Prediction of self-healing in engineered cementitious composite: Machine learning comparative analyses

Guangwei Chen 1*,*2*,*∗, WaiChing Tang 1, Shuo Chen 1, Shangyong Wang 1, Hongzhi Cui3

1The University of Newcastle, Callaghan, 2308, NSW, Australia

2Qiannan Normal College of Nationalities, Guizhou, China

3College of Civil and Transportation Engineering, Shenzhen University, Shenzhen 518060, China

**Abstract**

Engineered cementitious composite (ECC) is a unique material which can significantly contribute to self-healing of cracks due to the continued hydration process with time. Machine learning (ML) models, as a model-free approach, have been widely used to model the mechanical properties of concrete as they do not require any predefined prediction model. However, the literature on ML in predicting self-healing of ECC is very limited. This paper aims to provide a comparative analysis on performance of several ML approaches in predicting self-healing of ECC. These approaches include four individual methods (linear regression (LR), back-propagation neural network (BPNN), classification and regression tree (CRAT) and support vector regression(SVR) and three ensemble methods (bagging, AdaBoost, and stacking) with each of the four individual models used as the based learner. A series of experimental testing of self-healing performance of ECC samples was conducted and the results were used to compare the accuracy among the ML models. The results shows that the stacking model performs the best with the lowest error and highest accuracy on the self-healing prediction of ECC, compared to the experimental results. On average, ensemble methods improve the prediction performance of the individual models.

**Keywords** ECC, self-healing, machine learning, ensemble method

# Introduction

According to a research project commissioned by Materials for Life (M4L), the issues associated with cracking in concrete were experienced by more clients, design team members and contractors than any other problems [1]. Moreover, cracks are primarily responsible for the reduction of strength and stiffness of concrete. In European countries, the annual cost spent on maintenance, refurbishment and repair of concrete cracks in prolonging the service life of infrastructure is estimated around 50% of their annual construction budget [2]. It has been suggested by M4L that self-healing cementitious materials is of great potential to address the problems associated with concrete cracking and reduce the maintenance costs over a structure’s lifetime [1].

The inspiration of self-healing comes from the biomimicry concept and the healing process in living nature [3]. For example, the skin of humans or animals can biologically repair itself from simple injuries. In cement-based materials, the process of crack self-healing can be categorised into two major mechanisms, autogenous healing and autonomous healing [4]. The former indicates the self-healing ability resulted from the physical and/or chemical composition of the cementitious matrix, whereas the self healing mechanism of the latter is triggered by some biological agents, such as bacteria which are deliberately introduced into the cementitious matrix.

Generally, the autogenous self-healing of concrete is mainly controlled by two mechanisms including (1) further hydration of cement particles and/or swelling of calcium silicate hydrate; (2) calcium hydroxide carbonation [5]. It has been reported that the crack widths of 10*µ*m [6], 100 *µ*m [7], 200 *µ*m [8], 205*µ*m [5] and 300 *µ*m [9] of ECC can be self-healed completely [10].

Engineered cementitious composite (ECC) is a high performance fiber-reinforced cementitious composite and its matrix design is strongly associated with the autogenous self-healing mechanism [11]. ECC features high tensile ductility with a typical fiber-volume fraction of 2% [12, 13] to promote self-healing ability [4].

The intrinsic self-healing ability of ECC is complex and difficult to predict because of different mineral admixture types, interactivity between different composites in the cementitious matrix and its interaction with the exposed environment [14], and unpredictable crack location, orientation and width [15]. Previous studies have explored the influence of several factors such as limestone powders (LP) [16, 17], fly ash (FA) [18, 19], hydrated lime [20], water/binder ratio [21], water permeation [22] and different curing conditions (air, carbon dioxide, wet/dry and water) [23] on self-healing behavior of ECC. However, the relationship between multiple factors is unclear and non-linear, so it’s difficult to predict self-healing of ECC mathematically based on the available data. Moreover, mathematical models based on empirical data are generally in regression forms which cannot be used when the problem (e.g. prediction of self-healing potential of ECC) contains too many independent variables because of less accuracy and more assumption in the regression form (linear, non-linear, etc.) [24].

To compensate for the drawbacks of mathematical models, machine learning (ML) techniques have been used for solving many civil engineering problems with multiple variables. Many research works have been conducted using ML algorithms for the prediction of various properties of concrete. Gilan et al.[25] developed a hybrid Support Vector Regression (SVR) - Particle Swarm Optimization (PSO) algorithm model to predict the compressive strength and rapid chloride penetration test (RCPT) results of concretes containing metakaolin. Yan et al. [26] predicted bond strength of glass fiber-reinforced polymer bar in concrete by Artificial Neural Network (ANN) with Genetic Algorithm (GA). Yaseen et al. [27] proposed a ML method called Extreme Learning Machine (ELM) to predict the compressive strength of lightweight foamed concrete.

In the literature, the performance of various ML algorithms in predicting concrete properties have been evaluated and compared. Yan and Shi [28] reported that SVR performed better than other individual methods in predicting elastic modulus of normal and high strength concretes. Chou [29] compared the performance of individual and ensemble methods for predicting the mechanical properties of high performance concrete. Reuter et al. [30] employed three individual approaches for modeling concrete failure surfaces. Sobhani et al. [31] reported that ANN and a proposed fuzzy inference system are more reliable than traditional regression models in predicting no-slump concrete. Omran et al. [32] predicted the compressive strength of environmentally friendly concrete by using three individual methods, two ensemble methods, and four regression tree models.

Although different ML algorithms have been utilized to predict various properties of concrete, the application of ML on self-healing prediction of ECC is considerably rare. Recently, Mauludin and Oucif [33] reviewed the methods used for modeling autogenous self-healing of concrete, and stated that the methods can be classified into two categories: (1) numerical simulation and (2) ML. However, the only ML model reviewed in their study was GA-ANN proposed by Ramadan et al. [3] to predict the self-healing ability of cement-based materials using dataset collected from literature. The results showed that the proposed GA–ANN model was capable of capturing the complex effects of various self-healing agents (e.g., biochemical material, silica-based additive, expansive and crystalline components) on self-healing performance of cement-based materials. However, they didn’t compare their prediction performance with linear regression or neural network model to present the advantage of proposed GA-ANN model eliminating the data dependence of models.

Wang et al. [35] used ANN and ensemble classification models for a binary prediction (healed or not) of self-healing materials based on 30 samples from literature. However, their regression models were conducted by numerical simulations which assumed the relationship between the input and output variables following some logistic regression models. More recently, Chaitanya et al. [36] used an ANN model to predict the self-healing property of concrete containing ground granulated blast furnace slag in terms of compressive strength recovery based on 51 samples collected from their experimental studies. However, both studies [35] and [36] are lack of validation and testing dataset, and comparison with other benchmark models such as linear regression model. Particularly, using a limited number of samples (data) may not be enough for training a reliable model.

Zhuang and Zhou [34] conducted a comparative study on six ML algorithms including SVR, Decision Tree Regression (DTR), Gradient Boosting Regression (GBR), ANN, Bayesian Ridge Regression (BRR) and Kernel Ridge Regression (KRR) for crack-repairing capacity of the bacteria-based self-healing concrete. The results showed that GBR performed much better than other models with 0.93 and 0.74 as the *R*2 values of the training set and testing set, respectively. The *R*2 values of most models were less than 0.7 on both training and testing set. In their study, extensive experiments with different combinations of influencing variables were utilized to generate the empirical dataset. However, only three variables including the number of bacteria, the healing time and the initial crack width, were selected to predict the crack closure percentage as the output.

To the best of our knowledge, no study to date have applied ML model to predict the self-healing of ECC. The information about prediction performance of individual and ensemble ML models on self-healing of ECC are useful to the design of ECC with self-healing capacity. Thus, the objective of this paper is to propose and compare multiple ML techniques for predicting the self-healing behavior of ECC. The ML model with the best performance can be used as a baseline prediction model for developing advanced models in the future.

In this paper, four individual ML methods including linear regression (LR), SVR, back-propagation neural network (BPNN), and classification and regression tree (CART) were proposed to predict the self-healing capability of ECC. To improve prediction accuracy of each individual model, three ensemble methods namely bagging, AdaBoost and stacking were used. A series of experimental testing of self-healing performance of ECC samples was conducted and the results were used to develop the ML models and compare the accuracy among the ML models. Experimental data collected from experiments were first preprocessed and then divided into a 10-fold cross-validation algorithm (details refer to Section 4.1) to avoid overfitting and thus the bias associated with various ranges of data were minimized. Figure 1 summarizes the steps that were performed for predicting the self-healing capability of ECC.

This paper is organized as follows. Section 2 presents the experimental program detailing the materials used for ECC specimen preparation and the test set-up for crack data measurement. The concepts and formulations of individual and ensemble models used for predicting the self-healing capability of ECC are presented in Section 3, whereas the validation and evaluation methods are described in Section 4. In Section 5, the computational results are presented and compared, and the model with best prediction performance is identified. Finally, Section 6 draws the major conclusions from this work and suggests some directions for future research.

10-fold cross-validation

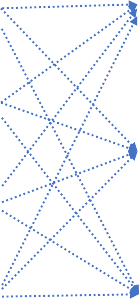
4

Preprocessing

Data

Individual models

Ensemble models



1

LR

Bagging

2

3

SVR

Output results

Input data

Prediction results

6 AdaBoost

7

8

BPNN

9

CRAT

Stacking

Output results

Input data

10

5

Figure 1: Flow chart of implementing prediction models for self-healing capability of ECC

# Experimental Program

## Materials and Mixture Proportion

In the experimental part, samples of ECC with different mineral admixtures were prepared. The materials used included general purpose cement (GPC), fly ash (FA), silica fume (SF), hydrated lime powder(LP), fine sand, polyvinyl alcohol (PVA) fibers, as well as water and high range water reducing admixture (HRWR). GPC and FA were supplied by Boral in accordance with Australian Standard AS 3972-2010, while LP was the Adelaide Brighton Hydrated Lime with a specific gravity of 2.2-2.3, and a typical fineness of 0.1% retained on a 75*µm* sieve and less than 0.05% on a 250*µm* sieve. The physical and chemical properties of cementitious materials are shown in Table 1. Fine sand with an average grain size of 150 *µ*m and a fineness modulus of 2.01 was used. The PVA fibers were supplied by Domocrete and their mechanical and geometrical properties are described in Table 2.

All ECC mixtures were prepared with a constant water to cementitious materials (W/CM) ratio of 0.29 and a constant sand to CM (PC + FA + LP+SF) ratio of 0.36. All fine aggregates were in saturated surface dried condition prior to mixing. The abbreviations for labelling specimens were adopted in such a way that the letters FA, SF and LP stand for samples with fly ash, silica fume and limestone as binder materials, respectively. The number after the letters shows the percentage of materials into the binder system. For example, the FA70 mixture is related to an ECC sample with binder containing 70% FA by weight, whereas FA60-SF10 was the mixture with 60% FA and 10% SF. A total of nine ECC mixtures were prepared and the details of mix proportion are shown in Table 3.

## Sample preparation and crack measurement

During the mixing process, the solid ingredients including cement, mineral admixtures and sand were initially placed into a planetary-type mixer of 50 L capacity and dry mixed for 30 seconds. Then, the water with HRWR was added and the mixture was mixed for 2 minutes. After that, the PVA fibers were slowly added and mixing was continued until uniform distribution of fibers in the mix. After mixing, ECC pastes were cast into standard moulds with dimension of Ø100*mm* × 200*mm*. The specimens were demolded 24 hours after casting and stored in a curing room with a temperature of 23 ± 2◦*C* and the relative humidity (RH) of 90 ± 5% for 28

Table 1: Physical and chemical properties of cementitious materials

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Chemical composition* (%) | | | GPC | FA | LP | SF | |
| Silica (SiO2) | | | 19.8 | 65.90 | 1.8 | 95.10 | |
| Alumina (Al2O3) | | | 5.3 | 24.0 | 0.5 | 0.21 | |
| Iron oxide (Fe2O3) | | | 3.0 | 2.87 | 0.6 | 0.29 | |
| Calcium oxide (CaO) | | | 64.2 | 1.59 | 72.0 | - | |
| Magnesia (MgO) | | | 1.3 | 0.42 | 1.0 | - | |
| R2O | | | 0.6 | 1.93 | - | - | |
| Sulfur trioxide (SO3) | | | 2.7 | - | - | - | |
| Titanium oxide (TiO2) | | | 0.28 | 0.91 | - | - | |
| Manganic oxide (Mn2O3) | | | 0.22 | - | - | - | |
| Zirconia (ZrO2) + Hafnium (HfO2) | | | - | - | - | 3.46 | |
| Loss on ignition (%) | | | 2.8 | 1.53 | 24.0 | 1.4 | |
| Density (*g/cm*3) | | | 3.08 | 2.43 | 2.25 | 2.26 | |
| Specific surface area (*m*2*/kg*) | | | - | 655 | 460 | 1*.*5 × 104 | |
| Table 2: Properties of PVA | | | | | | | |
| Length | Length/ | Young’s modulus | Elongation | | Tensile strength | | Density |
| (mm) | diameter ratio | (MPa) | (%) | | (MPa) | | (g/cm3) |
| 8 | 200 | 42000 | 7 | | 1600 | | 1.3 |

Table 3: Mix proportion of all ECC mixtures

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Mix | Water/CM | Sand | Water | fibre (V) | GPC | Fly ash | SF | LP | HRWR |
| FA70 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 816.03 | 0.00 | - | 5.13 |
| FA65-SF5 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 757.74 | 58.29 | - | 5.13 |
| FA60-SF10 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 699.45 | 116.58 | - | 5.13 |
| FA55-SF15 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 641.16 | 174.86 | - | 5.13 |
| FA65-LP5 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 757.74 | - | 58.29 | 5.13 |
| FA60-LP10 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 699.45 | - | 116.58 | 5.13 |
| FA55-LP15 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 641.16 - | 174.86 | 5.13 |  |
| FA55-SF5-LP10 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 641.16 | 58.29 | 116.58 | 5.13 |
| FA55-SF10-LP5 | 0.29 | 419.67 | 338.07 | 26 | 349.73 | 641.16 | 116.58 | 58.29 | 5.13 |

days. To prepare splitting tensile test samples, the cylinder specimens were cut into specimens with a diameter of 100*mm* and a thickness of50*mm* using a diamond blade saw.

×

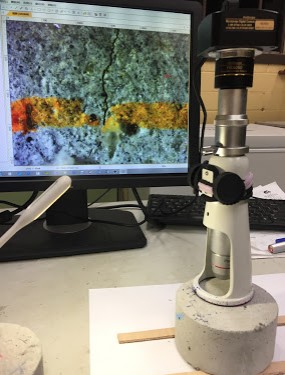
A newly developed splitting tensile test apparatus was used to generate micro-cracks and control the influence on crack differences, as shown in Figure 2 (a). It consisted of a steel frame, top member, bottom member, prestressed loading steel plates (5*mm* thick) on both sides with loading nuts and wire springs. Both steel plates were connected to the steel frame by nuts and wire springs. The specimen was placed inside the steel frame and then pre-stressed by the steel plates from both sides limiting the propagation and size of crack and preventing excessive crack growth.

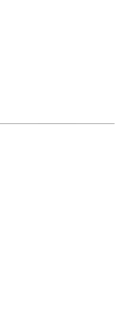
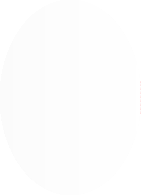
Micro-cracks less than 150*µm* were first produced by pre-loading the ECC samples up to 70% of their maximum splitting strength. A digital microscope was used to measure the crack width on the surface of specimens as shown

**Spring**

**Specimen**

**Crack observation areas**

**Prestressed loading steel**



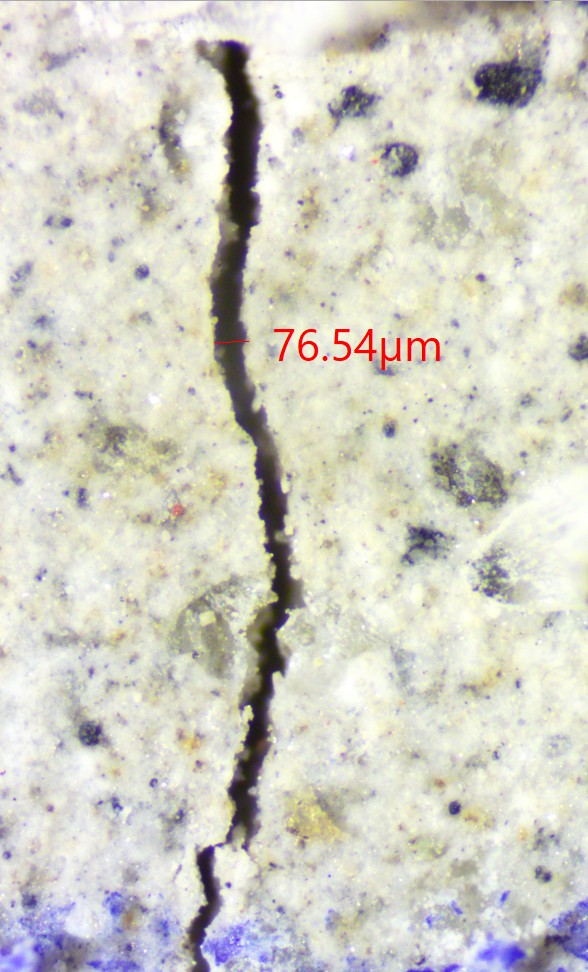
(a) (b) (c)

Figure 2: a) Splitting tensile test apparatus, b) schematic diagram of apparatus and c) microscope for measuring ECC cracks

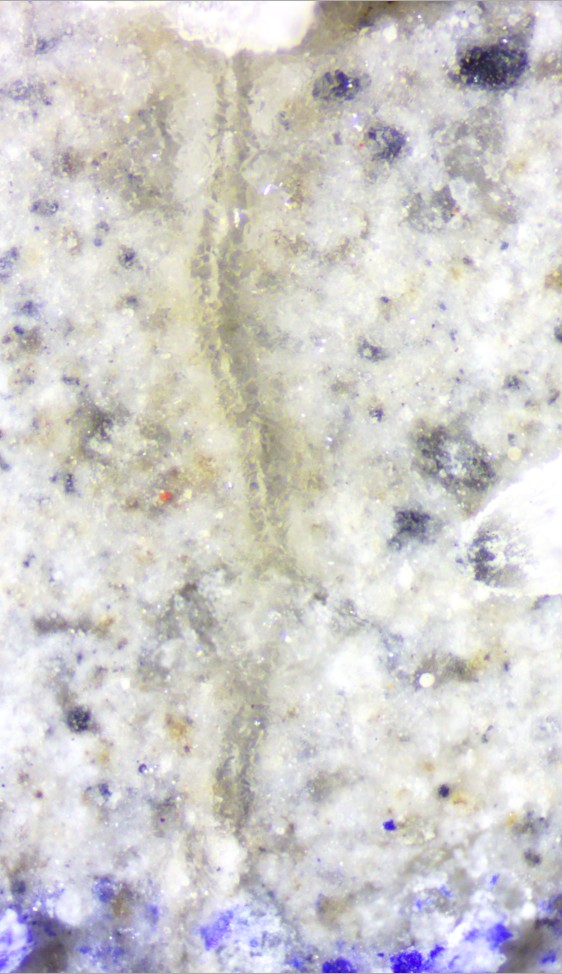
in Figure 2 (c). After the pre-loading, the cracked specimens were subjected to wet-dry (W/D) cycles to promote self-healing. Each W/D cycle consisted of submersion in water at a temperature of 23.2°Cfor 24 hours and drying in a controlled environment at 23 2°Cand a *RH* of50.5% for 24 hours. After 10 W/D cycles, the cracks were measured again by the digital microscope to examine partial or full closure of crack. Figure 3 illustrates the self-healing of cracks of an ECC mixture specimen before and after the 10 W/D cycles.

±

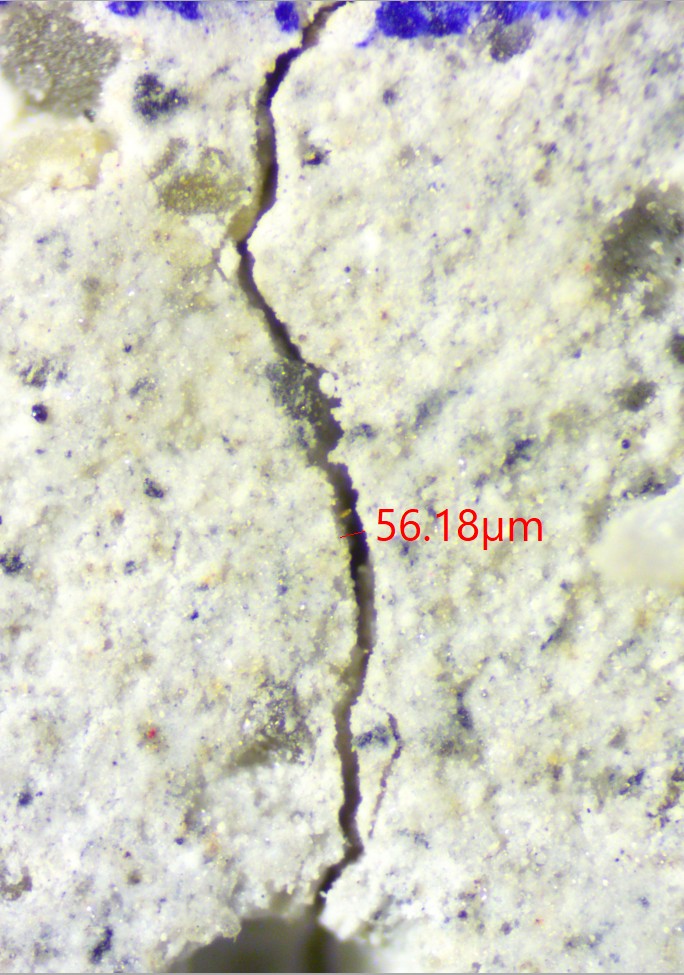
± ±



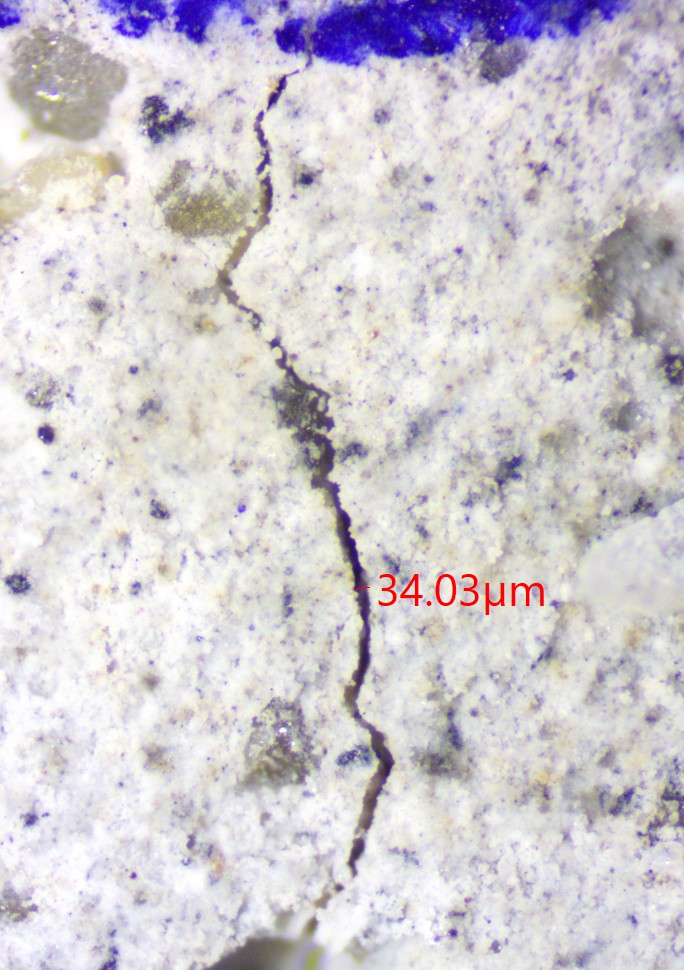
100μm



100μm



100μm



100μm

(a) S1: crack before self-healing (b) S1: crack after self-healing (c) S2: crack before self-healing (d) S2: crack after self-healing

Figure 3: Comparison of crack width changes in an ECC specimen, a) before and b) after self- healing

## Data Collection

Empirical data for prediction were gathered with four features, crack width before self-healing representing the influencing factor of self-healing, and the mineral contents of FA, SF, and LP illustrating the variable composites of ECC. It is noteworthy that some impact factors in the test, such as GPC, sand, W/CM, and healing time were kept constant and would have no effect on predction outputs and thus they were excluded from the prediction-model construction. For each ECC mix, crack data of six identical specimens were observed using digital microscope before and after self-healing. To collect enough crack data, four horizontal lines were drawn on the surface of each specimen along the direction of vertical force, which divided the specimen

×

OA 1

OA 2

OA 3

100 mm

20 mm

OA 4

OA 5

OA = Observation area

Figure 4: Schematic diagram of measuring observation areas on the surface of ECC mixture specimen

into five observation areas as shown in Figure 4. In each observation area, only one crack width was collected if the crack size showed little or no change along the vertical force, otherwise multiple crack data up to XX cracks in each area were examined. Totally, 617 crack data were collected [xx] and used to construct the ML training-testing dataset. Table 4 shows the number of collected samples and range of crack width before and after self-healing in each mixture.

Table 4: Number of crack samples and range of crack width before and after self-healing collected from the ECC mixes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | Crack width | before self-healing | Crack width | after self-healing |  |
| Mix | Number | of | crack | Min (*µm*) | Max (*µm*) | Min (*µm*) | Max (*µm*) |
|  | samples |  |  |  |  |  |  |
| FA70 | 87 |  |  | 3.28 | 134.69 | 0 | 121.37 |  |
| FA65-SF5 | 77 |  |  | 4.37 | 135.47 | 0 | 124.01 |  |
| FA60-SF10 | 88 |  |  | 5.18 | 121.78 | 0 | 113.11 |  |
| FA55-SF15 | 88 |  |  | 3.45 | 115.8 | 0 | 109.53 |  |
| FA65-LP5 | 112 |  |  | 7.65 | 119.45 | 0 | 105.65 |  |
| FA60-LP10 | 37 |  |  | 5.62 | 126.82 | 0 | 110.97 |  |
| FA55-LP15 | 61 |  |  | 6.42 | 132.65 | 0 | 115.95 |  |
| FA55-SF5-LP10 | 34 |  |  | 8.74 | 123.09 | 0 | 110.78 |  |
| FA55-SF10-LP5 | 33 |  |  | 4.64 | 131.57 | 0 | 119.79 |  |

## Preprocessing of Data

The input and output data of different features (referring to Tables 3 and 4) vary in range and units which weigh all features unequally for prediction models and might end up creating bias. To eliminate this effect, we preprocessed empirical data to the range [0,1] by the min-max scaling presented in the following function.

*x*j = *x* − *xmin*

*xmax* − *xmin*

(1)

Where *x*j was the scaled value of the variable *x*, *xmax, xmin* were the maximum and minimum values of variable *x* respectively.

# Methodology

The ML techniques used to predict the self-healing capability of ECC were four individual methods including LR, SVR, BPNN and CART, as well as three ensemble methods including bagging, AdaBoost and stacking. Ensemble methods were constructed using individual methods as base estimators to predict the self-healing capability of ECC. To establish a baseline for comparison, the modeling parameters were set to the same in both individual models and ensemble models. The reason of choosing these techniques was due to their popularity and some of them were even recognized as the top data mining algorithms in related field of concrete [29]. The proposed individual and ensemble techniques are described in the following subsections.

## Linear Regression

LR attempts to determine the relationship between a dependent variable (response variable) and one or more independent variables (explanatory variables) by fitting a linear regression equation [37]. Given our dataset *T* = (*xi, yi*)*, i* = 1*,* 2*, ..., n* , where *n* = 617 was the size of sample dataset. *xi Rn* was independent variables representing a sample of selected features from FA, SF, LP and crack width before self-healing, *Rn* was *n*-dimensional space, *yi R*1 was the target output (crack width after self-healing) that corresponded to *xi*. Let *d* = 4 as the number of an independent variable of an random vector *x* = *x*1; *x*2; *...*; *xd* , and *y* was the corresponding output ( dependent variable). The general formula of LR for predicting self-healing capability of ECC can be expressed as follows:

∈

{ } ∈

{ }

*y* = *w*1*x*1 + *w*2*x*2 + *......* + *wdxd* + *b* (2) where *wi,* (*i* = 1*,* 2*, ..., d*) was denoted as the regression coefficient and *b* was an error term.

## Support Vector Regression

The support vector machine (SVM) is a supervised machine learning method first introduced by Vapnik [38, 39] based on statistical learning theory [40]. Since then, it has gained popularity due to attractive features and promising empirical performance. SVM includes two main categories: support vector classification (SVC) and SVR. For classification purpose, SVMs often used the kernel functions to map the input data as vectors to a high-dimensional feature space so that an optimal separating hyperplane can be constructed [41]. By implementing the structural risk minimization (SRM), SVC can obtain good generalization.

For regression purpose, the basic idea is to provide a nonlinear function by mapping input data into a high-dimensional feature space where a special type of hyperplane is constructed. After that, a regression model

is established in the hyperplane [42]. It implements approximately the SRM to set up an upper bound of the generalization error so as to achieve generalized performance [43].

Given our dataset *T* = (*xi, yi*)*, i* = 1*,* 2*, ..., n* , where *n* = 617 was the size of sample dataset, *xi Rn* was the input vector representing a sample of selected features from FA, SF, LP and crack width before self-healing, *Rn* was the *n*-dimensional vector space, *yi R*1 was the target output indicating crack width after self-healing that corresponded to *xi*. The SVR aimed to seek an optimum regression function *f* (*x*) with minimal empirical risk, as expressed as follow:

∈

{ } ∈

*f* (*x*) = ⟨*w, x*⟩ + *b* with *w* ∈ *T, b* ∈ *R* (3)

where *,* wasdenoted as the dot product in *T* , *w* and *b* were the weight vector and bias value, respectively, and they which were estimated by minimizing the empirical risk, that was, the distance between the predicted crack width and the target crack width after self-healing.

⟨· ·⟩

SVR adopts an *ϵ*-insensitive loss function penalizing predictions that has a distance between the predicted crack width and the target crack width when the self-healing is greater than *ϵ*. Therefore, the problem of finding *w* and *b* to reduce the empirical risk with respect to an *ϵ*-insensitive loss function is equivalent to the convex optimization problem that minimizes the margin (*w*) with the full prediction error within the range of *ϵ*. Then this problem can be expressed as:

minimize 1 ||*w*||2

2

.

subject to *yi* − ⟨*w, xi*⟩ − *b* ≤ *ϵ*

⟨*w, xi*⟩ + *b* − *yi* ≤ *ϵ*

(4)

By introducing slack variables *ξ* and *ξi*∗ to allow some errors to cope with infeasible solution of the optimization problem, the formulation can be generated as [39]:

minimize 1 ||*w*||2 + *C* Σ(*ξ* + *ξ*∗)

*i*=1

*n*



2

*i*

subject to

*yi* − ⟨*w, xi*⟩ − *b* ≤ *ϵ* + *ξi*

⟨*w, xi*⟩ + *b* − *yi* ≤ *ϵ* + *ξi*∗ *ξi, ξi* ≥ 0

 ∗

(5)

The constant *C* was the penalty value imposed on predictions that lied outside the *ϵ* margin. A Lagrange can be constructed from objective function and all constraints after introducing a dual set of variables as follows [44]:

*L* = 1 ||*w*||2 + *C* Σ(*ξ* + *ξ*∗) − Σ(*η ξ*

*n*

*n*

+ *η*∗*ξ*∗)

*P* 2 *i i*

*i*=1

*n*

Σ

*i i i i*

*i*=1

* *αi*(*ϵ* + *ξi* − *yi* + ⟨*w, xi*⟩ + *b*)

*i*=1 *n*

Σ

* *αi*∗(*ϵ* + *ξi*∗ + *yi* − ⟨*w, xi*⟩ − *b*)

*i*=1

*s.t. αi, αi*∗*, ηi, ηi*∗ ≥ 0

Where *LP* was the Lagrangian and *αi, αi*∗*, ηi, ηi*∗ were Lagrange multipliers.

*.* (6)

The optimality can be achieved by the partial derivatives of *LP* with respect to the primal variables following the saddle point condition. Then the function of SVR is obtained as:

*n*

Σ

*f* (*x*) = (*αi* − *αi*∗)⟨*xi, x*⟩ + *b* (7)

*i*=1

As for the nonlinear regression, the input data have to be mapped into a high-dimensional feature space, in which the dot product can be replaced by a kernel function *k*(*xi, xj*) = *φ*(*xi*)*T φ*(*xj*), and the function (7) can be written as:

*n*

Σ

*f* (*x*) = (*αi* − *αi*∗)*k*(*xi, x*) + *b* (8)

*i*=1

Different SVM algorithms use differing kinds of kernel functions such as linear, polynomial, radial basis function and sigmoid kernel.

In this work, the Gaussian radial basis function (RBF) was chosen, which is defined as [45]:

||*xi* − *xj*||2

*k*(*xi, xj*) = *exp*(− 2*σ*2 ) (9)

## Artificial Neural Network

Artificial neural network (ANN), also called neural network, is originated from simulating biological neural networks. Generally, it consists of many neurons in layers including one input layer, one or several hidden layers and an output layer [46]. The neurons are fully interconnected between the neighboring layers by weight, and typically no inter-connections between neurons within the same layer [47].

There are many possible network structures available, BPNN was utilized in this study because of. A preliminary architecture of the BPNN was determined to be 4 - *n* - 1, where 4 input neurons represented the input features standing for FA, LP, SF and crack width before self-healing, *n* = 5 indicated the number of neurons in the hidden layer, and 1 target neuron in the output layer for the predicted crack width after self-healing. This is a three-layer network with one hidden layer capable to approximate most continuous functions, of which the complex nonlinear relationship could be approximated in accuracy [26]. The architecture of the BPNN model for predicting self-healing is demonstrated in Figure 5.

Given a set of inputs *x*1*, x*2*, x*3*, ..., xn* , while information was passed through the input layer to the hidden layer, each neuron in the input layer was multiplied by respective weights added by a bias and are summed together. After that, an activation function *f* was applied to form the output *z*. This can be expressed in the following equation [24]:

{ }

*n*

Σ

*z* = *f* ( *wijxi* + *bj*) (10)

*i*=1

where *wij* was the connection weights between the *i*th neuron of input and the *j*th neuron in the hidden layer, and *bj* was the bias of the *j*th neuron. The sigmoid function was applied as the activation function between the input, hidden, and output neurons to form the output.

1

*f* (*x*) = 1 + *e*−*x* (11)

The goal of training a neural network is to determine the values of the connection weights and the biases of the neurons. The back propagation indicates an iterated method to adjust the weights from output layer to input layer. At first, the outputs were calculated feed-forward from the input layer via the hidden layer to the output layer. Then an error was generated by comparing the output with the target output. After that, the error was back propagated to the hidden layer and input layer. By adjusting the connection weights and biases, the error was further reduced. The process was repeated until the error was minimized or reaching the termination to avoid over-fitting.

**Feed-forward**



**Fly ash**



**Silica fume Hydrated lime Initial crack width**

**Final Crack width**

**Output Layer**

**Input Layer Hidden layer Back-propagation**



Figure 5: Schematic diagram of BPNN model for predicting self-healing capability of ECC

## Classification and Regression Tree

The CART [48] is a tree decision algorithm that split data into mutually exclusive subgroups based on a recursive binary partitioning procedure. It develops the relationship between the target variables (the crack width after self-healing of ECC) and the independent variables (the input features of FA, SF, LP and crack width before self-healing of ECC). Thus, decision rules are created to form subgroups as branches and leaves. Figure 6 illustrates the schematic diagram of a decision tree. The process of CART starts from the root node which contains the entire dataset to construct two sub-nodes representing two categories. Then this recursion process is applied to each sub-node until all divided sub-nodes are leaf nodes. The CART tree can be either a classification tree [49] or regression tree [50] depending on the type of target and independent variables which may be categorical or numerical.

The key idea of constructing a CRAT tree is achieved by selecting a variable at each node that best splits the empirical data. To locate splits, *Gini* index was used to measure the impurity of the two child nodes containing subsets of data that were as homogeneous as possible with respect to the target variable.

Given a dataset had *K* classes and the probability of a sample which belongs to class *i* is *pi*, *i* 1*,* 2*,* 3*, ..., K* , the *Gini* impurity can be expressed as,

∈ { }

*K K*

*G*(*p*) = Σ *pi*(1 − *pi*) = 1 − Σ *p*2

*i*

(12)

## Ensemble Methods

*i*=1

*i*=1

In contrast to many ML approaches such as SVM and CART (which develop a single learner from training data), ensemble methods train multiple base learners and combine them [29] to improve generalizability over a single estimator. Therefore, weak learners (base learners) can be boosted to become strong learner [51] in an ensemble method. The base learners in an ensemble were developed from an individual learning algorithm such as decision tree, SVM, or other kinds of learning algorithms. Breiman [52] showed that ensemble methods are usually more accurate than individual learning methods



Node 0 (Root node)

***Split 1***

Node 1

Node 2

***Split 2***

***Split 3***

Node 3 Node 4 Node 5 Node 6

***Leaf 1***

***Leaf 2***

***Leaf 3***

***Split 4***

Node 7

Node 8

***Leaf 4 Leaf 5***

Figure 6: Structure of a classification and regression tree [50]

The input features of FA, SF, LP, and crack width before self-healing of ECC were considered as the *d*-dimensional predictor variable *X,* whereas, the crack widths after self-healing of ECC were the one dimensional output *Y.*  Each estimator used an individual algorithm to provide one estimated function *g*( ). The output presented by ensemble-based function *gen*( ) was obtained by a linear combination of individual functions. This ensemble approach can be expressed mathematically as:

·

·

*N*

Σ

*gen*(·) = *cj* ∗ *g*(·) (13)

*j*=1

Where *cj was* expressed as the combination coefficients, dependent on the used ensemble models.

### Bagging

Bagging method (bootstrap aggregating) can generate multiple versions of a predictor to obtain an aggregated predictor [53]. It generates multiple models independently on different versions of dataset via random bootstrapping of the original training set. In other words, several training examples may repeatedly appear in different bootstrap replicates. Then the individual predictions are aggregated through a combination method (either voting or averaging) to form the final prediction. Bagging method can be used to reduce the variance of a base estimator (e.g. a regression tree), by introducing randomization into its construction procedure and making an ensemble out of it.

### AdaBoost

Similar to bagging, AdaBoost method [54] manipulates the training examples to generate multiple predictions to form the final prediction. The main difference with bagging is that AdaBoost applies a weight to each of the training examples. In each iteration, the weights are individually updated to minimize the weighted error on the training set. For example, weights on those training examples incorrectly predicted in previous iteration increase, whereas the weights of the

correctly predicted training examples decrease. Therefore, AdaBoost tends to construct progressively more difficult learning problems in subsequent iterations. Once the training process has finished, the predictions are combined through a weighted majority vote (or sum) to produce the final prediction. So, the final classifier usually can achieve a high degree of accuracy in the test set.

### Stacking

Stacking regression combines multiple regression models via a meta-regressor, using out-of-fold prediction concept as shown in Figure 7 [55]. The method splits the data set into K folds, in which the k-1 folds are used to train the first level regressors in K successive rounds. In each round, the first level regressors are used to predict based on the remaining 1 subset. After that, the prediction results are used and stacked as input data to the second level regressors to form a final set of predictions [56]. In this study, SVR, BPNN and CRAT were used as regression models in the first level to get the prediction results, and LR was used as meta-regressor in the second level to combine and generate the final prediction results.

**Training Set**

**Training folds Validation folds**

**Training**

**C1**

**C2**

**Cm**

**Predictions**

**P1**

**P2**

**Pm**

**Regression**

**Repeat k times**

**models**

**Level-1 predictions in k-th iteration**

**All level-1 predictions**

**Final predication**

**Meta-Regressor**

**Pf**

Figure 7: Schematic diagram of Stacking model [56]

Table 5 shows the individual and ensemble models. The abbreviation for labelling models were adopted in a such a way that the letters Bag, Ada and Stack stand for the ensemble methods of Bagging, AdaBoost and Stacking, respectively. The letters LR, SVR, BPNN and CRAT stand for the base estimators. However, the Stack LR model refers to combining the base methods including SVR, BPNN, and CRAT using LR as a meta-regressor.

# Validation and Evaluation

## Cross-validation Method

Generally, the dataset is split into a training subset and a validation set while the properties of the original dataset are kept as much as possible to avoid misleading estimates. To minimize bias due to random data splitting,

the K-fold cross-validation is commonly used as it can yield optimal computational time and reliable variance [29, 57]. In this study, a ten-fold cross-validation approach was applied to assess model performance as shown in Figure 8. The dataset was split randomly into 10 equal-size subsets with a similar distribution. In each validation process, nine of the subsets were used for training and the rest for testing. The process was repeated 10 times [58]. The average accuracy after 10 times validation was reported as the model accuracy.

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

|  |
| --- |
| **1** |
| **2** |
| **3** |
| **4** |
| **5** |
| **6** |
| **7** |
| **8** |
| **9** |
| **10** |

**Testing data fold**

**Training data fold**

Figure 8: Ten-fold cross-validation approach

## Performance Evaluation

To show and validate the accuracy of the proposed ML models, three statistical indices namely mean absolute Error (MAE), root mean square error (RMSE), and the coefficient of determination *R*2 were used and expressed in equations 14, 15 and 16, respectively. The average deviation of the performance of an individual model or an ensemble model from a benchmark model in terms of three statistical measures (MAE, RMSE and *R*2) was calculated using equation (17).

* Mean absolute error (MAE).

*n*

Σ |

*MAE* = 1 *y*

*i*

*n i*

*i*=1

− *y′* | (14)

* Root mean square error (RMSE)

*n*

‚u Σ1

*RMSE* = , *n*

*i*=1

(*yi* − *y′* )2 (15)

*i*

* Coefficient of determination (*R*2)

2

*n i*=1

(*yi* − *y′* )2

*i*

Σ

(16)

* Deviation (*Dev*)

Σ

*R* = 1 −

*n*

*i*=1

(*yi*

− *y*)2

Where *yi* was the target output, *y′*

*i*

*Dev*(%) = *Pi* − *Pj* 100 (17)

*Pj*

∗

was the predicted output, *n* was the number of samples, *y* was the mean

of the target output. *Dev* indicated the statistical performance improvement compared with a benchmark model, *Pi* was the statistical performance (MAE, RMSE or *R*2) of an individual or ensemble method and *Pj* was the corresponding performance of a benchmark model, LR or an individual method used in the ensemble method as the base learner.

MAE statistics is a measure of errors between the predicted values (the estimated value of crack width of ECC after self-healing) with the estimated values (the observed value of crack width of ECC after self- healing in empirical data). RMSE statistics computes the square root of the average residual error between the predicted values and the target values. A lower value of MAE or RMSE indicates a better prediction performance of the model. *R*2 measures the strength of association between the predicted values and the target values, based on the proportion of total variation of outcomes. A greater value close to 1 represents a better prediction performance that commendably replicates the observed crack width of ECC after self- healing. Deviation statistics indicates the improvement of the prediction performance of an individual or an ensemble model from a benchmark model that can be the LR model or the individual model used as base learners in the corresponding ensemble model.

# Results and Discussion

Table 5 shows the ten-fold cross-validation results (MAE, RMSE, and *R*2) for both individual and ensemble models and their deviation with respect to the results of LR model. Generally, most of the proposed models were able to learn and predict empirical data with an acceptable degree of precision. Based on the results, the Stack LR model showed the best prediction performance as it has the highest *R*2 value and lowest MAE and RMSE values. Among the individual models, SVR performed the best in terms of MAE (4.296), but BPNN has the lowest RMSE value (6.515) and highest *R*2 of 0.899. For the single learning based ensemble methods, Bag CRAT gave the best performance in terms of MAE (4.093), and Bag BPNN performed better on RMSE value (6.341). In terms of *R*2, Bag CRAT and Bag BPNN models showed the same performance (0.901) and better than other ensemble methods except Stack LR.

Figure 9 compares the performance of all ML models in terms of MAE, RMSE and *R*2.

Table 5: Ten-fold cross-validation results of machine learning models on self-healing prediction for ECC

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Models | MAE | *Dev*(%) | RMSE | *Dev*(%) | *R*2 | *Dev*(%) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | LR | 5.012 - | 7.680 | - | 0.860 | - |
| Individual | BPNN | 4.329 -13.6 | 6.515 | -15.2 | 0.899 | 4.5 |
| models | CRAT | 4.305 -14.1 | 6.811 | -11.3 | 0.887 | 3.1 |
|  | SVR | 4.296 -14.3 | 6.826 | -11.1 | 0.883 | 2.7 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Ada LR | 4.784 -4.6 | 7.400 | -3.6 | 0.867 | 0.8 |
|  | Ada BPNN | 4.226 -15.7 | 6.435 | -16.2 | 0.900 | 4.7 |
|  | Ada CRAT | 4.207 -16.1 | 6.455 | -15.9 | 0.898 | 4.4 |
|  | Ada SVR | 4.145 -17.3 | 6.577 | -14.4 | 0.893 | 3.8 |
| Ensemble | Bag LR | 5.014 0.0 | 7.689 | 0.1 | 0.860 | 0.0 |
| models | Bag BPNN | 4.143 -17.3 | 6.341 | -17.4 | 0.901 | 4.8 |
|  | Bag CRAT | 4.093 -18.3 | 6.358 | -17.2 | 0.901 | 4.8 |
|  | Bag SVR | 4.302 -14.2 | 6.820 | -11.2 | 0.883 | 2.7 |
|  | Stack LR | 3.934 -21.5 | 6.118 | -20.3 | 0.904 | 5.1 |

Overall, all ensemble methods can noticeably reduce the error values and increase the prediction accuracy compared with LR, except for Bag LR. Among the models boosted by AdaBoost, Ada SVR performed the best with the lowest MAE value, whereas Ada BPNN performed the best on RMSE value showing the highest *R*2 value. In case of bagging, both Bag CRAT and Bag BPNN performed better in terms MAE, RMSE and *R*2 than those of the corresponding models boosted by AdaBoost. However, Bag LR showed a poor performance compared to LR on the MAE and RMSE values. This may result from several training examples repeatedly appearing in different replicated datasets to train multiple base regressors.

For a better comparison among the ensemble methods used, the performance results between the ensemble models and their corresponding individual (or benchmark) models are indicated in Table 6. The results indicate that most ensemble methods improved the performance of individual models. For example, the MAE and RMSE values of BPNN after bagging reduced by 4.3% and 2.7%, respectively, and with a higher value of *R*2 compared to those of the individual BPNN model. Among all the ensemble methods studied, stacking showed the best improvement on all performance measures.

Table 6: Performance deviation of ensemble models from benchmark models on self-healing of ECC

MAE RMSE *R*2 MAE RMSE *R*2

Benchmark Model *Dev*(%) Benchmark Model *Dev*(%)

LR Ada LR -4.6 -3.6 0.8 LR Bag LR 0.0 0.1 0.0

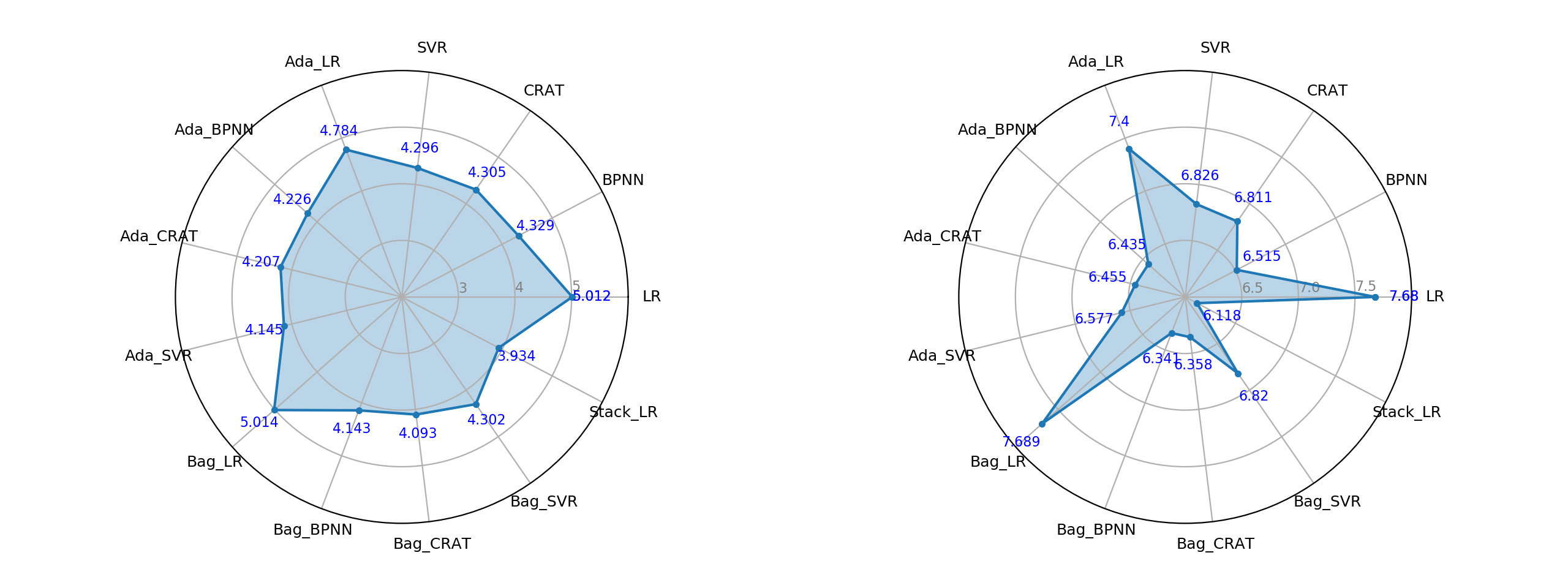
BPNN Ada BPNN -2.4 -1.2 0.1 BPNN Bag BPNN -4.3 -2.7 0.2

CRAT Ada CRAT -2.3 -5.2 1.2 CRAT Bag CRAT -4.9 -6.6 1.6

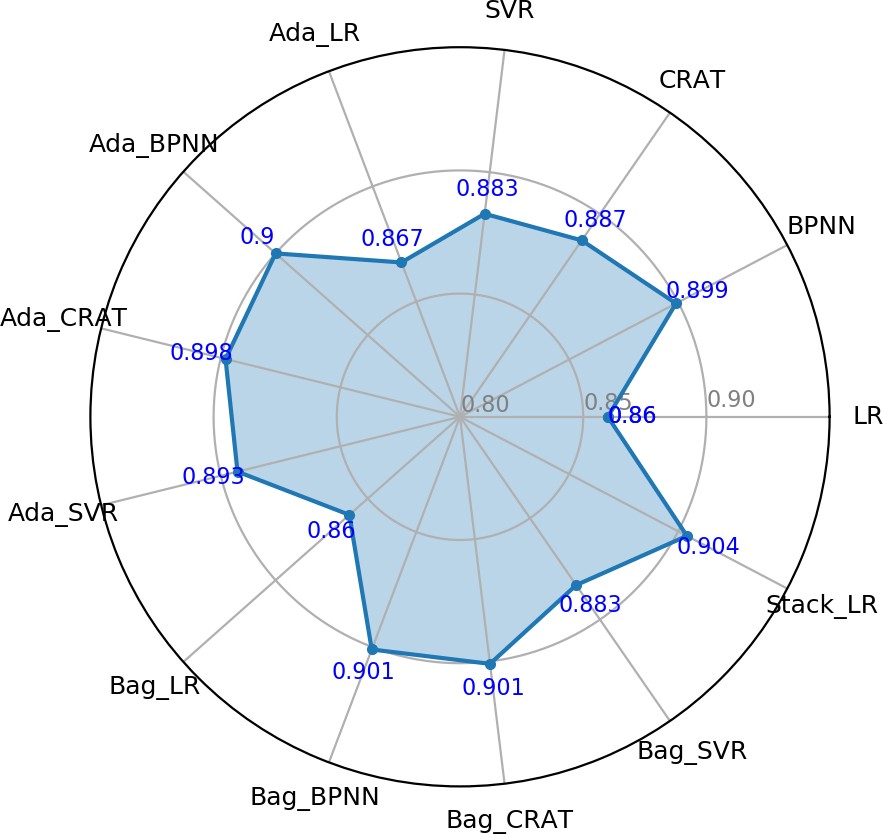
SVR Ada SVR -3.5 -3.6 1.1 SVR Bag SVR 0.1 -0.1 0.0

Ada LR Stack LR -17.8 -17.3 4.3 Bag LR Stack LR -21.5 -20.4 5.1

However, the results showed that the effectiveness of ensemble methods on individual models varied. For instance, bagging method enhanced the performance of BPNN and CRAT substantially, but not for both LR and SVR models. On the other hand, the AdaBoost method brought a considerable improvement for LR and SVR models. To improve the performance accuracy, researchers should employ different ensemble methods to compare their effectiveness on different ML models.



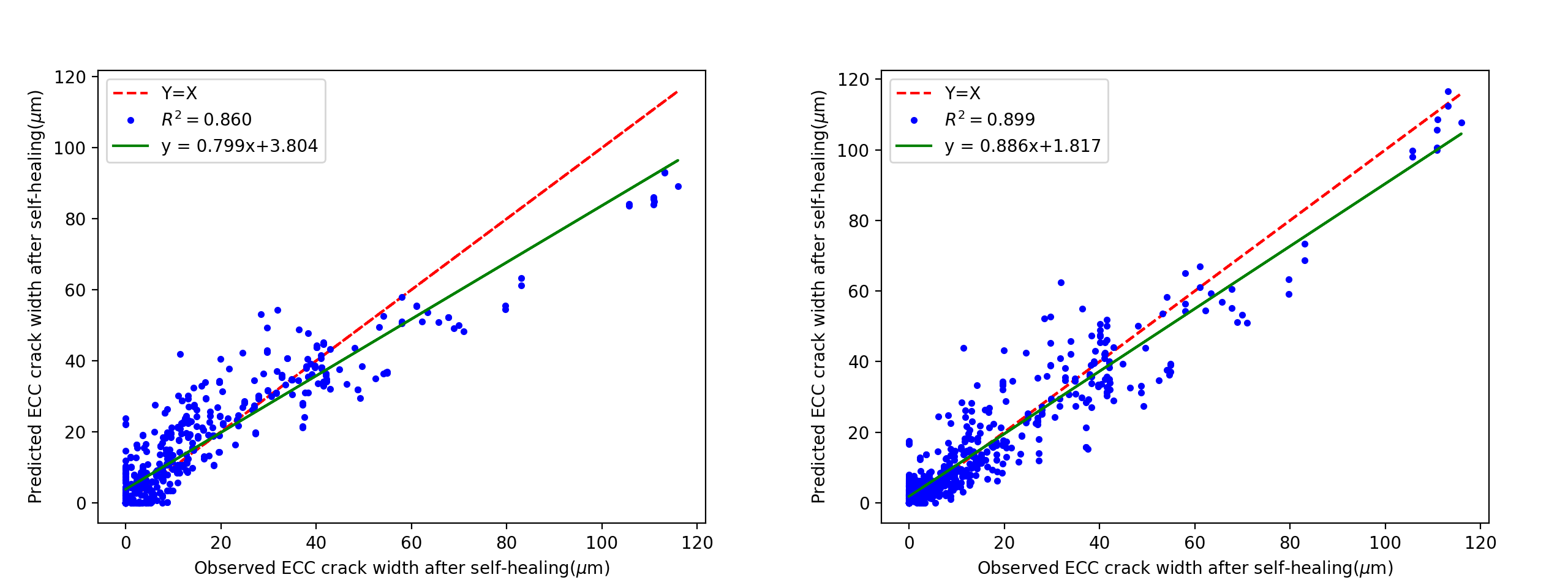
(a) MAE (b) RMSE



(c) *R*2

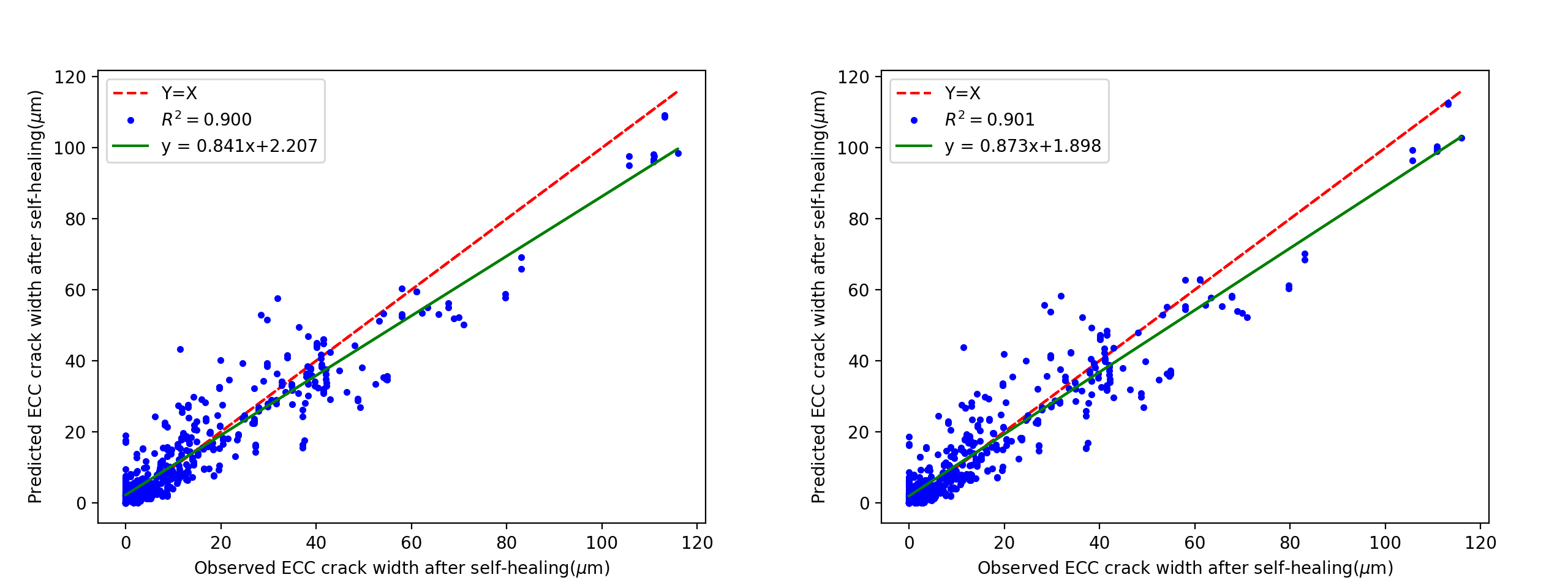
Figure 9: Average prediction performance of 10-fold cross-validation on all machine learning models for predicting self-healing ability of ECC

Figure 10 shows the observed results (crack widths after 10 W/D cycles) compared to those predicted by the proposed ML models. It should be noted that only the results for Stack LR, BPNN and its ensemble models are shown as they generally performed better than the rest. The LR model results are shown in Figure 10a for comparison.As it can be seen from Figure 10e, the Stack LR model showed the best performance as the predicted results are in good agreement with the observed results. The value of *R*2 (0.904) is also the highest among the others. It can be concluded that the Stack LR model is the best ML model for predicting self-healing of ECC. However, the crack width data observed from laboratory experiments was mainly distributed between 0 and 60*µm*, only a few data were over 100*µm*. Insufficient data may affect the prediction accuracy of ML models as a result of inadequate training of dataset.



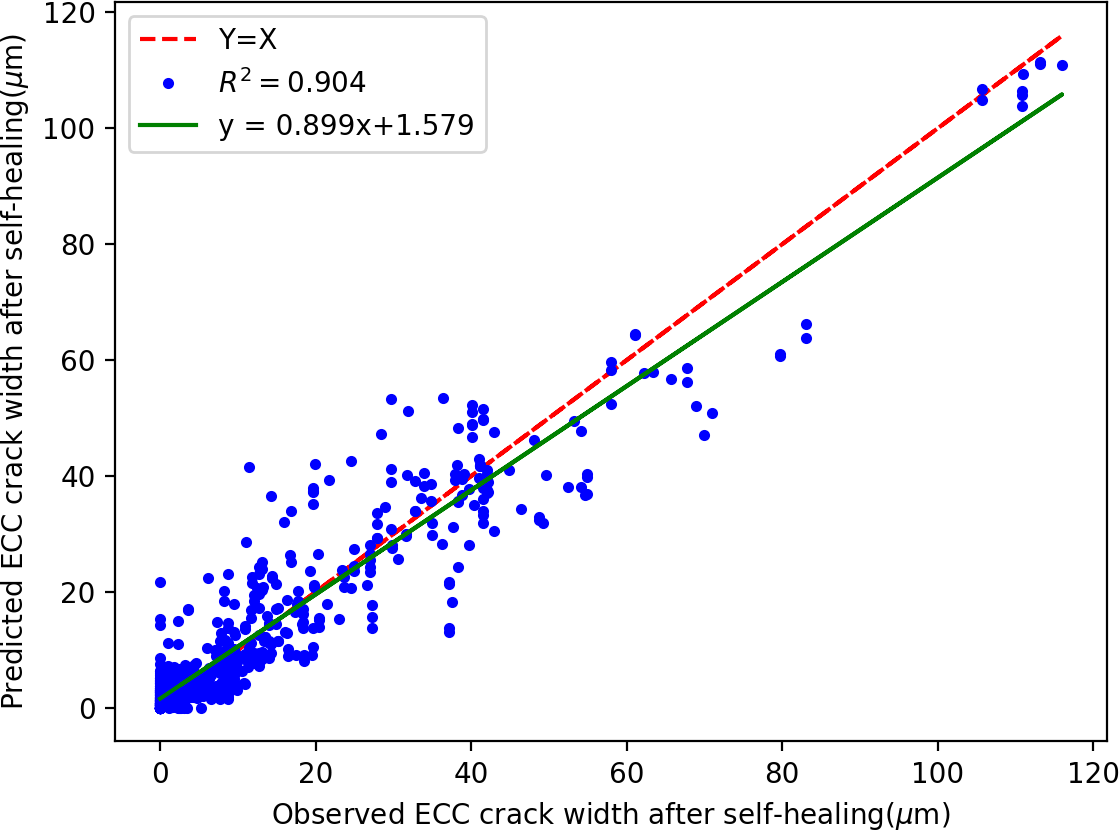
(a) Comparison of observed ECC crack width and predicted ECC(b) Comparison of observed ECC crack width and predicted ECC

crack width after self-healing by LR crack width after self-healing by BPNN



(c) Comparison of observed ECC crack width and predicted ECC(d) Comparison of observed ECC crack width and predicted ECC

crack width after self-healing by Ada BPNN crack width after self-healing by Bag BPNN



(e) Comparison of observed ECC crack width and predicted ECC crack width after self-healing by Stack LR

Figure 10: Comparison of observed ECC crack width and predicted ECC crack width after self-healing by individual and ensemble methods

# Conclusions

This study proposed and compared several individual and ensemble ML models to predict the self-healing ability of ECC. All the proposed ML models were trained, tested and validated based on the experimental results from nine ECC mixtures. Based on the results, the following conclusions can be drawn.

1. Among of the individual ML model studies, the BPNN model performed the best in terms of RMSE and *R*2.
2. All ensemble methods can generally improve the prediction accuracy of individual methods, however the improvement varies. It is found that Bagging method mainly enhanced the performance of BPNN and CRAT whereas AdaBoost method brought a considerable improvement for LR and SVR models.
3. Among all the ML models studies, the Stack LR model demonstrated great prediction on self-healing of ECC and performed the best on MAE, RMSE and *R*2 results.
4. The computational results indicate the individual and ensemble methods can be used to predict the self-healing ability of ECC. However, how to choose an appropriate base learner and ensemble method is critical. To improve the performance accuracy, researchers should employ different ensemble methods to compare their effectiveness on different ML models. The proposed individual and ensemble ML models can be used to predict the self-healing properties of ECC.
5. Future investigation and experimentation should be carried to extend the training dataset to include various crack width distributions and diverse influencing factors such as components, W/MC rate etc.. Also more research should be undertaken to optimize parameters in ML models and develop a hybrid model to improve the prediction accuracy.

# References

1. Diane Gardner, Robert Lark, Tony Jefferson, and Robert Davies. A survey on problems encountered in current concrete construction and the potential benefits of self-healing cementitious materials. *Case studies in construction materials*, 8:238–247, 2018.
2. E Cailleux and V Pollet. Investigations on the development of self-healing properties in protective coatings for concrete and repair mortars. In *Proceedings of the 2nd International Conference on Self- Healing Materials, Chicago, IL, USA*, volume 28, 2009.
3. Ahmed Ramadan Suleiman and Moncef L Nehdi. Modeling self-healing of concrete using hybrid genetic algorithm–artificial neural network. *Materials*, 10(2):135, 2017.
4. Waiching Tang, Omid Kardani, and Hongzhi Cui. Robust evaluation of self-healing efficiency in ce- mentitious materials–a review. *Construction and Building Materials*, 81:233–247, 2015.
5. Carola Edvardsen. Water permeability and autogenous healing of cracks in concrete. *Materials Journal*, 96(4):448–454, 1999.
6. Stefan Jacobsen, Jacques Marchand, and Hugues Hornain. Sem observations of the microstructure of frost deteriorated and self-healed concretes. *Cement and Concrete Research*, 25(8):1781–1790, 1995.
7. Hans-Wolf Reinhardt and Martin Jooss. Permeability and self-healing of cracked concrete as a function of temperature and crack width. *Cement and concrete research*, 33(7):981–985, 2003.
8. Mustafa S¸ ahmaran and ˙I O¨ zgu¨r Yaman. Influence of transverse crack width on reinforcement corrosion initiation and propagation in mortar beams. *Canadian Journal of Civil Engineering*, 35(3):236–245, 2008.
9. CA Clear. The effects of autogenous healing upon the leakage of water through cracks in concrete. Technical report, 1985.
10. Mustafa Sahmaran, Gurkan Yildirim, and Tahir K Erdem. Self-healing capability of cementitious composites incorporating different supplementary cementitious materials. *Cement and Concrete Com- posites*, 35(1):89–101, 2013.
11. Toshiro Kamada and Victor C Li. The effects of surface preparation on the fracture behavior of ecc/concrete repair system. *Cement and Concrete Composites*, 22(6):423–431, 2000.
12. Victor C Li and Tetsushi Kanda. Innovations forum: engineered cementitious composites for structural applications. *Journal of Materials in Civil Engineering*, 10(2):66–69, 1998.
13. Erdog˘an O¨ zbay, Mustafa Sˇahmaran, Mohamed Lachemi, and Hasan Erhan Yu¨cel. Self-healing of microcracks in high-volume fly-ash-incorporated engineered cementitious composites. *ACI Materials Journal*, 110(1), 2013.
14. Min Wu, Bjo¨rn Johannesson, and Mette Geiker. A review: Self-healing in cementitious materials and engineered cementitious composite as a self-healing material. *Construction and Building Materials*, 28(1):571–583, 2012.
15. Haoliang Huang, Guang Ye, and Denis Damidot. Characterization and quantification of self-healing behaviors of microcracks due to further hydration in cement paste. *Cement and Concrete Research*, 52: 71–81, 2013.
16. Ahmed R Suleiman, Andrew J Nelson, and Moncef L Nehdi. Visualization and quantification of crack self-healing in cement-based materials incorporating different minerals. *Cement and Concrete Com- posites*, 103:49–58, 2019.
17. J Zhou, S Qian, MG Sierra Beltran, G Ye, E Schlangen, and K van Breugel. Developing engineered cementitious composite with local materials. In *International conference on microstructure related durability of cementitious composites, Nanjing, China*, 2008.
18. Victor C Li and En-Hua Yang. Self healing in concrete materials. In *Self healing materials*, pages 161–193. Springer, 2007.
19. Zhigang Zhang, Shunzhi Qian, and Hui Ma. Investigating mechanical properties and self-healing be- havior of micro-cracked ecc with different volume of fly ash. *Construction and Building Materials*, 52: 17–23, 2014.
20. Gu¨rkan Yildirim, Mustafa Sahmaran, and Hemn Unis Ahmed. Influence of hydrated lime addition on the self-healing capability of high-volume fly ash incorporated cementitious composites. *Journal of Materials in Civil Engineering*, 27(6):04014187, 2014.
21. Y Yang, M Lepech, and Victor C Li. Self-healing of ecc under cyclic wetting and drying. 2005.
22. Mustafa Sahmaran, Mo Li, and Victor C Li. Transport properties of engineered cementitious compos- ites under chloride exposure. *ACI Materials Journal*, 104(6):604, 2007.
23. SZ Qian, J Zhou, and E Schlangen. Influence of curing condition and precracking time on the self- healing behavior of engineered cementitious composites. *Cement and concrete composites*, 32(9): 686–693, 2010.
24. Marai M Alshihri, Ahmed M Azmy, and Mousa S El-Bisy. Neural networks for predicting compressive strength of structural light weight concrete. *Construction and Building Materials*, 23(6):2214–2219, 2009.
25. Siamak Safarzadegan Gilan, Hamed Bahrami Jovein, and Ali Akbar Ramezanianpour. Hybrid sup- port vector regression–particle swarm optimization for prediction of compressive strength and rcpt of concretes containing metakaolin. *Construction and Building Materials*, 34:321–329, 2012.
26. Fei Yan, Zhibin Lin, Xingyu Wang, Fardad Azarmi, and Konstantin Sobolev. Evaluation and prediction of bond strength of gfrp-bar reinforced concrete using artificial neural network optimized with genetic algorithm. *Composite Structures*, 161:441–452, 2017.
27. Zaher Mundher Yaseen, Ravinesh C Deo, Ameer Hilal, Abbas M Abd, Laura Cornejo Bueno, Sancho Salcedo-Sanz, and Moncef L Nehdi. Predicting compressive strength of lightweight foamed concrete using extreme learning machine model. *Advances in Engineering Software*, 115:112–125, 2018.
28. Kezhen Yan and Caijun Shi. Prediction of elastic modulus of normal and high strength concrete by support vector machine. *Construction and Building Materials*, 24(8):1479–1485, 2010.
29. Jui-Sheng Chou, Chih-Fong Tsai, Anh-Duc Pham, and Yu-Hsin Lu. Machine learning in concrete strength simulations: Multi-nation data analytics. *Construction and Building Materials*, 73:771–780, 2014.
30. Uwe Reuter, Ahmad Sultan, and Dirk S Reischl. A comparative study of machine learning approaches for modeling concrete failure surfaces. *Advances in Engineering Software*, 116:67–79, 2018.
31. Jafar Sobhani, Meysam Najimi, Ali Reza Pourkhorshidi, and Tayebeh Parhizkar. Prediction of the compressive strength of no-slump concrete: A comparative study of regression, neural network and anfis models. *Construction and Building Materials*, 24(5):709–718, 2010.
32. Behzad Abounia Omran, Qian Chen, and Ruoyu Jin. Comparison of data mining techniques for pre- dicting compressive strength of environmentally friendly concrete. *Journal of Computing in Civil Engineering*, 30(6):04016029, 2016.
33. Luthfi Muhammad Mauludin and Chahmi Oucif. Modeling of self-healing concrete: A review. *Jour- nal of Applied and Computational Mechanics*, 5(Special Issue: Computational Methods for Material Failure):526–539, 2019.
34. Xiaoying Zhuang and Shuai Zhou. The prediction of self-healing capacity of bacteria-based concrete using machine learning approaches. *Computers, Materials & Continua 59 (2019), Nr. 1*, 2019.
35. Wenjun Wang, Nicolette G Moreau, Yingfang Yuan, Paul R Race, and Wei Pang. Towards machine learning approaches for predicting the self-healing efficiency of materials. *Computational Materials Science*, 168:180–187, 2019.
36. M Chaitanya, P Manikandan, V Prem Kumar, S Elavenil, and V Vasugi. Prediction of self-healing char- acteristics of ggbs admixed concrete using artificial neural network. In *Journal of Physics: Conference Series*, volume 1716, page 012019. IOP Publishing, 2020.
37. John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.
38. Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.
39. Vladimir Naumovich Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.
40. Xu Juncai, Ren Qingwen, and Shen Zhenzhong. Prediction of the strength of concrete radiation shield- ing based on ls-svm. *Annals of nuclear energy*, 85:296–300, 2015.
41. Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.
42. Yankun Li, Xueguang Shao, and Wensheng Cai. A consensus least squares support vector regression (ls-svr) for analysis of near-infrared spectra of plant samples. *Talanta*, 72(1):217–222, 2007.
43. SF Fang, MP Wang, WH Qi, and F Zheng. Hybrid genetic algorithms and support vector regression in forecasting atmospheric corrosion of metallic materials. *Computational Materials Science*, 44(2): 647–655, 2008.
44. P Yuvaraj, A Ramachandra Murthy, Nagesh R Iyer, SK Sekar, and Pijush Samui. Support vector regression based models to predict fracture characteristics of high strength and ultra high strength concrete beams. *Engineering fracture mechanics*, 98:29–43, 2013.
45. Alex J Smola and Bernhard Scho¨lkopf. A tutorial on support vector regression. *Statistics and comput- ing*, 14(3):199–222, 2004.
46. Abhijit Mukherjee and Sudip Nag Biswas. Artificial neural networks in prediction of mechanical behavior of concrete at high temperature. *Nuclear engineering and design*, 178(1):1–11, 1997.
47. Hosein Naderpour, Amir Hossein Rafiean, and Pouyan Fakharian. Compressive strength prediction of environmentally friendly concrete using artificial neural networks. *Journal of Building Engineering*, 16:213–219, 2018.
48. Leo Breiman. *Classification and regression trees*. 2017.
49. S Dan and P Colla. Cart: Tree-structured non-parametric data analysis. *Salford Systems, San Diego, CA*, 1995.
50. R Put, C Perrin, F Questier, D Coomans, DL Massart, and Y Vander Heyden. Classification and regression tree analysis for molecular descriptor selection and retention prediction in chromatographic quantitative structure–retention relationship studies. *Journal of Chromatography A*, 988(2):261–276, 2003.
51. Dimitrios Frosyniotis, Andreas Stafylopatis, and Aristidis Likas. A divide-and-conquer method for multi-net classifiers. *Pattern Analysis & Applications*, 6(1):32–40, 2003.
52. Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.
53. Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.
54. Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156. Citeseer, 1996.
55. Sebastian Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python’s scientific computing stack. *The Journal of Open Source Software*, 3(24), 2018. doi: 10.21105/joss.00638. URL [http://joss.theoj.org/papers/10.21105/joss.00638.](http://joss.theoj.org/papers/10.21105/joss.00638)
56. Joseph Sill, Ga´bor Taka´cs, Lester Mackey, and David Lin. Feature-weighted linear stacking. *arXiv preprint arXiv:0911.0460*, 2009.
57. Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selec- tion. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.
58. Ofer Dor and Yaoqi Zhou. Achieving 80% ten-fold cross-validated accuracy for secondary structure prediction by large-scale training. *Proteins: Structure, Function, and Bioinformatics*, 66(4):838–845, 2007.