machines

MDPI

Article

DMnet: A New Few-Shot Framework for Wind Turbine Surface Defect Detection

Jinyun Yu [1], Kaipei Liu [1], Liang Qin [1,*], Qiang Li [2], Feng Zhao [2], Qiulin Wang [2], Haofeng Liu [1], Boqiang Li [1], Jing Wang [1] and Kexin Li [1]

[1] School of Electrical Engineering and Automation, Wuhan University, Wuhan 430072, China; yujinyun0707@126.com (J.Y.); kpliu@whu.edu.cn (K.L.); 2018302070013@whu.edu.cn (H.L.); 2016302540098@whu.edu.cn (B.L.); wang-jing@whu.edu.cn (J.W.); li-kexin@whu.edn.cn (K.L.)
[2] State Grid Information & Telecommunication Group Co., Ltd., Beijing 102211, China; liqiang@sgitg.sgcc.com.cn (Q.L.); feng_zhao@sgitg.sgcc.com.cn (F.Z.); wqllwq@126.com (Q.W.)
* Correspondence: qinliang@whu.edu.cn; Tel.: +86-189-8617-2977

Abstract: In the field of wind turbine surface defect detection, most existing defect detection algorithms have a single solution with poor generalization to the dilemma of insufficient defect samples and have unsatisfactory precision for small and concealed defects. Inspired by meta-learning ideology, we devised a cross-task training strategy. By exploring the common properties between tasks, the hypothesis space shrinks so that the needed sample size that satisfies a reliable empirical risk minimizer is reduced. To improve the training efficiency, a depth metric-based classification method is specially designed to find a sample-matching feature space with a good similarity measure by cosine distance. Additionally, a real-time feedback session is innovatively added to the model training loop, which performs information enhancement and filtering according to the task relevance. With dynamic activation mapping, it alleviates the information loss during traditional pooling operations, thus helping to avoid the missed detection of small-scale targets. Experimental results show that the proposed method has significantly improved the defect recognition ability under few-shot training conditions.

Keywords: wind turbine surface defect detection; few-shot scenario; meta-learning

Citation: Yu, J.; Liu, K.; Qin, L.; Li, Q.; Zhao, F.; Wang, Q.; Liu, H.; Li, B.; Wang, J.; Li, K. DMnet: A New Few-Shot Framework for Wind Turbine Surface Defect Detection. *Machines* **2022**, *10*, 487. https://doi.org/10.3390/machines10060487

Academic Editor: Davide Astolfi

Received: 19 May 2022
Accepted: 13 June 2022
Published: 16 June 2022

1. Introduction

Blades and towers are the key components of wind turbines (WTs). The former carries the main load to obtain wind energy, while the latter plays a vital role in supporting and absorbing the vibration of the generator set. A survey has suggested that these two parts account for 42% of the total cost of a wind turbine [1]. However, WTs usually operate in remote fields. Due to the harsh environment and complex working conditions, the surfaces of wind turbine blades and towers may be prone to defects such as cracks, coating peeling, edge corrosion, blisters, and pits [2,3], which will reduce the energy conversion rate and have a negative impact on the power generation quality and the unit life. If the restoration work is not completed in time, it may become a major safety hazard, or even a serious power accident in the long run. Therefore, the fast and efficient detection of WT surface defects has become an urgent task.

Through a review of the literature for WT defect detection, it was found that the earlier traditional detection methods mainly rely on different types of signal sensors for diagnosis, such as ultrasonic [4], vibration [5], acoustic emission [6], and infrared thermography [7] techniques. However, the above methods with high cost have poor stability in harsh environments, resulting in low detection efficiency. Therefore, the mainstream research has gradually begun to focus on image-based visual detection.

With the wide application of UAV technology, the current research on WT surface defect detection based on UAV images is mainly divided into two categories: artificial

feature methods and deep learning methods. Methods based on artificial features first extract features such as contour, texture, and color (ex. Haar-like [8], LBP [9], SIFT [10], and HOG [11]) of image samples through traditional image processing techniques and then train commonly used classifiers such as SVM (support vector machine) [12], Adaboost [13], and Random Forest [14] for defect recognition. Considering the distribution, severity, and development trend of cracks, Peng and Liu [15] proposed a crack analysis method combining Wiener filtering and the adaptive median filtering algorithm, which can effectively reduce the negative impact of motion blur. Chen and Shen [16] investigated a three-point slope deviation method to monitor the operating condition of WT blades in a visual inspection way, but they did not consider the impact of blade pitch, non-fault deformation, and bad weather on the system. Long Wang et al. [17] select Haar-like features to depict cracks on WT blades. Additionally, an extended cascade classifier is developed to perform crack detection on UAV inspection images, using a stretchable scan window to locate crack regions. However, pre-processing and post-processing of the signal is still complex and time-consuming, and detection is limited to specific damage types.

In spite of having a certain degree of expressiveness, the manually extracted features lack sufficient effective information due to the low signal-to-noise ratio, so that the precision cannot meet the requirements of industrial applications. Meanwhile, it is time-consuming to select appropriate features.

Applying the automatic feature extraction of CNN networks, deep learning methods [18–22] have received considerable attention recently; their accuracy and efficiency are superior to those based on artificial features. Using the VGG-11 model, Xu et al. [23] provide an automatic feature extractor for defect blades and employ the "alternating direction method of the multiplier" algorithm for model compression to reduce the requirements for hardware equipment. However, as only an 11-layer CNN was employed, Xu's classification experiments yielded unfavorable results. Yang [24] used the ResNet50 algorithm to identify multiple types of leaf defects, achieving a recognition accuracy of more than 95%, while this result was obtained by an unbalanced dataset with only 10% of defect samples, which is an inadequate validation. Qiu et al. [25] designed a WT detection system YSODA with an improved YOLOv3 algorithm. They modified the YOLO architecture to support multi-scale feature pyramids in CNN and expanded the number of samples by image enhancement, to improve the performance on small-size defect detection. On the downside, the speed of the system is slower than before due to the increased network complexity. Shortly afterwards, Shihavuddin et al. [26] provide an automatic recommendation system by exploiting a faster RCNN algorithm. To adapt to high-resolution images and difficult-to-separate samples, an augmentation step called the "multi-scale pyramid and patching scheme" is proposed to achieve higher sensitivity.

Despite a proliferation of studies that have been published in this field, the following issues remain to be discussed:

- In practical applications, owing to the generally rare defect data, deep network training is prone to overfitting. Collecting large-scale annotation data from scratch, however, is time-consuming and expensive. Existing studies are limited to augmentation from a data perspective, which performs conventional deep learning by directly expanding the number of samples. However, few scholars have drawn on any systematic research into considerations in model composition or into training patterns. Knowing how to construct a robust detection model under a few-shot data scenario without sample expansion is a primary concern.
- The diversity of shooting time, angle, and distance of UAVs increases the difficulty of equipment image defect detection, which ensures the existing deep learning-based methods have low recognition accuracy for extremely small and concealed defects.

For the above situation, we developed a new few-shot training framework for wind turbine surface defect detection, and we present our phased design and field trials in this article. The contributions of this article are as follows:

(1) Inspired by meta-learning, a cross-task training strategy is designed pertinently for WT surface-defect recognition. Using the MVTec dataset as raw material, a series of different but related tasks are constructed, to find common guidelines of defect identification in all things rather than learning the given data itself. Without expanding the amount of original data, we achieve high-precision defect recognition with only 20 training samples.

(2) To alleviate the huge computational cost of traditional classifiers with supervised labels, we establish a non-parametric connection between samples by cosine distance in a high-dimensional vector space and expect the network to learn a general similarity metric that maintains identity across the entire data. It helps to quickly distinguish similarities and differences for unfamiliar data, thereby improving the training efficiency.

(3) To tap the potential of identifying small defects and hidden defects, the depth feature map is additionally overlaid with an equivalent soft mask map to enhance task-relevant information and filter redundant information according to task relevance, helping to make real-time feedback corrections in model training.

(4) In this article, class activation mapping (CAM) technology is innovatively integrated into each round of training. This study explores, for the first time, the dynamic interpretability of feature space in the training state, to understand and uncover the secrets of the "black box" in deep learning.

2. Methodology

2.1. Motivation

In supervised machine learning, the learning process is often approximated by fitting a function f to a dataset D. According to the empirical risk minimization theorem [27,28], since the true optimal hypothesis $\hat{h}$ is unknown, we give a hypothesis space H and expect to find the best approximation hypothesis h_I in H that satisfies the empirical risk minimization through model training.

Therefore, for a dataset D with I training samples, the total error between the empirical risk and the actual expected risk is as follows:

$$\mathbb{E}\left[R(h_I) - R\left(\hat{h}\right)\right] = \mathbb{E}\left[R(h^*) - R\left(\hat{h}\right)\right] + \mathbb{E}[R(h_I) - R(h^*)] = e_{app}(H) + e_{est}(H, I) \quad (1)$$

The approximation error $e_{app}(H)$ represents how close the function in H can approximate the true optimal hypothesis $\hat{h}$, and estimation error $e_{est}(H, I)$ measures the effect of replacing expected risk minimization with empirical risk minimization. Obviously, the total error is affected by hypothesis space H and sample size I.

As shown in Figure 1a,b, for common supervised learning, when there is sufficient training data with supervised information, h_I can provide a good approximation. Conversely, in the case of few training samples, $e_{est}(H,\ I)$ may exceed a reasonable range. h_I is no longer reliable, and overfitting occurs [29].

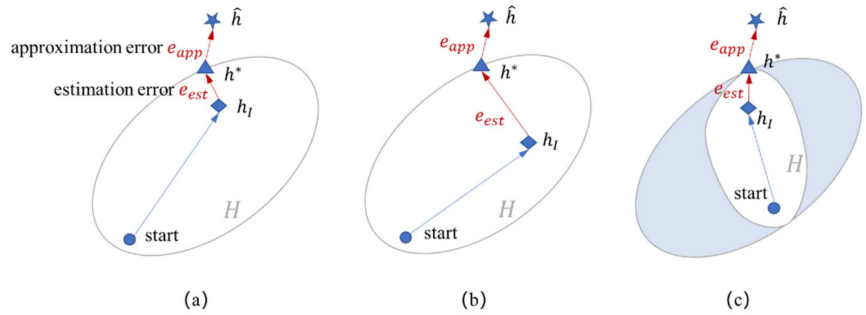


Figure 1. The total error under different circumstances. (**a**) sufficient sample data (**b**) insufficient sample data (**c**) meta-learning.

In response to this problem, most studies have mostly focused on performing data-augmentation techniques. However, since the augmentation strategy is only applicable to a specific data set, it is not easy to migrate to other datasets. Moreover, the effect relies heavily on the quality of newly generated data. It is reasonable to believe that such research does not tackle the problem radically.

Interestingly, we found the following phenomenon in human learning: when people learn a series of different but related tasks, they can draw inferences from each other by distilling the cross-task knowledge and skills. For new tasks, this kind of generalization ability ensures that the model has rules to follow and gets started quickly.

Inspired by the above idea, meta-learning, also known as learning to learn [30], enables the machine to find a set of effective "learning paths" in the past abundant tasks, so that it can generalize well on unseen tasks. Particularly, the generalization achieves good performance only with few training samples.

As Blumer argues [31], the lower bound on the number of training samples required for fitting a function can be estimated. Specifically, if sample size I satisfies:

$$I \geq \frac{1}{-\ln(1-\varepsilon)}\left(\ln(|H|) + \ln\left(\frac{1}{\delta}\right)\right) \tag{2}$$

Any hypothesis h in H that is consistent with the objective function f on dataset D will guarantee with probability $(1-\delta)$ that the error rate of predictions on future data is lower than ε.

It is clear that the sample size required depends only on three variables: the expected error rate ε, the guaranteed probability δ, and the size of hypothesis space H, regardless of the objective function f and the data's distribution. Therefore, under the premise of ensuring that H always contains f, the sample size can be reduced by shrinking H.

Meta-learning simultaneously learns on multiple tasks with common properties. For example, different image recognition tasks follow the translation invariance and rotation invariance of images. In Figure 1c, these common attributes prune the hypothesis space H, and narrow the search area of optimation parameters, thereby reducing the training complexity of the training and helping model adapt well to few-shot scenarios.

That is, meta-learning is practical for solving few-shot learning problems.

2.2. Introductory Definition

The idea of meta-learning is divided into two main phases: meta-train and meta-test [32] (Figure 2). In meta-train, we expect the model to acquire generalizability across tasks in the task distribution $P(T)$. To solve this problem in practice, a set of respective independent source tasks is usually sampled from $P(T)$, where each task $T \in P(T)$ has a respective task-related training and respective testing data, i.e., $D_T = \{D_s, D_q\}$, and D_s, D_q are called the support set and query set, respectively. The purpose of training is to optimize the meta-parameter θ such that

$$\min_{\theta} \mathbb{E}_{T \in P(T)} L(D_T; \theta) \tag{3}$$

where $L(D_T; \theta)$ denotes the loss over the data set D_T with the model, and the optimal solution θ^* can be regarded as the learned cross-task knowledge (or meta-knowledge). In the meta-test, the model is tested on a new task $T_{new} \in P(T)$ that is disjointed from the data used for the source tasks in meta-train.

In contrast to traditional machine learning, the fast adaptation to the training set of the target task T_new benefits from regarding the meta-knowledge θ^* as prior knowledge. θ^* could be the estimation of the model's initial parameters, hyperparametric optimization strategies, the neural network architecture design, etc.

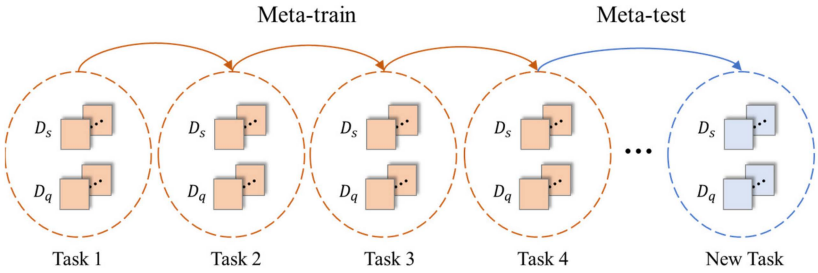


Figure 2. The sketch map of meta-learning.

2.3. DMnet: A New Few-Shot Training Framework

2.3.1. The Overall Looking

Inspired by meta-learning ideology, we construct a new few-shot training framework called DMnet for wind turbine defect detection. The basic flow chart is suggested in Figure 3. To absorb prior knowledge, the machine learns from a large number of tasks to obtain high-level generalization capability. In each task, the pre-processed image data undergoes feature extraction by CNN to get a deep feature map, which is finally input to the metric module for category determination. Further, the proposed dynamic activation mapping strategy monitors this process and provides real-time feedback and corrections. After that, the machine becomes a more powerful learner. For our target task, i.e., WT surface defect detection, defect recognition and location on unseen samples can be achieved with just only a small amount of supervised sample fine-tuning.

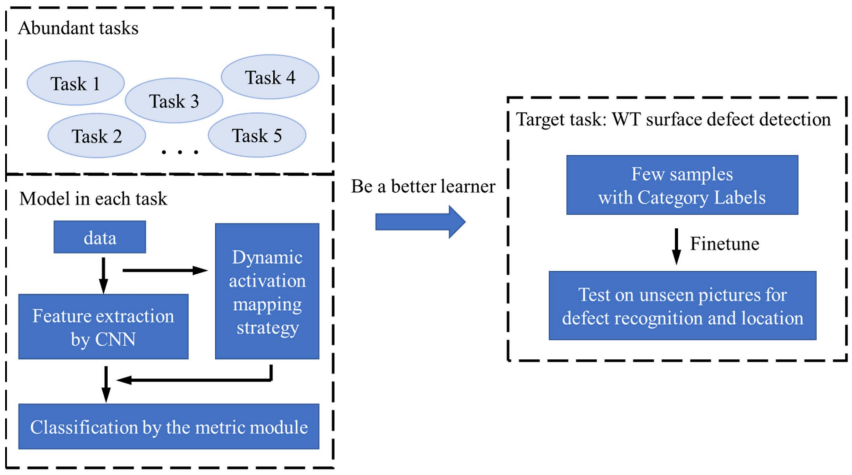


Figure 3. The basic flow chart of Dmnet.

2.3.2. The Cross-Task Training

From Section 2.3.1, meta-learning requires the support of multiple different but related tasks, each of which has its own training and test sets. To realize the snap recognition of WT surface defects, it is necessary to construct multiple tasks with similar settings to this, which will participate in meta-training as training sets.

In the paper, the MVTec anomaly detection dataset [33], including 15 items such as toothbrushes, leather, pills, wood, etc., is selected as data support for meta-learning (Figure 4).

As depicted in Figure 5, we specifically design the following three-stage process: meta-training, fine-tuning, and meta-testing. Dataset D is divided into three mutually exclusive meta-sets: meta-training set $D_{meta-train}$, meta-validation set $D_{meta-valid}$, and meta-testing set $D_{meta-test}$. MVTec is used to build the first two parts, in which the wood dataset builds $D_{meta-valid}$, while the other 14 items build $D_{meta-train}$.

Figure 4. The MVTec anomaly detection dataset.

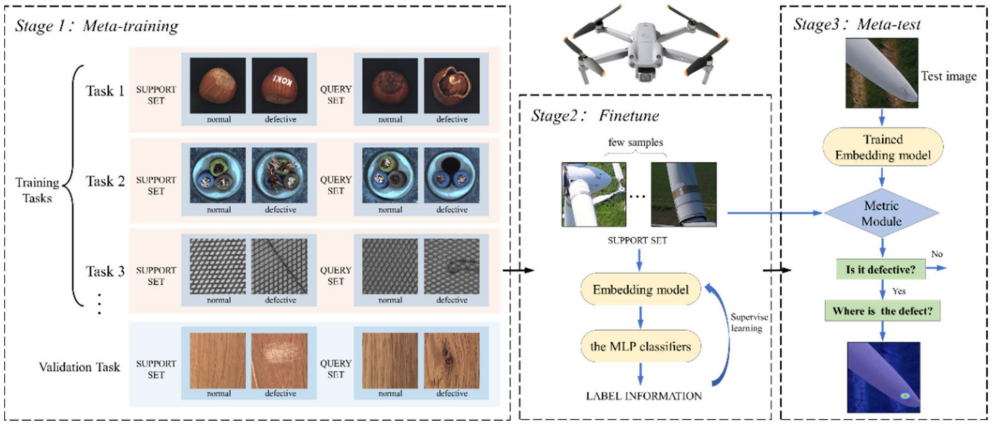


Figure 5. The detailed three-stage process.

In the first stage, we construct several tasks for meta-training and expect the model to learn the task-to-task generalization ability: meta-training set $D_{meta-train}$ is a dataset collection of n_{tasks} tasks, i.e., $D_{meta-train} = (D_{task1}, D_{task2}, \ldots)$. Each task dataset $D_{task} = \left(D_{Support}, D_{query}\right)$, where support set $D_{Support}$ and query set D_{query} correspond to the training process and testing process in a single task, respectively. Applying random sampling for each class, we get n_{train_s} sheets for support set and n_{train_q} for query set. Meta-validation set $D_{meta-valid}$ is built to identify defects of wood. Similarly, we have n_{val_s} sheets for support set and n_{val_q} for query set, to observe the trend of model generalization performance on a new task over model training.

However, we observed that the defects in the MVTec dataset are relatively simple and obvious, while WT surface defects often have a complex background, which makes the defect identification more challenging. Inevitably, the difference between our target task and either task in the meta-training stage will be significantly greater than the within-group task difference. To alleviate the negative influence, the fine-tuning in the second stage performs supervised learning on the support set in meta-test set $D_{meta-test}$.

Finally, the target few-shot task is tested. The meta-test set $D_{meta-test}$ serves our target task—WT surface defect detection, with n_{wind_s} sheets as the support set and n_{wind_q} sheets as the query set.

It should be noted that n_{val_s}, n_{train_s}, and n_{wind_s} must be small numbers in order to match the few-sample circumstance. In general, the size selection for both the support set and the query set in the meta-training stage requires comprehensive consideration of computing resources. The principle is to improve the information utilization rate as much as possible with limited resources.

2.3.3. Feature Extraction

By analyzing the characteristics of WT's drone images, it can be found that the defects such as cracks and edge corrosion are in various irregular forms, which are difficult to describe uniformly by a kind of specific feature. This raises demand for the semantic analysis ability of the model. Moreover, factors such as the large variety of defect scales, the

variable perspective of images and the complex background also increase the recognition difficulty. Therefore, we choose CNN for the feature extraction in this case.

Simulating the structure of the human brain, CNN-based deep learning techniques build multilayer neural networks to extract low-level and high-level features from the input data layer by layer [34]. Through this hierarchical method, it effectively establishes the mapping relationship from the underlying signal to the high-level semantics. While improving the target recognition rate, it avoids the complicated operation of manually designed features in the development of traditional defect recognition.

The convolutional layer and the pooling layer are the most important components of CNN. Combining these two elements can form a variety of CNN feature extractors. The shallow neurons express edge and angle information, while the role of the convolutional layer is to gradually extract more abstract structural information by increasing the local perceptual field. The convolution kernel is calculated as follows:

$$y_{p\prime}^{L} = f_{ac}\left(\sum_{i \in M_{p\prime}} y_{p}^{L-1} * k_{pp\prime}^{L} + b_{p\prime}^{L}\right) \tag{4}$$

where $M_{p\prime}$ denotes the input feature map; y_{p}^{L-1} represents the convolution kernel to which the pth feature map of the previous layer is connected with the $p\prime$th feature map of the L-th layer; $*$ is the convolution operation; $b_{p\prime}^{L}$ is the bias; and $f_{ac}(\cdot)$ is the activation function.

The pooling layer, also called the down-sampling layer, reduces the dimensionality of the feature map by compressing the image, which can improve model's noise immunity and maintain some invariance of the features (rotation, translation, stretching, etc.).

2.3.4. The Metric Classification Module

During the meta-training, the support set in each task helps the model build a classification pattern, and the query set validates and tunes the strengths and weaknesses of the pattern. Inspired by deep metric learning, we expect the model to learn the essential associations of things.

Therefore, our meta-learner aims to obtain a great cross-task feature space for similarity measurement, which performs well on new tasks. We develop a metric module for classification based on cosine distance measurement. The higher the similarity score between a test sample and the given data, the more likely they belong to the same class.

In Figure 6, the process of classifying defects in a single task is as follows:

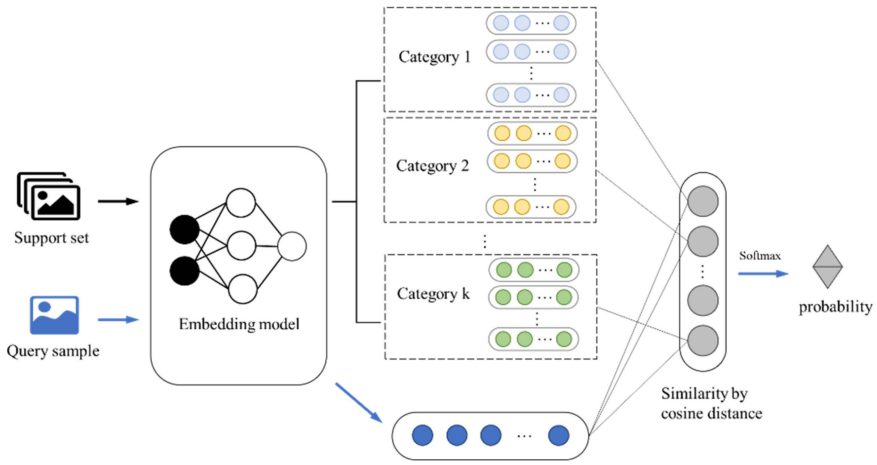


Figure 6. The metric classification module.

Define a task t with support set $D_{support}^{t}$ and query set D_{query}^{t}.

First, the embedding model is constructed by a convolutional neural network (CNN). In this way, each sample in $D_{support}^{t}$ can be converted into a vector representation, i.e., a

point in high-dimensional space. Next, calculate the similarity of any two vectors in the feature space by the metric module. Here, we have the cosine distance for measurement, which is a parameter-free distance metric. The similarity output forms the final prediction probability of the query sample. The overall similarity of a query sample x_q to the example set $\hat{x}_k$ of category k:

$$\sin\left(x_q, \hat{x}_k\right) = \frac{1}{N}\sum_{n=1}^{N}\cos\left(x_q, x_i^k\right) \tag{5}$$

where x_i^k is the n-th sample in $\hat{x}_k$ and cos represents the cosine function. Then, inscribe its prediction probability on all categories:

$$pred\left(x_q\right) = \frac{1}{\sum_{j=1}^{K} e^{\sin\left(x_q, \hat{x}_j\right)}}\left[e^{\sin\left(x_q, \hat{x}_1\right)}, e^{\sin\left(x_q, \hat{x}_2\right)}, \ldots, e^{\sin\left(x_q, \hat{x}_K\right)}\right] \tag{6}$$

The one with the highest score is the prediction category for x_q.

2.3.5. The Dynamic Activation Mapping Strategy

To address the issue of low recognition accuracy for small and concealed targets in WT surface defect detection, a dynamic activation mapping strategy is introduced in the embedding model at the single task level. Figure 7 describes how it implements:

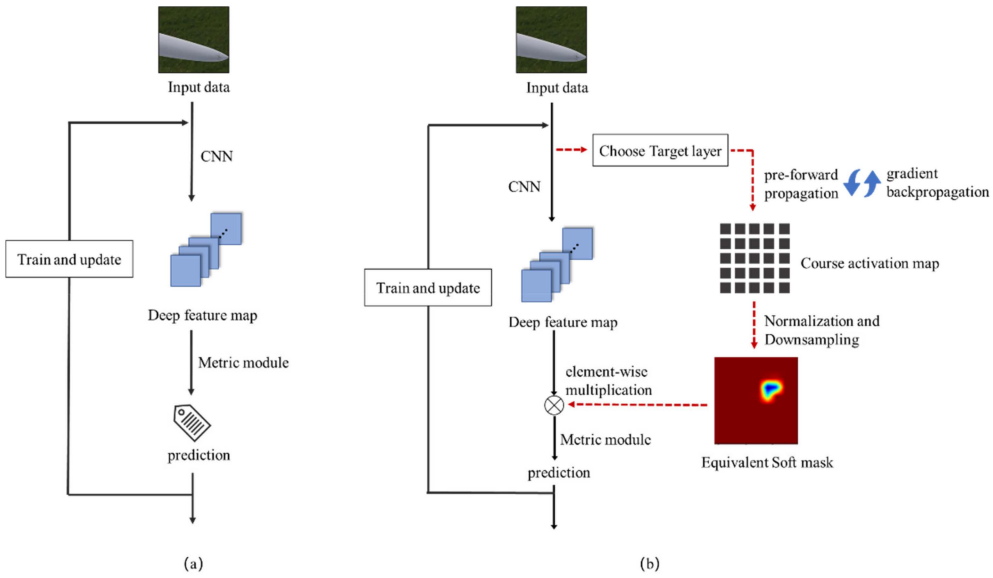


Figure 7. The process comparison between the original model and the model with dynamic cam. (**a**) original model (**b**) model with dynamic cam.

Assume that for any image sample z with category label c, a depth feature map fea is obtained after feeding it into a CNN-based embedding model. In the beginning, a target layer $Layer_{target}$ needs to be selected. In the context of this paper, we believe that the target layer should neither be too shallow nor too deep: if it is too shallow, excessive noise will interfere with the accurate representation of WT defective features, while if it is too deep, the features of small-scale defects will be drowned in the surrounding pixels due to the high level of feature map extraction. Collectively, target layer selection depends on the average percentage of defects in the original image. After performing pre-forward propagation and gradient back-propagation, record the target layer's activation value as well as the target layer's gradient information by backpropagation of the score on category c.

The channel weight vector on neurons is obtained,

$$\beta_z^c = \frac{1}{w_{tar} * h_{tar}}\sum_i\sum_j -\frac{\partial score_c}{\partial T_{ij}^z} \tag{7}$$

where w_{tar} and h_{tar} represent the length and width of $Layer_{target}$, $score_c$ represents sample score on category c, and T_{ij}^k represents pixel value in the i-th row and j-th column in $Layer_{target}$.

In accordance with the activation value and channel weight vector, we implement linear feature fusion on the channel dimension for $Layer_{target}$. Especially,

$$L_{map}^z = \sum_l \beta_z^c(l) Activation_l^z \tag{8}$$

where $ctivation_l^z$, $\beta_z^c(l)$ represent the activation map and contribution on the l-th channel, respectively.

Only pixel points that have a positive influence on the category are taken into account, so the Relu function is applied to the heat map. Generate a rough activation heat map, which means the weight contribution distribution in the spatial dimension on $Layer_{target}$.

$$Cam_{coarse} = Relu\left(L_{map}^z\right) \tag{9}$$

Based on the min-max normalization criterion, a normalization operation is applied. The pixel value in the i-th row and j-th column of the new heat map Cam'_{coarse} is

$$C_{ij}^{k\prime} = \frac{C_{ij}^k - \min(Cam_{coarse})}{\max(Cam_{coarse}) - \min(Cam_{coarse}) + eps} \tag{10}$$

where C_{ij}^k represents pixel value in the i-th row and j-th column in Cam_{coarse}. $\max(Cam_{coarse})$ and $\min(Cam_{coarse})$ are the maximum and minimum values of all elements in Cam_{coarse}, respectively. eps is a pretty small number to prevent overflow, which is taken as 10×10^{-6}.

Then, adapt Cam'_{coarse} to the same size as the depth feature map fea to get the equivalent soft mask map:

$$Cov = avgpooling\{Cam'_{coarse} | r = \frac{size_{cam}}{size_{fea}}\} \tag{11}$$

where $size_{cam}$ and $size_{fea}$ are the length (or width) of Cam'_{coarse} and fea, r is the downsampling rate, and $avgpooling$ means the average pooling operation.

Finally, overlap the equivalent soft mask map on fea to achieve the enhancement of valid information and the filtering of redundant information on the basis of defect correlation. The enhanced feature map

$$fea\prime = Cov \otimes fea \tag{12}$$

where $\otimes$ refers to the element multiplication operation.

We replace fea with $fea\prime$ as input to the subsequent classification module for defect recognition. The equivalent soft mask map can be continuously and dynamically updated according to the current parameters, thus assisting the model training.

As for the reason why this strategy can alleviate the accuracy degradation caused by small and concealed defects, we take the following two underlying factors into consideration: on the one hand, a dynamic activation mapping strategy, as a means of information filtering, not only alleviates the pressure of high convolutional computation but also enhances task-related information on the depth feature map based on defect correlation metrics, which increases the training efficiency and the exploitation of the model's potential on the target task. On the other hand, the pooling operation performs forced-down sampling in order to make the image size smaller. In the base model, scholars are generally accustomed to choosing the last layer of features as the input to the classification module. When the defect accounts for a small proportion of the original image, important defect information is likely to be lost in pooling. The dynamic activation mapping pays attention to the intermediate layer information so that the lost information can be reused, thus helping to avoid the missed detection of small-scale targets.

2.3.6. Optimization Goal

In contrast to the base-level learning in ordinary supervised training, we proposed meta-level learning, which is dedicated to uncovering the commonalities that exist between multiple similar tasks.

Different tasks have their own adapted optimal function, while our strategy is done over the whole function space to get the common properties that all these functions follow. The objective function in this paper considers all training tasks $t \in \mathcal{T}_{train}$ and minimizes the sum of their loss functions on their respective test sets D^t_{query}, i.e.,

$$\min \sum_{t \in \mathcal{T}_{train}} \sum_{(x,y) \in D^t_{query}} L\left(y, f\left(x, D^t_{support}; \Theta\right)\right) \tag{13}$$

where Θ represents the meta parameter and $f\left(x, D^t_{support}; \Theta\right)$ represent the predicted value for sample x.

In our approach, the meta parameter Θ is not expressed in an explicit parametric way. Instead, the deep metric learning in 2.3.4 ensures that Θ is implicitly incorporated into the model parameters.

3. Experiments and Discussion

3.1. Implementation Details

The experimental environment in the article is NVIDIA Tesla V100 32GB and the Linux operating system. With PyTorch, the code is implemented in python3.8, CUDA11.3. $n_{tasks} = 1000, n_{train\_s} = 10, n_{train\_q} = 30, n_{val\_s} = 10,\ n_{val\_q} = 10,\ n_{val\_q} = 80, epochs = 100,$ and $batchsize = 128$. The uniform input size of the image is $224 \times 224 \times 3$, and we used an Adam optimizer with a 1e-3 initial learning rate for optimization training.

3.2. The Wind Turbine Inspection Dataset

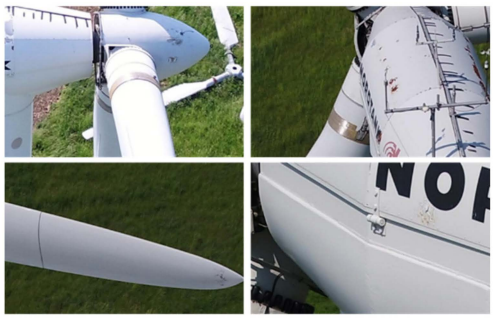
The data set used in this experiment comes from the UAV aerial wind turbine dataset provided by the Power Grid Company. To satisfy the few-shot condition, we have 20 samples for training and 2710 samples for test, i.e., $n_{wind\_s} = 20,\ n_{wind\_q} = 2710$. The data distribution is shown in Table 1. Figure 8 contains examples of normal and defective aerial images.

Table 1. Data distribution.

	Training Set	Testing Set
Normal	10	905
Defective	10	1805
Sum Up	20	2710

(a)

Figure 8. *Cont.*



(b)

Figure 8. Examples of normal and defective aerial images. (**a**) normal samples (**b**) defective samples.

3.3. Defect Visualization

Figure 9 shows some examples of defect visualization. It illustrates that our DMnet is capable of providing accurate identification for cracks, coating breakage, and corrosion in samples. Furthermore, even small target defects in a more difficult situation can be pinpointed.

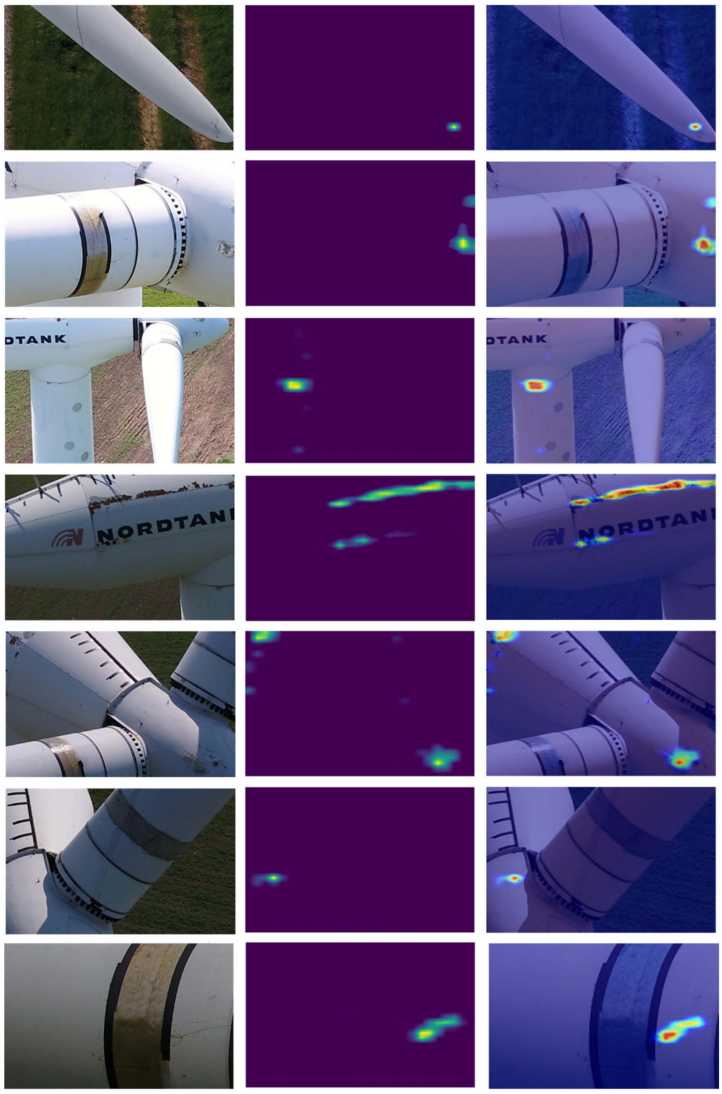


Figure 9. Examples of defect visualization.

We have selected a sample to show the dynamic recognition process under the proposed activation mapping strategy, as illustrated in Figure 10. We can see that in the early stage, the model is not yet good at targeting defects for recognition, so features in abruptly discordant locations are more likely to be noticed, such as edge lines and spindle connection of the blade. In the intermediate stage, some of the noise has been filtered out and edge lines are no longer mistaken for "defects". The spindle connection, however, is more difficult to distinguish since it has a more similar form to defects. In the later stage, the defective parts have already been well focused. Overall, as training epochs increase, the strategy directs the model's attention toward the parts that are more highly associated with the defect in the picture, ultimately helping the model to accurately identify and locate the defect.

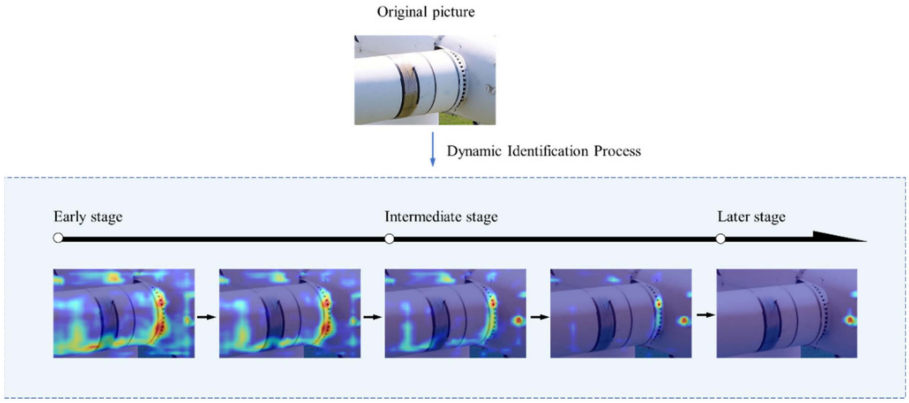


Figure 10. The dynamic recognition process.

3.4. Comparison with State-of-the-Art Methods

For a more objective evaluation, we use several indicators here, including accuracy (Acc), precision (Pre), recall (R), and F1-score. Specifically,

$$Acc = \frac{TN + TP}{FP + TN + TP + FN} \tag{14}$$

$$Pre = \frac{TP}{FP + TP} \tag{15}$$

$$R = \frac{TP}{TP + FN} \tag{16}$$

$$F_1 = \frac{2 * Pre * R}{Pre + R} \tag{17}$$

TP is the number of positive samples that are correctly distinguished. TN is the number of negative samples that are correctly distinguished. FP and FN, respectively, represent the negative and positive samples that were misclassified.

To verify the effectiveness of our method, we set up three groups of comparison models:

1. Conventional machine learning algorithms with manual feature extraction: LBF for feature extraction and SVM for classification (one of the most widely used methods).
2. Classical image classification algorithms based on deep learning: VGG, Res2net.
3. Classical few-shot learning algorithms: MetaBaseline [35], RelationNet [36], Baseline-plus [37], and NegMargin [38].

Table 2 compares the model performance on the metrics. The recognition accuracy of the first group does not exceed 50%, which indicates that the manually extracted features are likely to be poorly adapted to WT surface defect detection. The second group showed an overall increase in performance, but the f1-score was essentially the same as the first group. Since models in the third group are designed specifically for the few-samples situation, they performed significantly better than the first two groups. Clearly, the proposed DMnet

achieves the best scores on every metric. Compared to the second place, DMnet reflects an improvement ranging from 3% to 7% on each metric.

Table 2. Comparative experiments.

	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
LBF + SVM	49.89	62.41	62.27	62.34
VGG-16	63.32	65.02	66.74	62.80
Res2net-50	62.25	60.84	61.48	62.87
MetaBaseline	69.82	67.58	69.24	67.85
RelationNet	69.96	66.34	66.52	66.42
Baseline-plus	73.76	71.73	73.86	72.13
NegMargin	73.25	71.09	73.06	71.49
DMnet	**80.41**	**78.80**	**75.70**	**76.83**

Two additional notes on the data for Table 2:

1. Training vgg16 and Res2net from scratch with only 20 samples would be difficult to converge due to the unreliable empirical risk minimizer described in 2.1; thus, the results here are based on ordinary supervised learning in the context of pre-training weights with ImageNet [39].
2. To keep the irrelevant variables consistent in the comparison, the embedding models for both DMnet and the four classical few-shot learning algorithms in Table 2 are VGG.

3.5. Ablation Experiments

To further validate the respective validity of the innovation points in this paper and disentangle the contributions made by each component, we set up two sets of ablation experiments, as depicted in Table 3.

Table 3. The ablation experiments.

Embedding Model	Applied Strategy	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
VGG	baseline	63.32	65.02	66.74	62.80
	Baseline + approach 1	76.94	74.22	72.52	73.19
	Baseline + approach 2	73.91	70.65	68.24	69.02
	Baseline + approach 1 + approach 2	80.41	78.80	75.70	76.83
Res2net	baseline	62.25	60.84	61.48	62.87
	Baseline + approach 1	70.85	67.66	68.42	67.96
	Baseline + approach 2	66.46	67.21	69.31	65.74
	Baseline + approach 1 + approach 2	74.10	70.78	69.34	69.90

Note: Baseline: general supervised learning model (i.e., the second set of comparison models in Table 2). Approach 1**:** the three-stage construction introduced in Sections 2.3.2–2.3.4 Approach 2: the dynamic activation mapping strategy introduced in Section 2.3.5.

From the first four rows, the three-stage meta-learning framework provides a significant boost to the experimental results, enabling the model to achieve an increase of almost 14% and 11% in accuracy and F1-score, respectively. The dynamic activation mapping strategy also contributes further gains to the final results, with each metric improving by 3.5–4.5%. This demonstrates that every component of our method is practical and effective.

The last four rows were experimented under the condition that the embedding model was changed to res2net, and the results illustrate a similar pattern to the first four rows. With different embedding models, DMnet is always able to make great progress compared to baseline, which shows good generalizability and scalability.

4. Conclusions

In this paper, we propose a new few-shot WT surface defect detection framework—DMnet. To address the few-shot issue, a cross-task training strategy is designed. Searching for a set of generalized defect identification criterion, the hypothesis space shrinks to

achieve rapid learning and fast adaptation to new tasks in a few-shot scenario. Moreover, a metric-learning-based classification module is developed to learn a general similarity measure that is transferable between tasks. Its non-parametric structure allows the network to adapt more quickly to matching (trained and tested) samples under the same task. Additionally, we introduce a new activation mapping strategy with a dynamic feedback session for training to improve the recognition accuracy for small targets.

To ensure a few-shot scenario, we specified to conduct experiments with a training set of only 20 wind turbine defect samples. Under the same experimental conditions, our method has the best defect recognition ability compared with conventional machine learning algorithms, classical deep-learning-based image classification algorithms, and classical few-shot learning algorithms. In particular, compared to the most commonly used deep learning methods, ours reflects improvements ranging from 9% to 14% in accuracy, precision, recall, and F1-score metrics. The ablation results illustrate that by chunking the model, it is confirmed that both the three-stage construction and the dynamic activation mapping strategy are valid components and each contributes to the gain. Further, the results present consistent patterns after replacing the base model, which indicates that DMnet can be adapted to different deep learning models, which is highly scalable.

In general, DMnet alleviates the information scarcity caused by insufficient defect samples by constructing a large number of similar tasks to obtain prior knowledge, which is applied to solve few-shot defect detection problems. Additionally, it provides the first dynamic interpretability within the network during training by formulating a feedback mechanism, which greatly improves the sensitivity of recognizing defects in small and concealed targets, and finally achieves high-precision universal defect detection, which is of great significance to the research of equipment diagnostics and condition monitoring.

Due to the data limitation, this paper only discusses whether the wind turbine is defective or not. In the next step, we consider subdividing the defect categories by collecting more samples of different defect types. Additionally, for the more difficult new tasks, cosine distance may not be effective enough to reflect the similarity between samples owing to its limitations as a predefined metric. The similarity measure that is dynamically updatable (e.g., a learnable network) may be a good choice for greater expressiveness.

Author Contributions: Conceptualization, J.Y.; methodology, K.L. (Kaipei Liu), L.Q. and J.Y.; software, B.L. and J.Y.; validation, J.Y.; formal analysis, J.Y.; investigation, Q.L., F.Z. and Q.W.; resources, Q.L., F.Z. and Q.W.; data curation, H.L. and J.Y.; writing—original draft preparation, J.Y.; writing—review and editing, J.Y.; visualization, K.L. (Kexin Li) and J.W.; supervision, Q.L., F.Z. and Q.W.; project administration, L.Q.; funding acquisition, L.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the science and technology project of State Grid Information and Telecommunication Group Co., Ltd. (SGTYHT/19-JS-218).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dutton, A.; Backwell, B.; Fiestas, R.; Joyce, L.; Qiao, L.; Zhao, F. Balachandran NGlobal Wind Report 2019. 2020. Available online: https://gwec.net/global-win-report-2019/ (accessed on 12 June 2022).
2. Shi, Y. Phased array ultrasonic detection of glass fiber composites for Wind Turbine Blades. *Nondestruct. Test.* **2018**, *40*, 56–58. [CrossRef]
3. Yang, Q. How to detect wind Turbine blade defects. *Sci. Technol. Wind.* **2019**, 1.
4. Tiwari, K.A.; Raisutis, R.; Samaitis, A. Hybrid signal processing technique to improve the defect estimation in ultrasonic non-destructive testing of composite structures. *Sensors* **2017**, *17*, 2858–2879. [CrossRef] [PubMed]

5. Tarfaoui, M.; Khadimallah, H.; Shah, O.; Pradillon, J.Y. Effect of spars cross-section design on dynamic behavior of composite wind turbine blade: Modal analysis. In Proceedings of the International Conference on Power Engineering, Istanbul, Turkey, 13–17 May 2013; pp. 1006–1011.
6. Bo, Z.; Yanan, Z.; Changzheng, C. Acoustic emission detection of fatigue cracks in wind turbine blades based on blind deconvolution separation. *Fatigue Fract. Eng. Mater. Struct.* **2017**, *40*, 959–970. [CrossRef]
7. Hwang, S.; An, Y.-K.; Sohn, H. Continuous-wave line laser thermography for monitoring of rotating wind turbine blades. *Struct. Health Monit.* **2019**, *18*, 1010–1021. [CrossRef]
8. Lienhart, R.; Maydt, J. An Extended Set of Haar-like Features for Rapid Object Detection. In Proceedings of the Image Processing International Conference, Rochester, NY, USA, 22–25 September 2002.
9. Ojala, T.; Pietikainen, M.; Maenpaa, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *247*, 971–987. [CrossRef]
10. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
11. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
12. Hearst, M.A.; Dumais, S.T.; Osman, E.; Platt, J.; Schölkopf, B. Support vector machines. *IEEE Intell. Syst.* **1998**, *13*, 18–28. [CrossRef]
13. Viola, P.A.; Jones, M.J. Rapid Object Detection using a Boosted Cascade of Simple Features. In Proceedings of the IEEE Computer Society Conference on Computer Vision Pattern Recognition, Kauai, HI, USA, 8–14 December 2001.
14. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
15. Peng, L.; Liu, J. Detection and analysis of large-scale WT blade surface cracks based on UAV-taken images. *IET Image Process.* **2018**, *12*, 2059–2064. [CrossRef]
16. Chen, J.; Shen, Z. Study on visual detection method for wind turbine blade failure. *Int. Conf. Energy Eng. Environ. Prot.* **2018**, *121*, 042031. [CrossRef]
17. Wang, L.; Zhang, Z. Automatic Detection of Wind Turbine Blade Surface Cracks Based on UAV-taken Images. *IEEE Trans. Ind. Electron.* **2017**, *64*, 7293–7303. [CrossRef]
18. Ioffe, S.; Szegedy, C. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015*; PMLR: Online, 2015.
19. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
21. Gao, S.-H.; Cheng, M.-M.; Zhao, K.; Zhang, X.-Y.; Yang, M.-H.; Torr, P. Res2net: A new multi-scale backbone architecture. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 652–662. [CrossRef]
22. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019.
23. Xu, D.; Wen, C.; Liu, J. Wind turbine blade surface inspection based on deep learning and UAV-taken images. *J. Renew. Sustain. Energy* **2019**, *11*, 053305. [CrossRef]
24. Yang, P.; Dong, C.; Zhao, X.; Chen, X. The Surface Damage Identifications of Wind Turbine Blades Based on ResNet50 Algorithm. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020; pp. 6340–6344.
25. Qiu, Z.; Wang, S.; Zeng, Z.; Yu, D. Automatic visual defects inspection of wind turbine blades via YOLO-based small object detection approach. *J. Electron. Imaging* **2019**, *28*, 043023. [CrossRef]
26. Shihavuddin, A.S.M.; Chen, X.; Fedorov, V.; Nymark Christensen, A.; Andre Brogaard Riis, N.; Branner, K.; Bjorholm Dahl, A.; Reinhold Paulsen, R. Wind Turbine Surface Damage Detection by Deep Learning Aided Drone Inspection Analysis. *Energies* **2019**, *12*, 676. [CrossRef]
27. Bottou, L.; Bousquet, O. The tradeoffs of large-scale learning. In Proceedings of the 21st Annual Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 3–6 December 2007; pp. 161–168.
28. Bottou, L.; Curtis, F.E.; Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Rev.* **2018**, *60*, 223–311. [CrossRef]
29. Wang, Y.; Yao, Q.; Kwok, J.T.; Ni, L.M. Generalizing from a Few Examples: A Survey on Few-shot Learning. *ACM Comput. Surv.* **2020**, *53*, 1–34. [CrossRef]
30. Vanschoren, J. Meta-learning: A survey. *arXiv* **2018**, arXiv:1810.03548.
31. Schmidhuber, J. Evolutionary Principles in Self-Referential Learning. On Learning How to Learn: The Meta-Meta-Hook. Ph.D. Thesis, Institut f. Informatik, Technische Universität München, Munich, Germany, 1987. Volume 1.
32. Lu, J.; Gong, P.; Ye, J.; Zhang, C. Learning from very few samples: A survey. *arXiv* **2020**, arXiv:2009.02653.
33. Bergmann, P.; Fauser, M.; Sattlegger, D.; Steger, C. MVTec AD—A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; IEEE: Piscataway, NJ, USA, 2020.
34. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional networks. In Proceedings of the Conference Neural Information Processing Systems (NIPS), Lake Tahoe, NV, USA, 3–6 December 2012; p. 1097.

35. Chen, Y.; Liu, Z.; Xu, H.; Darrell, T.; Wang, X. Meta-baseline: Exploring simple meta-learning for few-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 9062–9071.
36. Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P.H.S.; Hospedales, T.M. Learning to Compare: Relation Network for Few-Shot Learning. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
37. Chen, W.-Y.; Liu, Y.-C.; Kira, Z.; Wang, Y.-C.F.; Huang, J.-B. A closer look at few-shot classification. *arXiv* **2019**, arXiv:1904.04232.
38. Liu, B.; Cao, Y.; Lin, Y.; Li, Q.; Zhang, Z.; Long, M.; Hu, H. Negative Margin Matters: Understanding Margin in Few-shot Classification. *Eur. Conf. Comput. Vis.* **2020**, *12349*, 438–455.
39. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: Piscataway, NJ, USA, 2009.