high accuracy and speed while maintaining robust generalization capabilities.

The VanillaNet architecture, with its emphasis on simplicity and efficiency, has proven to be an excellent choice for this task. Its ability to gradually eliminate nonlinear layers during training not only facilitates easy integration but also preserves inference speed, making it highly suitable for real-time applications. The comprehensive evaluation of the model using four key metrics has provided valuable insights into its strengths and areas for potential improvement.

Overall, our research contributes to the field of road maintenance and safety by offering a reliable and efficient solution for automated road damage detection. Future work may focus on expanding the dataset to include a wider variety of road conditions and exploring additional techniques to further enhance the model's performance. The findings of this study lay a solid foundation for the development of more advanced systems for road infrastructure management.

References

[1] A Study on Enhancing Chip Detection Efficiency Using the Lightweight Van-YOLOv8 Network. Comput. Mater. Contin. 79, 531–547 (2024).

[2] Tao, Z., He, Y., Lin, S., Yi, T. & Li, M. SnakeNet: An adaptive network for small object and complex background for insulator surface defect detection. Comput. Electr. Eng. 117, 109259 (2024).

[3] Zhou, K. et al. An improved YOLOv10 algorithm for automated damage detection of glass curtain-walls in high-rise buildings. J. Build. Eng. 101, 111812 (2025).

[4] Xu, S., Zhang, M., Chen, J. & Zhong, Y. YOLO-HyperVision: A vision transformer backbone-based enhancement of YOLOv5 for detection of dynamic traffic information. Egypt. Inform. J. 27, 100523 (2024).

[5] Liu, J. & Ren, Q. Fixed-time synchronization of coupled neural networks under Hölder continuous nonlinear activation function. Eur. J. Control 81, 101165 (2025).