A Novel Deep Class-Imbalanced Semisupervised Model for Wind Turbine Blade Icing Detection

Cheng, Xu; Shi, Fan; Liu, Xiufeng; Zhao, Meng; Chen, Shengyong

Published in:
IEEE Transactions on Neural Networks and Learning Systems

Link to article, DOI:
10.1109/TNNLS.2021.3102514

Publication date:
2022

Document Version
Peer reviewed version

A Novel Deep Class-Imbalanced Semisupervised Model for Wind Turbine Blade Icing Detection

Xu Cheng, *Member, IEEE*, Fan Shi, *Member, IEEE*, Xiufeng Liu, Meng Zhao, *Member, IEEE*, and Shengyong Chen, *Senior Member, IEEE*

***Abstract*— Wind energy is of great importance for future energy development. In order to fully exploit wind energy, wind farms are often located at high latitudes, a practice that is accompanied by a high risk of icing. Traditional blade icing detection methods are usually based on manual inspection or external sensors/tools, but these techniques are limited by human expertise and additional costs. Model-based methods are highly dependent on prior domain knowledge and prone to misinterpretation. Data-driven approaches can offer promising solutions but require a massive amount of labeled training data, which are not generally available. In addition, the data collected for icing detection tend to be imbalanced because, most of the time, wind turbines operate under normal conditions. To address these challenges, this article presents a novel deep class-imbalanced semisupervised (DCISS) model for estimating blade icing conditions. DCISS integrates class-imbalanced and semisupervised learning (SSL) using a prototypical network that can rebalance features and measure the similarities between labeled and unlabeled samples. In addition, a channel calibration attention module is proposed to improve the ability to extract features from raw data. The proposed model has been evaluated using the blade icing datasets of three wind turbines. Compared to the classical anomaly detection and state-of-the-art SSL algorithms, DCISS shows significant advantages in terms of accuracy. Compared to five different class-imbalanced loss functions, the proposed DCISS is competitive. The generalization and practicability of the proposed model are further verified in the use case of online estimation.**

***Index Terms*— Anomaly detection, imbalanced learning, semisupervised learning (SSL), time-series classification (TSC), wind turbine.**

Manuscript received November 29, 2020; revised April 9, 2021; accepted July 31, 2021. This work was in part supported by the National Natural Science Foundation of China under Grant 62020106004, Grant 92048301, Grant 61906133, and Grant 61703304; and in part by the Opening Foundation of Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology, China, under Grant TJUT-KLICNST-K20180002. *(Corresponding authors: Fan Shi; Shengyong Chen.)*

Xu Cheng is with the Engineering Research Center of Learning-Based Intelligent System, Ministry of Education, Tianjin 300384, China, also with the Key Laboratory of Computer Vision and System of Ministry of Education, School of Computer Science and Technology, Tianjin University of Technology, Tianjin 300384, China, and also with the Department of Manufacturing and Civil Engineering, Norwegian University of Science and Technology (NTNU), 2815 Gjøvik, Norway (e-mail: xu.cheng@ieee.org).

Fan Shi, Meng Zhao, and Shengyong Chen are with the Engineering Research Center of Learning-Based Intelligent System, Ministry of Education, Tianjin 300384, China, and also with the Key Laboratory of Computer Vision and System of Ministry of Education, School of Computer Science and Technology, Tianjin University of Technology, Tianjin 300384, China (e-mail: shifan@email.tjut.edu.cn; zh_m@tju.edu.cn; sy@ieee.org).

Xiufeng Liu is with the Department of Technology, Management and Economics, Technical University of Denmark, 2800 Kongens Lyngby, Denmark (e-mail: xiuli@dtu.dk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3102514.

Digital Object Identifier 10.1109/TNNLS.2021.3102514

I. Introduction

WIND energy is of great importance in dealing with global climate change and in promoting green energy transitions. In recent years, the capacity of installed wind turbines has increased dramatically. In order to make full use of wind energy, wind farms are usually built in regions at high altitudes and latitudes [1]. When wind turbines are installed in these regions, blade icing often occurs, which has a significant impact on aerodynamic performance, site accessibility (due to ice throwing), annual energy production, and turbine lifetime [2]. Ice accretion on blades not only affects power generation with up to a 30% loss in annual power generation in severe cases but also gives rise to safety problems in the vicinity of wind power plants [3]. These operational and safety issues have prompted research on ways of identifying ice accretion and discovering the methods for ice removal at an early stage [4]. Therefore, the identification of blade icing on wind turbines is of vital importance to the proper maintenance of wind farms.

At present, human observation is one of the most common methods in wind farms for blade icing identification. This approach estimates icing conditions by analyzing the power curve of the wind turbine, in combination with external factors, such as environmental sensor readings and blade speeds. The advantage of human observation is that the results are fairly consistent and do not rely solely on external sensors. However, one drawback is that it is inherently subjective; different observers may draw different conclusions. This can lead to the serious consequences of not detecting blade icing after a certain level has been reached. To address the drawbacks of human observation, other techniques of antiblade icing/deblade icing methods for wind turbines have been investigated, which includes the following three types: passive, active, and hybrid. The passive method is to prevent ice from forming on the blade surface by applying special materials [5]. Such a method has the advantage of reducing operating costs and making blade surface ice-free without the need for external control systems [6]. The active method is to avoid blade icing by employing thermal or mechanical techniques. The advantage is that such applications are controllable, but a disadvantage is that it requires additional mechanical maintenance and replacement work. The hybrid method combines the advantages of both active and passive methods and has received extensive attention in recent years [2], [7]. All of these techniques, however, are conventional methods that can result in high costs and require additional effort. More importantly,

these methods are not sufficiently accurate, as they are based on the estimation of blade icing conditions without taking the turbine's internal unreliability into account. Furthermore, these methods may harm the mechanical structure of the turbine when replacement or new installation takes place [8].

There are also extensive studies to identify the blade icing condition of wind turbines based on the physical processing of icing [9], [10]. Icing models can be divided into empirical statistical models (ESMs) and icing growth models (IGMs). The ESM requires long-term monitoring of the environmental parameters in a particular area and analyses of a large amount of historical data to predict the frequency and amount of ice accretion in that area; however, the ESM is not very accurate. The IGM takes blade icing as a complex physical process that combines aerodynamics, the hydrodynamics of gas–liquid two phases, and heat transfer. The IGM is useful for understanding the icing process but requires a great deal of domain knowledge [11]. In addition, the IGM relies heavily on specific assumptions and requires some external experimental tools, such as wind tunnels.

With the wide use of sensor technologies in wind turbines, a large amount of data have been collected, including information on healthy conditions and operating data. Increasing research attention has been focused on data-driven approaches that detect blade icing by mining the hidden information in historical data [12]. Compared to other approaches, data-driven approaches do not require prior domain knowledge, special materials, or mechanical tools. Data-based methods for blade icing detection can be further divided into machine-learning- and deep-learning-based methods. Machine-learning-based methods identify blade icing by extracting the representative features of characterizing icing conditions and then creating classification models from these extracted features. The most commonly used models include the shallow machine learning models, such as logistic regression, support vector machines, and random forest [13], [14]. However, the feature extraction process is time-consuming and requires some domain knowledge and expertise. Furthermore, feature extraction is typically separate from blade icing detection, which requires a separate design and execution process. This can affect overall performance. In recent years, deep learning has emerged as a powerful tool and has been successfully applied in many fields, including blade icing identification for wind turbines [8], [15]. Deep-learning-based methods provide a promising and effective solution for blade icing detection. Deep-learning-based methods attempt to offer high-level representations of sensor data and identify icing conditions via a hierarchical structure [16]. Nevertheless, there are still some challenges. First, deep-learning model training requires a large amount of labeled data. The labeling of sensor data from wind turbines involves significant human effort, which is often impossible. Therefore, how to make use of unlabeled data while building deep-learning models is an ongoing challenge in blade icing identification. Second, there is a heavy data imbalance between the ice-free (i.e., normal) and icing states, as, most of the time, wind turbines operate in a normal state. This can be further complicated by the fact that the imbalance problem is unknown in unlabeled data. Therefore, it is also challenging to deal with the imbalance between the labeled and unlabeled data. Third, wind turbines often operate under various environmental and climate conditions, and therefore, the measured data are typically highly nonlinear and nonstationary. Thus, it is not trivial to extract useful features from sensor data when creating icing detection models.

To address the above challenges, this article proposes a novel deep class-imbalanced semisupervised (DCISS) model for blade icing detection in wind turbines. To make use of unlabeled data, we apply a semisupervised learning (SSL) framework. However, SSL has the limitation that classes of training data should be balanced; otherwise, the performance of the models created from these data may be compromised. Therefore, we propose a prototypical network-based method for rebalancing the classes. In this method, we first construct the prototypes for all classes from labeled data. Then, the class imbalance is rebalanced with the help of prototypes [17]. At the same time, information from unlabeled data can be utilized to update these balanced prototypes of labeled data. This is achieved by computing the similarity of the prototype of the unlabeled data (i.e., unknown classes) to all the prototypes of the labeled data (i.e., known classes). The similarity is calculated based on the model training. In addition, a channel attention module is proposed to enhance the ability to extract features from the training data. In summary, the following contributions are made in this article.

1) We propose a novel end-to-end deep learning framework that integrates SSL and class-imbalanced learning methods. The proposed framework can not only overcome the problem of class imbalance but also make full use of available data, both labeled and unlabeled, for training by using a prototypical learning approach. In addition, we introduce a channel calibration attention module to a convolutional neural network (CNN) to improve its feature extraction capabilities. The proposed SSL method does not require prior knowledge of data labeling, which provides great potential for use in other machine learning applications.
2) We comprehensively evaluate the proposed framework using real-world supervisory control and data acquisition (SCADA) data. We compared it with state-of-the-art SSL methods and imbalanced learning methods in blade icing detection. The results indicate the superiority of the proposed framework. In addition, we perform online testing, which also demonstrates the effectiveness, generalizability, and feasibility of the proposed framework in real cases of online blade icing detection.

The rest of this article is structured as follows. Section II reviews related work on this topic. Section III presents the proposed model. Section IV describes the model evaluation experiments. Section V discusses the results. Section VI concludes this study.

II. Related Work

A. Wind Turbine Blade Icing Detection

Blade icing detection for wind turbines has been studied for years. The used methods can broadly be classified into

the following three categories: traditional physical, mathematical modeling, and data-driven methods. Traditional physical methods improve anti-icing or deicing capabilities using special materials (i.e., passive methods), external sensors, and equipment (i.e., active methods) or hybrids of the two. For example, liquid-infused surfaces and porous superhydrophobic polyvinylidene fluoride coatings have been employed to prevent icing [18], [19]. External sensors or equipment, such as electrothermal and ultrasonic deicing devices, have also been used widely in wind farms [20], [21]. In contrast, mathematical methods model (or simulate) the physical processes of blade icing, allowing for the ice accretion to be inferred. The drawback of mathematical modeling is that it depends heavily on specific assumptions and constraints that require good domain knowledge [22], and the models become complex. In recent years, a large amount of data have been collected due to the wide use of sensor technologies in wind farms. At the same time, machine/deep learning becomes increasingly popular due to its proven success in many areas. Data-driven methods can provide state-of-the-art performance because of their excellent automated feature learning capabilities, which are crucial to the identification of blade icing. Recently, Jiménez *et al.* [13] proposed a machine learning model for detecting the presence and thickness of the ice. Liu *et al.* [15] presented an ensemble deep learning model based on multilevel features extracted from SCADA data. Yuan *et al.* [8] proposed a deep learning model that combined wavelets and a CNN. However, most of these data-driven methods detect blade icing based on well-labeled data. In reality, obtaining labeled data is a well-known challenge since labeling data is an expensive and time-consuming process; thus, most wind farm data collected have no labels. To address this issue, this article proposes an SSL framework to close this gap.

B. Anomaly Detection

With the increasing demand and application, such as risk management, financial supervision, and health and medical risk assessment, anomaly detection is playing an increasingly important role in data mining, machine learning, computer vision, and statistics [23]. Anomaly detection (also known as outlier detection) aims to identify rare values, items, events, or observations that differ significantly from the majority of the data. Anomaly detection methods can be classified into three categories depending on how samples should be labeled: fully supervised, semisupervised/weakly supervised, and unsupervised methods [23]. Fully supervised methods require that both normal and abnormal classes were well labeled and include the traditional binary classification method, such as one-class SVM. The drawback of fully supervised methods is that they require significantly extra work of labeling the data. In contrast, unsupervised anomaly detection methods do not require labeled data for training; instead, they build the detection model based on normal samples and then detect anomalies according to data statistics, such as the distribution of values. Therefore, they require much less human effort and have received a lot of research attention in the past decade. The well-known unsupervised methods include Isolation Forest [24], Local Outlier Factor (LOF) [25], and REPEN [26]. The third category is semisupervised/weakly supervised methods, where some of the samples have labels, but these may be incomplete or inaccurate. In practice, most of the data fall into this category, so we need to address it more vigorously. The most used semisupervised/weakly supervised methods include DevNet [27], Deep SAD [28], and PReNet [27]. In this article, we also have partially labeled wind-turbine icing data, where a semisupervised binary classification has been applied to detect anomalies. However, our method emphasizes the integration of SSL and class-imbalanced learning into a single framework, where a prototypical network is introduced so that our method can identify anomalies more effectively.

C. Time-Series Classification

Supervised time-series classification (TSC) methods can be classified into those that are distance-based, feature-based, and deep-learning-based. For example, the distance-based approach [29], which integrates discrete SVM and warping distance, has been effectively verified on various datasets. Feature-based approaches classify time-series data based on extracted patterns. The most popular feature-based approaches include the bag-of-features framework [30], Bag-of-SFA-Symbols (BOSS) [31], and the hidden unit logic model [32]. However, the main challenge for feature-based approaches is on how to obtain high-quality features as this often requires substantial human effort and domain knowledge. In recent years, deep learning methods have been investigated extensively to tackle feature extraction challenges. Several deep learning models have been presented. Zheng *et al.* [33] proposed a multichannel deep learning model. Karim *et al.* [34] developed a hybrid parallel structure with long short-term memory (LSTM) and a fully convolutional network (FCN) (LSTM-FCN). This structure has shown a superior level of performance over the others. Cheng *et al.* [35] [35] also presented a similar deep learning model, but they introduced a spectral branch to the LSTM-FCN. Moreover, they developed a novel model with a combination of dense connections and CNN to achieve state-of-the-art performance [12]. All of these works serve as inspiration for the design of the feature encoder described in this article.

Machine learning from well-labeled data is very successful, but there are various applications where massive amounts of labeled data are not available. As a result, SSL has gained increasing interest in recent years. Most existing approaches for SSL for TSC focus on univariate learning, such as in [36]–[38]. One of the best known semisupervised methods is dynamic time warping (DTW) for estimating labels for unlabeled time series [38]. Marussy and Buza [39] presented a semisupervised approach called SUCCESS, which consists of restricted hierarchical clustering and DTW. These researchers reported that this approach could significantly outperform the original DTW classifier. Begum *et al.* [40] introduced a stop criterion based on the minimum description length for SSL. Zhang *et al.* [41] developed an SSL framework based on CNN and prototype learning for TSC, which shows competitive performance over the others. Nevertheless, existing SSL

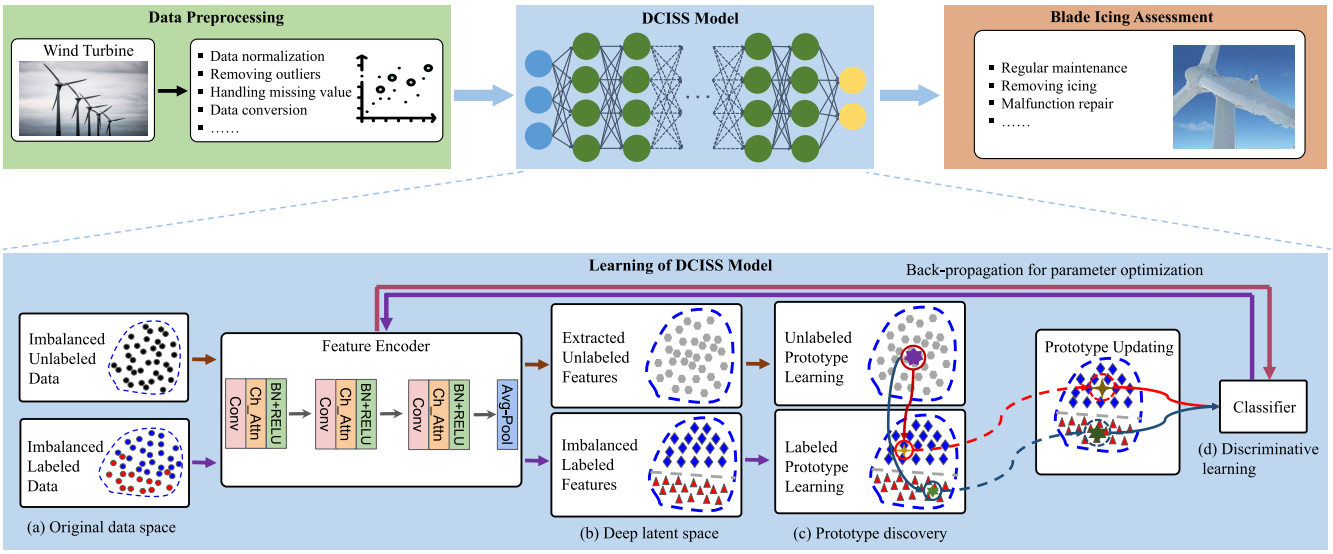

Fig. 1. Overview of the proposed DCISS model for blade icing detection in wind turbines. For the training of the DCISS model, the forward step of the DCISS training is suggested by the arrow from left to right. (a) Original data space for imbalanced labeled and unlabeled data (minority positive and majority negative). (b) Data distribution in a projected deep latent space by the feature encoder. (c) Prototypical discovery by the prototypical network. (d) Learning of a discriminatory classifier by comparing the Euclidean distances between features and prototypes (discriminative learning). Parameter optimization is performed in an inverse path (from right to left) by backpropagating the gradients of discriminative loss to update the parameters in both the feature encoder and the discriminative classifier.

approaches can be grouped according to the following two assumptions. The first is that the class types of the unlabeled data are the same as those of labeled data [42]. The second is that the data about classes are balanced; in other words, each class has almost the same number of samples [43], [44]. This assumption can have a negative impact on model performance if the classes are actually imbalanced in the unlabeled data. Yang and Xu [45] investigated class-imbalanced SSL both theoretically and empirically using pseudolabeling and self-training. In contrast to existing work, we address the challenge of unlabeled data in this article by creating a prototypical network that provides a uniform framework for class-imbalanced SSL rather than by using pseudolabeling or self-training.

III. Class-Imbalanced Semisupervised Model for Wind Turbine Blade Icing Detection

In this section, we first introduce an overview of the proposed imbalanced SSL framework for blade icing detection in wind turbines. Then, we present the details of each component of the framework, including data preprocessing, class-imbalanced SSL, feature encoder, and prototype learning.

A. Overview

As shown in Fig. 1, the overall framework consists of three successive steps: data preprocessing, DCISS, and assessment of blade icing detection. Data preprocessing aims to improve the data quality for modeling. The cleaned data are then used for class-imbalanced SSL. Finally, the trained model is used to assess the blade icing detection of wind turbines. Based on the results, regular maintenance or early prediction of blade icing can be performed accordingly.

To reduce the impact of outliers and noise, it is necessary to clean the raw time series in a data preprocessing step. Data preprocessing includes cleansing, correlation analysis, data splitting, and normalization. The correlation analysis is to investigate the relationship among the time series, with the goal of reducing redundant information. In this study, we use the Pearson correlation analysis. Finally, the cleaned time series is partitioned according to a fixed-size time window in order to train the deep learning model.

The goal of DCISS is to make full use of the available data (with and without labeling) to train the model and, at the same time, address the class imbalance problem. Since a wind turbine usually operates under normal conditions most of the time, the obtained data are highly imbalanced (i.e., the number of samples from a normal state is much greater than that of an icing state). It should be noted that the imbalance problem occurs in both the labeled and unlabeled data. More importantly, there is no guarantee that the imbalance degree between these two states will be the same. In this article, we propose a class-imbalanced SSL based on prototypes of each class for addressing the imbalance problem in SSL. With this method, the prototype for each class is learned from the labeled data. Then, a prototype is calculated for the unlabeled data. Finally, the classes of the labeled data are updated, using the prototype of the unlabeled data. This update is automatically conducted during the model training process. Therefore, the learning prototype can solve not only the class imbalance problem but also the information utilization of the unlabeled data. We combine both functionalities in a unified framework to provide improved simultaneous support for the SSL and class-imbalanced learning.

In the last step, the obtained model is used for the evaluation of blade icing detection of wind turbines. The results of the model include a probability value of icing conditions. A user can set a threshold value (a common choice is 0.5) to determine whether the blade is icing or not. Therefore, the online

evaluation can help the user identify blade conditions and take the appropriate actions, such as activating an anti-icing/deicing system or stopping a wind turbine.

B. DCISS

The SSL of the framework can be formalized as follows: given a training dataset with N_l labeled samples $S_l = \{x_1, x_2, \ldots, x_{N_l}\}$, corresponding labels $Y_l = \{y_1, y_2, \ldots, y_{N_l}\}$, and N_u unlabeled samples $S_u = \{x_{N_l+1}, x_{N_l+2}, \ldots, x_{N_l+N_u}\}$, where $N_u \gg N_l$, $x_i \in \mathbb{R}^D$ is a sample with D dimensions, $y_i \in \{1, 2, \ldots, n_c\}$ is the set of classes of the labeled data, and n_c is the number of classes. The goal of SSL is to learn the model $f([S_l; S_u]; \theta) \rightarrow Y_l$, parameterized by θ and learned from the training dataset. This learning problem is to minimize the generalization risk $R(f) = E_{([S_l;S_u],Y_l)}[L(f([S_l; S_u]; \theta), Y_l)]$, where L is the loss function.

Most traditional SSL algorithms assume that labeled and unlabeled samples have approximately the same distribution across different classes. That is, the number of samples in different classes will be almost identical. Since this is often not the case, the decision boundaries of traditional SSL algorithms may be broken, degrading the performance of the model. What makes matters worse is that the imbalance degree is different in labeled and unlabeled samples. There are two possible approaches to the class imbalance problem, including data distribution rebalancing and class-balanced-loss methods. The data distribution rebalancing method solves the class imbalance problem by oversampling the data of the minority or undersampling the data of the majority, [46], [47]. However, one drawback is that this algorithm is of high complexity, and the performance can be easily affected by data sizes; therefore, the computation is very time-consuming. In contrast, the class-balanced-loss method aims to reweight the losses to match a given distribution in order to improve the generalization of the minor classes. This method can easily be applied to any deep learning model without the need for any sampling scheme [48]–[50]. In addition, previous study [45] has also verified that the class-balanced-loss method is a better choice for SSL, especially for generating pseudolabels. However, when the imbalanced ratio of labeled to unlabeled samples is different, the class-balanced-loss method is not as competitive for generating pseudolabels. In this article, we propose the class rebalancing method when using a prototypical network since a prototypical network is inherently good at solving class imbalance problems [17]. This network first projects imbalanced features onto prototypes in the latent space, then learns the class for each prototype, and balances the classes.

DCISS integrates SSL and class-imbalanced learning in a single framework by introducing a prototypical network. The structure of the DCISS is presented in Fig. 1. Given a partially labeled dataset with imbalanced classes, we use a feature encoder Ψ, which is a CNN, to project the data into deep latent space. The reason for using CNN is that it has good performance and convergence speed. Furthermore, we have added a channel calibration attention module to improve the feature extraction capability of the used CNN. When the features are projected, the prototypical network calculates the prototype for each class (icing or nonicing) from the labeled data. According to Zhang *et al.* [41], the prototype of unlabeled samples is helpful for model training, Thus, we update the prototype of each class with the prototype from unlabeled samples (see the bottom figure in Fig. 1). However, it should be noted that we only combine the unlabeled samples for our training with the aim of optimizing the model (i.e., to obtain more training data). In the testing phase, each test sample is mapped to the deep latent space by the feature extraction function, $\Psi(\cdot)$, and then classified by the trained classifier.

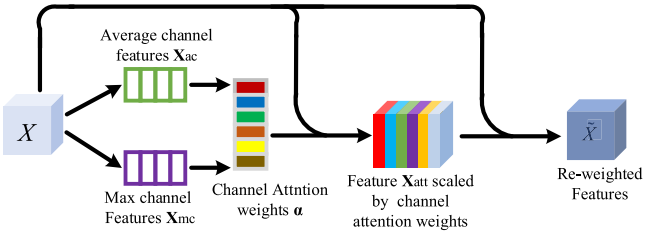

Fig. 2. Illustration of the channel calibration attention module.

C. Feature Encoder

The feature encoder is implemented as a CNN, which consists of three blocks shown in Fig. 1. Each block is composed of a convolution layer (Conv1D), channel calibration attention layer (Ch_Attn), normalized layer (Norm1D), and activation layer (ReLU). In each block, we introduce the Ch_Attn layer to improve the feature learning capability and speed of convergence. For an input of $X_{\text{raw}} \in \mathbb{R}^{d \times T}$, where d and T are the dimension and the window sizes of the samples, respectively, the outputs of the CNN block can be represented as follows:

$$\begin{aligned} X_c &= \text{Conv1D}(X_{\text{raw}}) \\ X_{\text{Attn}} &= \text{Ch\_Attn}(\text{X}_\text{c}) \\ X_{\text{AttnCNN}} &= \text{ReLU}(\text{Norm1D}(X_{\text{Attn}})) \end{aligned} \tag{1}$$

where X_c, X_{Attn}, and X_{AttnCNN} are the output of the Conv1D layer, the attention layer, and the ReLU layer, respectively. All three outputs have the same shape, $\mathbb{R}^{F \times T}$, where F is the number of filters in the Conv1D layer.

The process of Ch_Attn is illustrated in Fig. 2. The input for Ch_Attn is the output of the Conv1D layer with the shape of $\mathbb{R}^{F \times T}$. As mentioned above, F is the number of 1D CNN filters, each of which represents a feature channel. As a conventional CNN treats all feature channels equally, this will lead to the loss of some important information [12]. Therefore, we introduce the additional attention layer Ch_Attn to choose the most important feature channels.

Assume that the output of the Conv1D layer is the feature map, $X_c = [x_1, x_2, \ldots, x_F]$, where $x_k \in \mathbb{R}^{T \times 1}$ represents the kth channel. To select the informative channels, we apply global average and max pooling to obtain the distinctive features, X_{ac} and X_{mc}, respectively (see Fig. 2). The two features both have the shape of $\mathbb{R}^{F \times 1}$. They are then forwarded

to two multilayer perceptrons (MLPs) with shared weights (not shown in Fig. 2) to compute the attention weights α, defined as follows:

$$\begin{aligned} \alpha &= \sigma(\boldsymbol{W_2}(\boldsymbol{W_1}(X_{\text{ac}})) + \boldsymbol{W_2}(\boldsymbol{W_1}(X_{\text{mc}}))) \\ \boldsymbol{X_{att}} &= \alpha \otimes \boldsymbol{X} \end{aligned} \tag{2}$$

where $\boldsymbol{X}$ is the input of the feature encoder, α is the weights of the attention module, $\boldsymbol{X_{att}}$ is the weighted features, $\otimes$ represents elementwise multiple, and $\boldsymbol{W_1} \in \mathbb{R}^{F/r \times F}$ and $\boldsymbol{W_2} \in \mathbb{R}^{C \times F/r}$ represent the weights of the first and second MLP, respectively. Inspired by the success of residual blocks [51], we integrate the channel attention with the residual connection, called residual channel attention. That is, in Fig. 2, we have $\widetilde{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{X_{att}}$, where $\widetilde{\boldsymbol{X}}$ is the weighted features, $\boldsymbol{X}$ is the input of the feature encoder, and $\boldsymbol{X_{att}}$ is the feature scaled by the attention weight, α.

D. Prototype Learning

In prototypical networks, the prototype is a vector presenting each class. In this research, the prototype for labeled samples is the average of all features in the deep latent space, defined as follows:

$$C_l^i = \frac{1}{n_l^i} \sum_{k=1}^{n_l^i} X_l^{i,k} \tag{3}$$

where $C_l^i \in \mathbb{R}^{1 \times H}$ is the computed prototype for the class $i \in [1, \ldots, n_c]$ in the labeled samples (for simplicity, we use natural numbers to represent classes), n_c is the total number of classes, n_l^i is the number of labeled instances in the deep latent space for class i, and X_l denotes the features of the labeled samples in the deep latent space. For the unlabeled samples, the prototype can be calculated as follows:

$$C_u = \frac{1}{n_u} \sum_{k=1}^{n_u} X_u^k \tag{4}$$

where $C_u \in \mathbb{R}^{1 \times H}$ is the computed prototype for the unlabeled samples and n_u is the total number of unlabeled samples in the deep latent space. X_u represents the extracted unlabeled features in deep latent space.

In this article, we compute a single prototype for all unlabeled samples. Then, we update each class obtained from the labeled samples with the prototype of the unlabeled samples, which is defined as follows:

$$C_{\text{new}}^i = \frac{\sum_{k=1}^{n_l^i} X_l^{i,k} + \sum_{k=1}^{n_u} X_u^k}{2\left(n_l^i + n_u\right)}. \tag{5}$$

Since the calculated unlabeled prototype contains information for both classes (icing and nonicing), the obtained information can be used to increase the information entropy of the two classes obtained from the labeled data. Therefore, we update both labeled prototypes based on the unlabeled prototype according to the specific distance between the unlabeled and labeled prototypes, essentially. The proposed method for updating the prototypes is not only simple but also effective, as verified by the experiments described in Section IV.

After we obtain the prototypes for all classes, the distribution across the classes for a given time series can be calculated by measuring its distances to the prototypes in the deep latent space. Given a sample $x \in \mathbb{R}^d$, the projected features in the deep latent space are $\Psi(x)$. The probability of belonging to class k is measured by Softmax according to the distances to the prototypes in the deep latent space

$$p(y = k|x) = \frac{\exp\left(-\text{dist}\left(\Psi(x), C_{\text{new}}^k\right)\right)}{\sum_i \exp\left(-\text{dist}\left(\Psi(x), C_{\text{new}}^i\right)\right)} \tag{6}$$

where dist is the distance function for two given vectors. Here, we use the Euclidean distance because it has been shown to have good performance [52]. Then, we use the categorical cross entropy to guide the model training, which is defined as follows:

$$\mathcal{L} = -\sum_{(x_i, y_i)} \log(p(y = y_i|x_i, C_k)) \tag{7}$$

where (x_i, y_i) refers to the samples in the training data. C_k is the prototype corresponding to the sample of (x_i, y_i). The model parameters can be derived by minimizing the average loss, iterating the training epochs, and performing gradient descent updates for each iteration. In this article, we use the backpropagation algorithm to train the model in gradient descent and the Adam algorithm to optimize the loss function due to its low computing complexity and memory footprint. Algorithms 1 and 2 provide more details regarding the learning and testing processes, respectively.

Algorithm 1 Learning Process of DCISS

Input : Number of epochs: T
Output: Well-trained feature encoder: Ψ, Prototypes of classes: C_{new}
Data : Labeled samples: S_l and the corresponding labels: Y_l; Unlabeled samples: S_u

for $iter \leftarrow 1$ *to* T **do**
  $X_l \leftarrow \Psi(S_l)$; /* Extracted features in latent space */
  $X_u \leftarrow \Psi(S_u)$
  for $i \leftarrow 1$ *to* n_c **do**
    $idx \leftarrow$ Get the index of the class i in Y_l
    $C_l^i \leftarrow \frac{1}{n_l^i} \sum_{k=1}^{n_l^i} X_l^{i,k}(idx)$; /* Eq.(3) */
  end
  $C_u \leftarrow \frac{1}{n_u} \sum_{k=1}^{n_u} X_u^k$; /* Eq.(4) */
  for $i \leftarrow 1$ *to* n_c **do**
    $C_{new}^i \leftarrow \frac{\sum_{k=1}^{n_l^i} X_l^{i,k} + \sum_{k=1}^{n_u} X_u^k}{2(n_l^i + n_u)}$; /* Eq.(5) */
  end
  $p(y = k|x) \leftarrow \frac{exp(-dist(\Psi(x), C_{new}^k))}{\sum_i exp(-dist(\Psi(x), C_{new}^i))}$; /* Eq.(6) */
  $\mathcal{L} = -\sum_{(x_i, y_i)} log(p(y = y_i|x_i, C_{new}^k))$; /* Eq.(7) */
  $minimize \ \mathcal{L}$
end

Algorithm 2 Testing Process of DCISS

Input : Feature encoder: Ψ; Prototypes of classes: C_{new}
Output: Set of lablels: L
Data : Testing dataset: V
$L \leftarrow \{\}$
for ***each*** $v \in V$ **do**
  $X_v \leftarrow \Psi(v)$
  $p \leftarrow \frac{exp(-dist(X_v, C_{new}))}{\sum_i exp(-dist(X_v, C_{new}^i))}$
  $l \leftarrow argmax(p)$
  $L \leftarrow L \cup \{l\}$
end

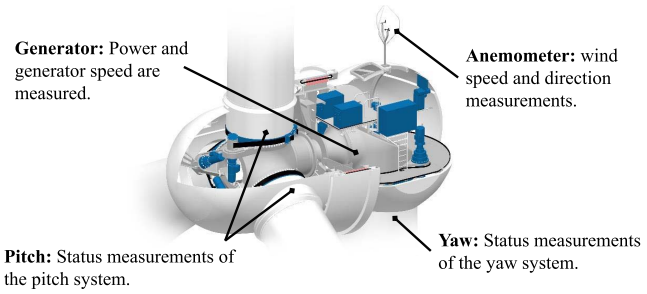

Fig. 3. Illustration of sensors in the wind turbine.

IV. Experiments

A. Experimental Setup

1) Settings: We implement the deep learning algorithm using PyTorch v1.6.0 and Python v3.8.0 and perform all experiments on a server equipped with NVIDIA Titan V GPUs. We do not search for the best hyperparameters in the hyperparameter space and, instead, use the same training parameters as those in the corresponding baselines.

2) Datasets: The data used in the experiments were provided by Goldwind Inc., one of the largest wind energy companies in the world. The data were recorded by the SCADA system in two wind turbines located in Inner Mongolia, China. One of the wind turbines logged data for 306 h and the other for 696 h. The resolution of the data is 7 s. In fact, over 100 different signals were collected by the sensors (some are shown in Fig. 3). Wind turbine experts labeled all the data and identified 26 variables related to blade icing, as shown in Table I. We mix the datasets from two wind turbines and use 70% for training (with nearly 10% labeled data and 60% unlabeled data) and 30% for testing. Note that the data used for training include labeled and unlabeled data, which are imbalanced, while the data for testing are balanced.

To investigate the impact of the data imbalance, we design three different test cases for our experiments, including $\rho_u = \rho_l$, $\rho_u = 2\rho_l$, and $\rho_u = 1$, where ρ is the imbalance ratio between the normal and icing classes (u and l represent the unlabeled and labeled data, respectively), defined as

$$\rho = \frac{N_{\text{normal}}}{N_{\text{icing}}} \tag{8}$$

where N_{normal} and N_{icing} are numbers of samples labeled with normal and icing class, respectively.

3) Evaluation Metrics: As the data are imbalanced, the conventional evaluation metric of "accuracy" would not have been suitable for performance evaluation. Instead, we use **Precision**, **Recall**, **F1 Score**, and the **Matthews correlation coefficient** (MCC) for the evaluation, defined as

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

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

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

$$\text{MCC} = \frac{\text{TP} \cdot \text{TN} - \text{FP} \cdot \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{12}$$

where TP, FP, FN, and TN indicate true positive, false positive, false negative, and true negative, respectively. Since it is crucial to identify as many icing periods as possible, the recall performance is more important. F1 establishes the balance between precision and recall, which is suitable for evaluating the overall performance. MCC is a specially designed binary classification metric that is particularly suitable for evaluating imbalanced classifications.

TABLE I
Variables for the SCADA Data

No.	Variable name	Description
1	wind_speed	Wind speed
2	generator_speed	Generator speed
3	power	Active power
4	wind_direction	Wind direction
5	wind_mean	Average wind direction angle within 25s
6	yaw_position	Yaw position
7	yaw_speed	Yaw speed
8	pitch1_angle	Angle of pitch 1
9	pitch2_angle	Angle of pitch 2
10	pitch3_angle	Angle of pitch 3
11	pitch1_speed	Speed of pitch 1
12	pitch2_speed	Speed of pitch 2
13	pitch3_speed	Speed of pitch 3
14	pitch1_moto_temp	Temperature of pitch motor 1
15	pitch2_moto_temp	Temperature of pitch motor 2
16	pitch3_moto_temp	Temperature of pitch motor 3
17	acceleration_x	Horizontal acceleration
18	acceleration_y	Vertical acceleration
19	environment_temp	Environment temperature
20	internal_temp	Internal temperature of nacelle
21	pitch_1_ng5_temp	Switching temperature of pitch 1
22	pitch_2_ng5_temp	Switching temperature of pitch 2
23	pitch_3_ng5_temp	Switching temperature of pitch 3
24	pitch_1_ng5_DC	DC power of pitch 1 switch charger
25	pitch_2_ng5_DC	DC power of pitch 2 switch charger
26	pitch_3_ng5_DC	DC power of pitch 3 switch charger

B. Evaluation of Anomaly Detection and Semisupervised Learning

In this study, we use the following five baselines to evaluate anomaly detection and SSL.

1) *IsForest (Isolation Forest):* We use the parameters recommended by Liu *et al.* [24], with the number of trees set at 100 and the subsampling size set at 256.
2) *One Class SVM (OC-SVM):* We use an RBF kernel, test the performances with the RBF scale parameters of {0.01, 0.05, 0.1, 0.2, 0.5}, and report the best.

TABLE II
PERFORMANCE COMPARISON WITH BASELINES FOR ANOMALY DETECTION AND SSL

Method	$\rho_u = \rho_l$				$\rho_u = 2\rho_l$				$\rho_u = 1$			
	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC
IsForest	51.7	97.2	53.2	13.2	52.6	97.7	54.8	18.7	52.9	95.6	55.3	17.8
LOF	50.2	**99.5**	50.3	2.55	50.0	**99.4**	50.1	1.24	50.3	**99.6**	50.5	5.04
OCSVM	51.2	94.9	52.2	8.26	51.7	94.7	53.2	11.5	51.9	94.1	53.4	11.6
TapNet	92.0	46.9	71.4	49.1	88.7	31.0	63.5	35.6	88.2	41.7	68.1	42.7
ImbalancedSSL	**96.1**	50.3	74.1	54.9	**96.5**	43.4	70.9	50.1	**96.3**	51.3	74.7	55.8
Ours	91.9	86.1	**89.2**	**78.6**	90.7	89.8	**90.3**	**80.5**	92.8	77.7	**85.8**	**72.7**

TABLE III
PERFORMANCE COMPARISON IN DIFFERENT NUMBERS OF LABELED SAMPLES WITH AN IMBALANCE RATIO OF 10

Method	10%				30%				50%			
	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC
Focalloss	59.4	70.5	61.2	22.9	66.9	74.6	68.8	38.0	67.6	82.8	71.6	44.3
GaussianAffinity	54.0	77.4	55.7	12.6	54.8	67.8	56.0	13.3	54.1	77.8	55.9	13.1
Class-balance	55.1	**83.3**	57.7	21.6	55.3	**94.7**	59.1	26.0	55.1	**92.9**	58.6	25.9
LDAM	72.7	51.1	66.0	33.4	76.8	71.4	74.9	49.9	81.7	70.6	77.4	55.4
WCE	65.7	55.9	63.3	27.0	68.2	76.4	70.4	41.1	76.9	68.4	73.9	48.4
Ours	**94.4**	60.5	**78.5**	**61.0**	**97.0**	81.3	**89.4**	**79.8**	**97.9**	79.0	**88.6**	**78.7**

3) *LOF:* This is an outlier detection algorithm in which the LOF is calculated to reflect the abnormal degree of test data [25].
4) *TapNet:* This is a semisupervised algorithm for TSC. TapNet is equipped with a feature extractor, which is a combination of an LSTM layer and stacked CNN layers. An attention prototypical network is implemented for SSL [41].
5) *ImbalancedSSL (Imbalanced SSL):* This is the algorithm for class-imbalanced SSL that uses ResNet as a feature extractor [45]. This algorithm offers state-of-the-art performance for class-imbalanced SSL. We replace the ResNet with our feature encoder and compare the performance.

The window size of the sensor data is 128 (15 min). The number of filters in the feature extractor is {128, 256, 128} with a kernel size of {7, 5, 3}. These parameters will be applied to the following experiments and will not be elaborated. We repeat the experiment three times and report the best outcomes. The results are shown in Table II, with the best results indicated in bold. From the table, we can see that our model achieves the highest performance with respect to F1 and MCC. In contrast, the three classical unsupervised anomaly detection algorithms are the lowest; surprisingly, however, they achieve the highest recall. Compared with TapNet, our model achieves an improvement in MCC of 60.0%, 126.1%, and 70.3% for the three imbalance ratio cases. The improvements in F1 for the three cases are 24.9%, 42.2%, and 26.0%. ImbalancedSSL is a state-of-the-art method for class-imbalanced SSL, but its performance is found to be inferior to ours in terms of F1 and MCC. However, ImbalancedSSL has the best performance in terms of precision. Regarding the effects of the different imbalance ratios, it can be seen that our model performs best when $\rho_u = 2\rho_l$.

C. Evaluation of Class-Imbalanced Learning

We compare the proposed model with five class-imbalanced learning algorithms. In this study, we use the following percentages of labeled samples, 10%, 30%, and 50%, and set the imbalance ratio [see (8)] to 10. The five algorithms are described as follows.

1) *Focalloss:* This is one of the most frequently used loss functions for class-imbalanced learning. In this study, $\gamma = 1$, and we use the same weights for each class as in [48].
2) *GaussianAffinity:* This is a flexible loss function for class-imbalanced learning. Based on the Euclidean space, affinity is defined as using the Gaussian similarity in the Bregmen divergence. GaussianAffinity is used for feature space clustering and max-margin classification based on the prototypes [53].
3) *Class-Balance:* This algorithm introduces the concept of the effective number of samples, which takes data overlap into account. The class balance loss is calculated based on the effective number of samples per class [49].
4) *Label-Distribution-Aware Margin (LDAM):* This algorithm can replace the standard cross-entropy objective during training and may be used with prior class-imbalanced training strategies, such as reweighting or resampling [50].
5) *Weighted Cross Entropy (WCE):* This algorithm uses a classical loss function for class-imbalanced learning to assign a weight to each class [45].

In this study, we evaluate the rebalancing capability of our model when only the imbalance component is used. The results are presented in Table III, indicating that our model has the highest performance in terms of precision, F1, and MCC. LDAM is ranked second, and GaussianAffinity is the least. Our model achieves an improvement of 82.6%, 59.9%, and 42.1% in MCC over LDAM for the three percentages of labeled samples and an improvement of 18.9%, 19.4%, and 14.5% in F1. Interestingly, WCE is slightly better than Focalloss, possibly due to the fact that we only used the recommended hyperparameters without tuning them. The results also show that performance increases with increasing percentages of the

TABLE IV
PERFORMANCE COMPARISON OF THE FEATURE ENCODER

Method	$\rho_u = \rho_l$				$\rho_u = 2\rho_l$				$\rho_u = 1$			
	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC
MLP	62.6	62.1	62.5	24.9	59.7	64.3	60.4	20.9	58.3	68.2	59.7	19.7
LSTM	57.3	56.8	57.2	14.4	57.3	62.5	58.0	16.0	54.6	55.6	54.7	9.3
ResNet	90.7	65.4	79.3	61.1	92.6	54.6	75.1	55.1	89.8	72.0	81.9	65.2
DenseNet	89.7	67.0	79.7	61.3	92.1	55.6	75.4	55.4	91.0	71.9	82.4	66.2
MLSTM_FCN	89.7	66.2	79.3	60.7	91.6	63.0	78.6	60.2	92.2	78.2	**85.8**	72.5
Ours	**91.9**	**86.1**	**89.2**	**78.6**	90.7	**89.8**	**90.3**	**80.5**	**92.8**	**77.7**	**85.8**	**72.7**

TABLE V
INFLUENCE OF THE IMBALANCE RATIO

Imbalance factor	$\rho_u = \rho_l$				$\rho_u = 2\rho_l$				$\rho_u = 1$			
	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC
$\rho_l = 10$	**91.9**	**86.1**	**89.2**	**78.6**	90.7	**89.8**	**90.3**	**80.5**	**92.8**	**77.7**	**85.8**	**72.7**
$\rho_l = 20$	88.3	64.2	77.9	58.7	**91.4**	61.5	77.9	59.0	85.5	64.6	76.8	57.8
$\rho_l = 50$	83.6	52.2	71.0	47.4	67.1	82.5	71.0	47.7	72.0	76.7	73.4	47.0

labeled samples. Finally, with respect to the recall, the Class-balance shows the best result.

D. Evaluation of the Feature Encoder

We use the following five neural networks as baselines for the evaluation of the proposed feature extractor. They are described as follows.

1) *MLP:* It uses the following settings: three FC layers, each with 500 hidden nodes, and a dropout layer employed between the FC layers.
2) *LSTM:* We set the hidden nodes of the LSTM to {8, 16, 32, 64}, test the performance, and report the best result.
3) *MLSTM-FCN:* This deep learning model consists of an LSTM layer and an FCN layer with an SE module. The parameters recommended in [34] are used.
4) *ResNet:* We use the same parameters as in [54].
5) *DenseNet:* This model has state-of-the-art performance according to Cheng *et al.* [12]. We use the same network structure but remove all attention layers.

The results are presented in Table IV and indicate that our model performs the best in the three test cases (except F1) when $\rho_u = 1$, as does MLSTM_FCN. MLSTM_FCN is in second place, except for in the test of $\rho_u = \rho_l$, where it is slightly less than DenseNet, possibly due to the fact that this model simultaneously learns spatial and temporal features. ResNet and DenseNet are almost identical for all three test cases. LSTM is lowest, even worse than MLP for all three test cases. For MCC, our model shows improvements of 28.2%, 33.7%, and 0.28% over the second-ranked method for all three test cases. For F1, our model shows absolute improvement levels of 9.9% and 11.7% in the test cases for $\rho_u = \rho_l$ and $\rho_u = 2\rho_l$, respectively. Another interesting finding is that the performance does not improve with the increased complexity of the network structure. For example, DenseNet has the most complex structure, but its performance is not the best. This also suggests that choosing a suitable model is more important than increasing the number of network layers in that model.

E. Sensitivity and Ablation Analysis

We describe our evaluation of the effectiveness of the proposed model in terms of sensitivity and ablation analysis.

1) Influence of the Imbalance Factor: We measure performance by using different imbalance ratios [see (8)]: $\rho_l = 10$, 20, and 50; for each imbalance ratio, we set ρ_u accordingly. The results are shown in Table V and indicate that, if the imbalance ratio is 10, the proposed model would perform the best. Furthermore, we can observe that the performance decreases when the imbalance ratio increases from 10 to 50.

2) Influence of the Number of Labeled Samples: To investigate the influence of the number of labeled samples, we vary the labeled sample percentage in the train dataset from 10% to 50%. In each case, we also perform three comparisons to examine the impact of the relationship between the imbalance ratios and labeled samples on the classification results. Table VI shows the results, from which it can be seen that, in the cases where $\rho_u = \rho_l$ and $\rho_u = 2\rho_l$, when the number of labeled samples is 50%, the model performs the best. It is interesting to see that in the case where $\rho_u = 1$, when the number of labeled samples is 50%, the model performance is actually slightly lower than when the number of labeled samples is 30%. This result suggests that there is a strong correlation between the number of labeled samples and classification performance.

3) Impact of Window Size: We also evaluate the impact of using different window sizes to split training data on model performance, including 32, 64, 128, and 256. The values of ρ_u and ρ_l are set to 20 and 10, respectively, and the ratio of labeled samples in the test dataset is set to 10%. The results in Fig. 4 show that our model can achieve the best performance with respect to F1 and MCC when the window size is set to 128. When the window size increases from 32 to 128, the values of MCC and F1 also increase, but, when the window size is 256, the performance decreases slightly. When the time window size is limited, the larger the window, the more information it obtains, and the better the performance. Therefore, it is also important to select an

TABLE VI
INFLUENCE OF THE NUMBER OF LABELED SAMPLES

Number of labeled samples	$\rho_u = \rho_l$				$\rho_u = 2\rho_l$				$\rho_u = 1$			
	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC	Precision	Recall	F1	MCC
10%	91.9	86.1	89.2	78.6	90.7	**89.8**	90.3	80.5	92.8	77.7	85.8	72.7
30%	95.5	87.3	91.6	83.5	96.1	86.9	91.7	83.7	96.7	**83.9**	**90.5**	**81.8**
50%	**97.8**	**86.8**	**92.4**	**85.4**	**97.8**	85.6	**91.8**	**84.3**	**97.7**	81.9	90.0	81.0

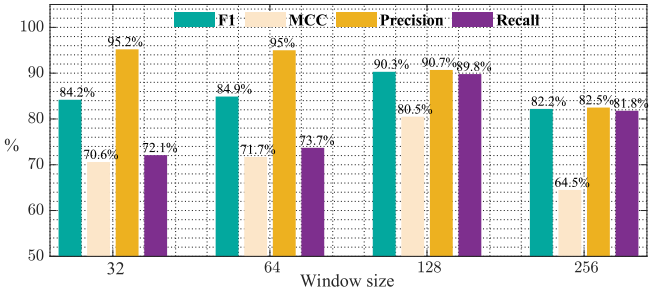

Fig. 4. Influence of different window sizes on performance.

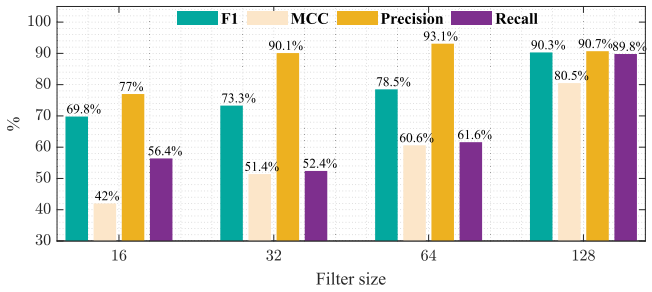

Fig. 5. Influence of different filter sizes on performance.

appropriate time window size to split the data for model training; in this study, the size is 128.

4) Impact of Filter Size: The filter size is a key hyperparameter in the feature encoder. Thus, we study its impact by varying the filter size between 16, 32, 64, and 128. The values of ρ_u and ρ_l are set to 20 and 10, respectively. The results in Fig. 5 show that the best performance is achieved when the filter size is 128. When the filter size is increased from 16 to 128, the values of MCC and F1 also increase. The results indicate that the performance of the proposed model is sensitive to the filter size. Therefore, it is crucial to select an appropriate filter size in a real-world application.

5) Effectiveness of the Channel Attention Module: To evaluate the effectiveness of the proposed channel attention module, we compare the following variants in our model.

*Squeeze-and-Excitation (**SE**):* The proposed channel attention module is replaced by the classic **SE** module [55]. *NAN:* The proposed channel attention module is removed. As shown in Fig. 6, the proposed channel attention module achieves the best performance in all three test cases.

The results show that the performances of **SE** and **NAN** are almost the same in all three test cases. It is also noteworthy that the precision of the three models does not make much of a difference, but the recalls for the other two methods are less than for our model. Regarding F1 and MCC, the three models have almost the same performance in the test case of $\rho_u = 1$. For F1, ours outperforms **NAN**, with 9.5% and 12.3% higher results for the tests of $\rho_u = \rho_l$ and $\rho_u = 2\rho_l$, respectively. For MCC, ours outperforms **NAN**, with 17.2% and 21.6% higher performance, respectively.

F. Online Detection

To evaluate the blade icing detection for wind turbines in real time, we implement an online detection scheme. In this scheme, the model used for online detection is obtained from the off-line model training process. A buffer with the same length of window size in the model training is used to accumulate the online data. When the buffer is full, the scheme identifies icing or not by computing a probability value. We use the ensemble algorithm called majority voting [8] in this scheme since this algorithm is more robust and better at eliminating random errors.

The online detection is performed by using the model trained by the data obtained from the two wind turbines described in Section IV-A and applying the detection model to a third wind turbine, which has continuous logging data around 330 h. To further evaluate the robustness of our model, we compare our proposed model and a supervised learning model in online detection. The proposed model is trained with 10% labeled data and 60% unlabeled data that are from the mixed data of two wind turbines. The supervised learning model is trained using 50% of labeled samples that are also from the mixed data of two wind turbines. The blade condition is predicted almost every 4 min, with a duration of 23 h. The results are shown in Fig. 7, where the magenta background is the icing period (icing has occurred) labeled by a turbine engineer. The green dashed line is the classical threshold (0.5) for icing detection. During the icing period (i.e., the magenta area), it can be seen that both methods can ensure almost 100% accurate prediction of icing. The supervised method alarms the icing at approximately 3500 s, while the semisupervised method does so at about 7200 s. Thus, the proposed semisupervised method outperforms the supervised learning model in predicting the normal operating state of the turbine. Compared to the supervised learning method, although our method predicts the icing at a later stage, there is still sufficient time for a turbine engineer to prepare for deicing measurements (i.e., time to start a deicing system).

V. DISCUSSION

In this research, we proposed a prototype network that unifies the class rebalancing method and an SSL method for

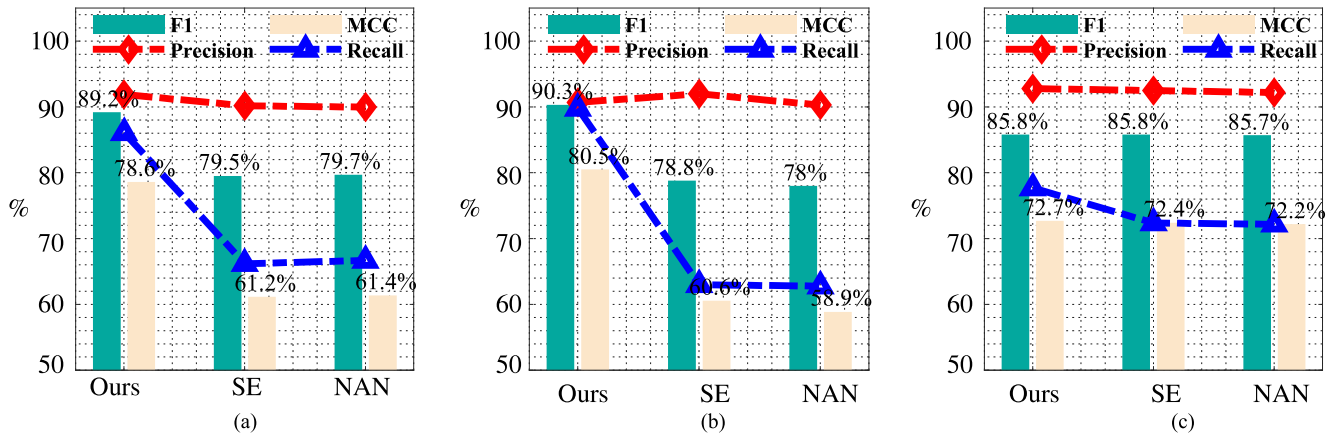

Fig. 6. Ablation study for evaluating the channel attention module. (a) $\rho_u = \rho_1$. (b) $\rho_u = 2\rho_1$. (c) $\rho_u = 1$.

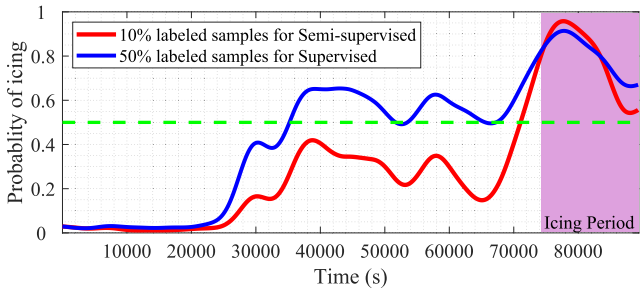

Fig. 7. Evaluation of online detection.

class estimation for unlabeled data within a single framework. Although the experiments demonstrated their effectiveness and superior performance in the detection of wind turbine blade icing, two aspects are worthy of further discussion. First, according to the above experiments, our model achieved better performance when the imbalance ratio is 10, but reduced performance when it is 20 and 50, respectively. The reason why different imbalance ratios can lead to differences in performance is interesting. It may be due to the design of the network, and it is worthy of further investigation in future work. Second, the proposed model is not suitable for determining the severity of blade icing in wind turbines. This is due to the fact that the data used for training have only two discrete states (i.e., normal or icing). Therefore, this requires further improvements in the detection of continuous states, such as any intermediate state between the normal and icing states. A possible solution for the current implementation is, as in online testing, to adjust the icing detection threshold value (i.e., the green dotted line in Fig. 7). This, however, requires domain knowledge for tuning the parameter, and users must adjust the value manually. In the future, however, it may be possible to set the hyperparameters automatically, such as by using reinforcement learning methods that learn the settings by the interaction of the environment and agent. In addition, it would also be ideal to make the proposed model more adaptable such that it is able to adapt to different types of wind turbines and other external conditions related to blade icing. We will leave this to future work seeking to improve the robustness and adaptability of the model.

VI. Conclusion

In this research, we proposed a novel deep class-imbalanced, semisupervised model for estimating the blade icing condition of wind turbines. To solve the class imbalance problem, we implemented a prototypical network that can balance the classes for labeled and unlabeled data. The prototype network presented can not only rebalance extracted features but also determine the classes of unlabeled data by measuring the similarity of the prototypes for labeled and unlabeled samples in a latent feature space. To improve the feature extraction capability, we also proposed adding an additional channel attention module to the feature extractor. In the end, we comprehensively evaluated the proposed model, including a comparison with baselines, an ablation study, and an online detection. The results demonstrate the superiority and effectiveness of the proposed model.

Acknowledgment

The authors thank Goldwind Inc. for the data sharing.

References

[1] G. M. Ibrahim, K. Pope, and Y. S. Muzychka, "Effects of blade design on ice accretion for horizontal axis wind turbines," *J. Wind Eng. Ind. Aerodyn.*, vol. 173, pp. 39–52, Feb. 2018.

[2] Y. Wang, Y. Xu, and Y. Lei, "An effect assessment and prediction method of ultrasonic de-icing for composite wind turbine blades," *Renew. Energy*, vol. 118, pp. 1015–1023, Apr. 2018.

[3] K. Wei, Y. Yang, H. Zuo, and D. Zhong, "A review on ice detection technology and ice elimination technology for wind turbine," *Wind Energy*, vol. 23, no. 3, pp. 433–457, Mar. 2020.

[4] S. A. Saleh, R. Ahshan, and C. R. Moloney, "Wavelet-based signal processing method for detecting ice accretion on wind turbines," *IEEE Trans. Sustain. Energy*, vol. 3, no. 3, pp. 585–597, Jul. 2012.

[5] O. Parent and A. Ilinca, "Anti-icing and de-icing techniques for wind turbines: Critical review," *Cold Regions Sci. Technol.*, vol. 65, no. 1, pp. 88–96, 2011.

[6] O. Fakorede, Z. Feger, H. Ibrahim, A. Ilinca, J. Perron, and C. Masson, "Ice protection systems for wind turbines in cold climate: Characteristics, comparisons and analysis," *Renew. Sustain. Energy Rev.*, vol. 65, pp. 662–675, Nov. 2016.

[7] L. Gao, Y. Liu, L. Ma, and H. Hu, "A hybrid strategy combining minimized leading-edge electric-heating and superhydro-/ice-phobic surface coating for wind turbine icing mitigation," *Renew. Energy*, vol. 140, pp. 943–956, Sep. 2019.

[8] B. Yuan *et al.*, "WaveletAE: A wavelet-enhanced autoencoder for wind turbine blade icing detection," 2019, *arXiv:1902.05625*. [Online]. Available: http://arxiv.org/abs/1902.05625

[9] N. Jolin, D. Bolduc, N. Swytink-Binnema, G. Rosso, and C. Godreau, "Wind turbine blade ice accretion: A correlation with nacelle ice accretion," *Cold Regions Sci. Technol.*, vol. 157, pp. 235–241, Jan. 2019.

[10] S. C. Pryor and R. J. Barthelmie, "Climate change impacts on wind energy: A review," *Renew. Sustain. Energy Rev.*, vol. 14, no. 1, pp. 430–437, Jan. 2010.

[11] L. Shu, G. Qiu, Q. Hu, X. Jiang, G. McClure, and H. Yang, "Numerical and field experimental investigation of wind turbine dynamic de-icing process," *J. Wind Eng. Ind. Aerodyn.*, vol. 175, pp. 90–99, Apr. 2018.

[12] X. Cheng, G. Li, A. L. Ellefsen, S. Chen, H. P. Hildre, and H. Zhang, "A novel densely connected convolutional neural network for sea-state estimation using ship motion data," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 9, pp. 5984–5993, Sep. 2020.

[13] A. A. Jiménez, F. P. G. Márquez, V. B. Moraleda, and C. Q. G. Muñoz, "Linear and nonlinear features and machine learning for wind turbine blade ice detection and diagnosis," *Renew. Energy*, vol. 132, pp. 1034–1048, Mar. 2019.

[14] L. Zhang, K. Liu, Y. Wang, and Z. Omariba, "Ice detection model of wind turbine blades based on random forest classifier," *Energies*, vol. 11, no. 10, p. 2548, Sep. 2018.

[15] Y. Liu, H. Cheng, X. Kong, Q. Wang, and H. Cui, "Intelligent wind turbine blade icing detection using supervisory control and data acquisition data and ensemble deep learning," *Energy Sci. Eng.*, vol. 7, no. 6, pp. 2633–2645, Dec. 2019.

[16] G. Jiang, H. He, J. Yan, and P. Xie, "Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3196–3207, Apr. 2019.

[17] C. Huang, X. Wu, X. Zhang, S. Lin, and N. V. Chawla, "Deep prototypical networks for imbalanced time series classification under data scarcity," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 2141–2144.

[18] M. Zhang *et al.*, "Highly transparent and robust slippery lubricant-infused porous surfaces with anti-icing and anti-fouling performances," *J. Alloys Compounds*, vol. 803, pp. 51–60, Sep. 2019.

[19] C. Peng, S. Xing, Z. Yuan, J. Xiao, C. Wang, and J. Zeng, "Preparation and anti-icing of superhydrophobic PVDF coating on a wind turbine blade," *Appl. Surf. Sci.*, vol. 259, pp. 764–768, Oct. 2012.

[20] C. Q. G. Muñoz, F. P. G. Márquez, and J. M. S. Tomás, "Ice detection using thermal infrared radiometry on wind turbine blades," *Measurement*, vol. 93, pp. 157–163, Nov. 2016.

[21] J. Zeng and B. Song, "Research on experiment and numerical simulation of ultrasonic de-icing for wind turbine blades," *Renew. Energy*, vol. 113, pp. 706–712, Dec. 2017.

[22] M. L. Corradini, A. Cristofaro, and S. Pettinari, "A model-based robust icing detection and estimation scheme for wind turbines," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2016, pp. 1451–1456.

[23] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, no. 2, pp. 1–38, 2021.

[24] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 413–422.

[25] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 93–104.

[26] G. Pang, L. Cao, L. Chen, and H. Liu, "Learning representations of ultrahigh-dimensional data for random distance-based outlier detection," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2041–2050.

[27] G. Pang, C. Shen, and A. van den Hengel, "Deep anomaly detection with deviation networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 353–362.

[28] L. Ruff *et al.*, "Deep semi-supervised anomaly detection," 2019, *arXiv:1906.02694*. [Online]. Available: http://arxiv.org/abs/1906.02694

[29] C. Orsenigo and C. Vercellis, "Combining discrete SVM and fixed cardinality warping distances for multivariate time series classification," *Pattern Recognit.*, vol. 43, no. 11, pp. 3787–3794, Nov. 2010.

[30] M. G. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2796–2802, Nov. 2013.

[31] P. Schäfer, "Scalable time series classification," *Data Mining Knowl. Discovery*, vol. 30, no. 5, pp. 1273–1298, 2016.

[32] W. Pei, H. Dibeklioğlu, D. M. J. Tax, and L. van der Maaten, "Multivariate time-series classification using the hidden-unit logistic model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 920–931, Apr. 2018.

[33] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Frontiers Comput. Sci.*, vol. 10, no. 1, pp. 96–112, Feb. 2016.

[34] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate LSTM-FCNs for time series classification," *Neural Netw.*, vol. 116, pp. 237–245, Aug. 2019.

[35] X. Cheng, G. Li, R. Skulstad, S. Chen, H. P. Hildre, and H. Zhang, "Modeling and analysis of motion data from dynamically positioned vessels for sea state estimation," in *Proc. Int. Conf. Robot. Automat. (ICRA)*, May 2019, pp. 6644–6650.

[36] M. González, C. Bergmeir, I. Triguero, Y. Rodríguez, and J. M. Benítez, "Self-labeling techniques for semi-supervised time series classification: An empirical study," *Knowl. Inf. Syst.*, vol. 55, no. 2, pp. 493–528, May 2018.

[37] P. Gianniou, X. Liu, A. Heller, P. S. Nielsen, and C. Rode, "Clustering-based analysis for residential district heating data," *Energy Convers. Manage.*, vol. 165, pp. 840–850, Jun. 2018.

[38] Y. Chen, B. Hu, E. Keogh, and G. E. A. P. A. Batista, "DTW-D: Time series semi-supervised learning from a single example," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2013, pp. 383–391.

[39] K. Marussy and K. Buza, "Success: A new approach for semi-supervised classification of time-series," in *Proc. Int. Conf. Artif. Intell. Soft Comput.* Berlin, Germany: Springer, 2013, pp. 437–447.

[40] N. Begum, B. Hu, T. Rakthanmanon, and E. Keogh, "A minimum description length technique for semi-supervised time series classification," in *Integration of Reusable Systems*. Cham, Switzerland: Springer, 2014, pp. 171–192.

[41] X. Zhang, Y. Gao, J. Lin, and C.-T. Lu, "TapNet: Multivariate time series classification with attentional prototypical network," in *Proc. AAAI*, 2020, pp. 6845–6852.

[42] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3235–3246.

[43] S. Li, Z. Wang, G. Zhou, and S. Y. M. Lee, "Semi-supervised learning for imbalanced sentiment classification," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1–6.

[44] A. Stanescu and D. Caragea, "Semi-supervised self-training approaches for imbalanced splice site datasets," in *Proc. 6th Int. Conf. Bioinf. Comput. Biol. (BICoB)*, 2014, pp. 131–136.

[45] Y. Yang and Z. Xu, "Rethinking the value of labels for improving class-imbalanced learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1–21.

[46] S. Pouyanfar *et al.*, "Dynamic sampling in convolutional neural networks for imbalanced data classification," in *Proc. IEEE Conf. Multimedia Inf. Process. Retr. (MIPR)*, Apr. 2018, pp. 112–117.

[47] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Netw.*, vol. 106, pp. 249–259, Oct. 2018.

[48] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[49] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9268–9277.

[50] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1567–1578.

[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[52] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.

[53] M. Hayat, S. Khan, W. Zamir, J. Shen, and L. Shao, "Max-margin class imbalanced learning with Gaussian affinity," in *Proc. ICCV*, 2019, pp. 1–10.

[54] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1578–1585.

[55] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

Xu Cheng (Member, IEEE) received the Ph.D. degree in engineering from the Intelligent Systems Laboratory, Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology (NTNU), Ålesund, Norway, in 2020.

He is currently a Researcher with the Department of Manufacturing and Civil Engineering, NTNU, Gjøvik, Norway. He has published more than 20 articles as a first author and a coauthor in the areas of his research interests that include data analytics and artificial intelligence in maritime operations, time-series analytics, and predictive maintenance of wind turbines.

Fan Shi (Member, IEEE) received the Ph.D. degree from Nankai University, Tianjin, China, in 2012.

From June 2018 to August 2019, he was a Research Scholar with West Virginia University, Morgantown, WV, USA. He is currently an Associate Professor with Tianjin University of Technology, Tianjin, China. His research interests include machine vision, pattern recognition, and optics.

Xiufeng Liu received the Ph.D. degree in computer science from Aalborg University, Aalborg, Denmark, in 2012.

He was a Post-Doctoral Researcher with the University of Waterloo, Waterloo, ON, Canada, from 2013 to 2014. He is currently a Senior Researcher with the Department of Management Engineering, Technical University of Denmark (DTU), Kongens Lyngby, Denmark. His research interests include smart meter data analytics, data warehousing, energy informatics, and big data.

Meng Zhao (Member, IEEE) received the B.S. degree in automation and the M.S. and Ph.D. degrees in control science and engineering from Tianjin University, Tianjin, China, in 2010, 2016, and 2016, respectively.

Since 2016, she has been a Lecturer with the School of Computer Science and Engineering, Tianjin University of China, Tianjin. From May 2019 to April 2020, she held a post-doctoral position supported by the European Research Consortium for Informatics and Mathematics (ERCIM) "Alain Bensoussan Fellowship Programme." Her research interests include medical image processing, medical/biomedical engineering, and machine learning/deep learning in medical informatics.

Shengyong Chen (Senior Member, IEEE) received the Ph.D. degree in computer vision from the City University of Hong Kong, Hong Kong, in 2003.

He was a Visiting Professor with Imperial College London, London, U.K., and the University of Cambridge, Cambridge, U.K. He was with the University of Hamburg, Hamburg, Germany, from 2006 to 2007. He is currently a Professor with Tianjin University of Technology, Tianjin, China. He published over 400 scientific articles in international journals and conferences, with 50 in IEEE TRANSACTIONS. His selected publications can be found at his website http://sychen.com. He holds over 100 patents. His research interests include autonomous robots, computer vision, and machine intelligence.

Dr. Chen is also an IET Fellow. He received the Fellowship from the Alexander von Humboldt Foundation of Germany. He received the National Outstanding Youth Foundation Award of China in 2013, the IEEE SENSORS JOURNAL Best Paper Award in 2012, and the IET Premium Award in 2013. He organized over 20 international conferences and serves as an Associate Editor for three international journals, e.g., IEEE TRANSACTIONS ON CYBERNETICS.