Acta Materialia 169 (2019) 225–236

Contents lists available at ScienceDirect

Acta Materialia

journal homepage: www.elsevier.com/locate/actamat

Full length article

Machine-learning phase prediction of high-entropy alloys

Wenjiang Huang, Pedro Martin, Houlong L. Zhuang*

Artificial Intelligence for Alloys Laboratory, School for Engineering of Matter Transport and Energy, Arizona State University, Tempe, AZ, 85287, USA

ARTICLE INFO

Article history:
Received 26 October 2018
Received in revised form
7 March 2019
Accepted 10 March 2019
Available online 15 March 2019

Keywords:
High-entropy alloys
Phase selection
Machine learning

ABSTRACT

High-entropy alloys (HEAs) have been receiving intensive attention due to their unusual properties that largely depend on the selection among three phases: solid solution (SS), intermetallic compound (IM), and mixed SS and IM (SS + IM). Accurate phase prediction is therefore crucial for guiding the selection of a combination of elements to form a HEA with desirable properties. It is widely accepted that the phase selection is correlated with elemental features such as valence electron concentration and the formation enthalpy, leading to a set of parametric phase-selection rules [1]. Previous studies on predicting the phase selection employed density functional theory (DFT) calculations to obtain some correlated parameters. But DFT calculations are time consuming and exhibit uncertainties in terms of treating the *d* orbitals of transition-metal atoms that are often components of HEAs. Here we employ machine learning (ML) algorithms to efficiently explore phase selection rules using a comprehensive experimental dataset consisting of 401 different HEAs including 174 SS, 54 IM, and 173 SS + IM phases. We adopt three different ML algorithms: K-nearest neighbours (KNN), support vector machine (SVM), and artificial neural network (ANN). To avoid overfitting, we divide the whole dataset into four nearly equal portions to perform a cross validation. For the classification of the three phases at the same time, the testing accuracy values from the KNN, SVM and ANN calculations achieve 68.6%, 64.3% and 74.3%, respectively. We then focus on the classification of two of the three phases using SVM and ANN. We find that the testing accuracy values using ANN in classifying the SS and IM phases, the SS + IM and IM phases, and the SS and SS + IM phases, are 86.7%, 94.3%, and 78.9%, respectively, which are higher than the corresponding testing accuracy values using SVM. As such, the trained ANN model performs the best among the three ML algorithms and is useful for predicting the phases of new HEAs. Our work provides an alternative route of computational design of HEAs, which is also applicable to accelerate the discovery of other metal alloys for modern engineering applications.

1. Introduction

Metal alloys have been playing crucial roles in the world history since the Bronze Age. Modern metal alloys find a number of applications in automotive and aerospace engineering industries [2,3], and as nuclear and bio-medical materials [4,5]. In spite of forming from a variety of elements, different (traditional) metal alloys have a commonality—they contain only one or two principal elements. For example, for a ternary metal alloy, its composition locates at a corner of a phase diagram represented by a triangle.

High entropy alloys (HEAs) are a novel type of metal alloys—generally consisting of more than five elements with equal or nearly equal concentrations [6]. The compositions of HEAs therefore locate at/near the center of the composition space. HEAs have attracted immense interest through the pioneering work by Cantor and Yeh et al. [7–9], due to the reported excellent mechanical and chemical properties that are absent in traditional metal alloys, such as high hardness and ductility [10–14], high-temperature strength [15,16], and antioxidant capacity and wear resistance [17–19], and HEAs are expected to be applied in a wider range of fields than traditional alloys.

Whether a HEA exhibits excellent properties depends on which phase the HEA adopts among three possible ones: solid solution (SS), intermetallic compound (IM), and mixed SS and IM (SS + IM). It is often because of the presence of the SS phase that leads HEAs to exhibit the above-mentioned desirable properties. By contrast, the presence of a brittle IM phase degrades the ductility [20]. Accurate prediction of the resulting phase for a given combination of constituent elements is therefore crucial to the development and

* Corresponding author..
E-mail address: zhuanghl@asu.edu (H.L. Zhuang).

https://doi.org/10.1016/j.actamat.2019.03.012
1359-6454/

applications of new HEAs. Density functional theory (DFT) calculations are a useful technique to simulate HEAs [21]. Huhn and Widom applied DFT calculations to predict the phase transformation in HEAs [22]. But DFT calculations are impractical to deal with large simulation cells accounting for the phases in a HEA, due to intense computational cost and uncertainties in treating the *d* orbitals of transition-metal atoms that are often components of HEAs [23].

Parametric approaches have been commonly used to predict the phase selection of HEAs. For instance, Zhang et al. proposed that the phase selection of HEAs is determined by parameters such as the atomic size difference (δ), mixing entropy (ΔS_{mix}) and mixing enthalpy (ΔH_{mix}) [24]. Unlike the parametric approaches, machine learning (ML) provides a state-of-the-art tool that offers insight from given data of relevant properties of alloys without the need of explicit programming. Fast and accurate ML models have been increasingly used for material discovery in recent years [25–28]. For example, Lin et al. successfully predicted the performance of a steel under an elevated temperature flow via ML [29].

How to select the most relevant input features is an important question to address in ML calculations. Regarding the issue of phase selection in HEAs, it is widely accepted that the phase selection correlates with the properties of individual species forming a HEA such as the valence electron concentration (VEC) and electronegativity difference $\Delta\chi$ [1]. This correlation is in keeping with the Hume-Rothery rules, established nearly a century ago [30]. These rules state that several elemental properties such as atomic radius and VEC correlate with the crystal structures and stability of metal alloys, providing an invaluable tool for designing a new alloy. But these rules originated from observing and summarizing the pattern of a small set of experimental data of binary alloys, which expectedly limit the applicability of the rules to a broader range of metal alloys. For instance, similar atomic radii and electronegativities of constituent alloying elements were regarded as a desirable condition to generate the SS phase, but it is incapable of forming this phase in HEA [31]. A number of other factors like the atomic polarizability and the formation enthalpy have been proposed to improve the predictive power of the rules and the list of such factors keeps growing longer. How these additional factors are independent from each other or from those already considered in the existing rules, however, remains unclear due to the lack of in-depth data analysis and to a dearth of experimental data. Consequently, all the proposed factors exhibit restraints in guiding the design of novel HEAs.

In this work, with the premise of ensuring independence of parameters, we apply three common ML algorithms: (1) K-nearest neighbours (KNN), (2) support vector machines (SVM), and (3) artificial neural network (ANN), to predict the phase selection of the SS, IM, and SS + IM phases in HEAs. Our overarching goal is to screen out the most suitable ML model for the future design and discovery of new HEAs.

2. Computational methods

2.1. Data collection and analysis

We use an experimental dataset summarized by Miracle and Senkov in their review article of HEAs [32]. The given dataset contains 648 entries, but we remove a portion of them because there are multiple entries corresponding to the same composition. This results in a new data set applied in our ML models, composed of 401 HEAs consisting of 174 SS, 54 IM, and 173 SS + IM phases. The following equations are employed to obtain the input numerical values of the five features [24,33–35]:

$$\mathrm{VEC} = \sum_{i=1}^{n} c_i \mathrm{VEC}_i, \tag{1}$$

$$\Delta\chi = \sqrt{\sum_{i=1}^{n} c_i(\chi_i - \overline{\chi})^2}, \tag{2}$$

$$\delta = 100 \times \sqrt{\sum_{i=1}^{n} c_i(1 - r_i/\overline{r})^2}, \tag{3}$$

$$\Delta S_{\mathrm{mix}} = -R\sum_{i=1}^{n} c_i \ln c_i, \tag{4}$$

$$\Delta H_{\mathrm{mix}} = \sum_{i=1, i<j}^{n} 4H_{ij}c_i c_j, \tag{5}$$

where VEC_i and c_i refer to the valence electron and atomic concentrations of the *i*th element, respectively. *n* is the total number of species in a HEA. χ_i and r_i denote the Pauling electronegativity and radius of *i*th element, respectively [36]. The averaged Pauling electronegativity $\overline{\chi}$ and averaged atomic radius $\overline{r}$ are calculated as $\overline{\chi} = \sum_{i=1}^{n} c_i\chi_i$ and $\overline{r} = \sum_{i=1}^{n} c_i r_i$, respectively. The VEC_i, χ_i, and r_i are also taken from Ref. [32]. *R* in Eq. (4) denotes the gas constant. H_{ij} is the enthalpy of atomic pairs of the *i*th and *j*th elements computed by Takeuchi and Inoue with the Miedema method [37].

Before transferring the 401 data of HEAs to the ML process, we perform a statistical data analysis on these data as shown in Fig. 1. The diagonal panels show that the phases of the HEAs depend on more than one input features. In other words, one cannot draw a clear boundary using a single feature to distinguish the phases. We also compute the Pearson correlation coefficients *P* between each pair of features and there is no strong correlation ($P \sim 1.0$) between any two features. As a result, we suggest that all of the five features should be involved in the ML process.

2.2. KNN and SVM

Before introducing these ML algorithms, we describe a problem, *i.e.*, overfitting that a ML calculation often encounters. Overfitting occurs when a ML model generates results that are too close to the training data set, and may therefore fail to be practical to testing data or predict future observation. This is because the model learns both the detail and noise of the training data set, negatively affecting the testing accuracy on testing stage.

To remedy the problem of overfitting, we apply a typical technique—*n*-fold cross validation. We use a four-fold cross validation, where we divide our data set into four nearly equal portions. In each portion, the number of each phase is almost the same. For example, each portion has about 43 (174/4) SS phases. We also assign IDs (P1, P2, P3, and P4) to the four portions of data. Each portion is used once as the testing data. When this portion of data represents the testing data, the other three portions are treated as the training data. This cross validation process allows to train and test our data four times on different subsets of the full data set and create a reliable model for improving the testing accuracy [38]. The final reported testing accuracy value is the average of the four testing accuracy results.

K-nearest neighbours (KNN) is a common supervised learning, non-parametric method used for solving both regression and classification problems [39–41]. For a classification problem, all the

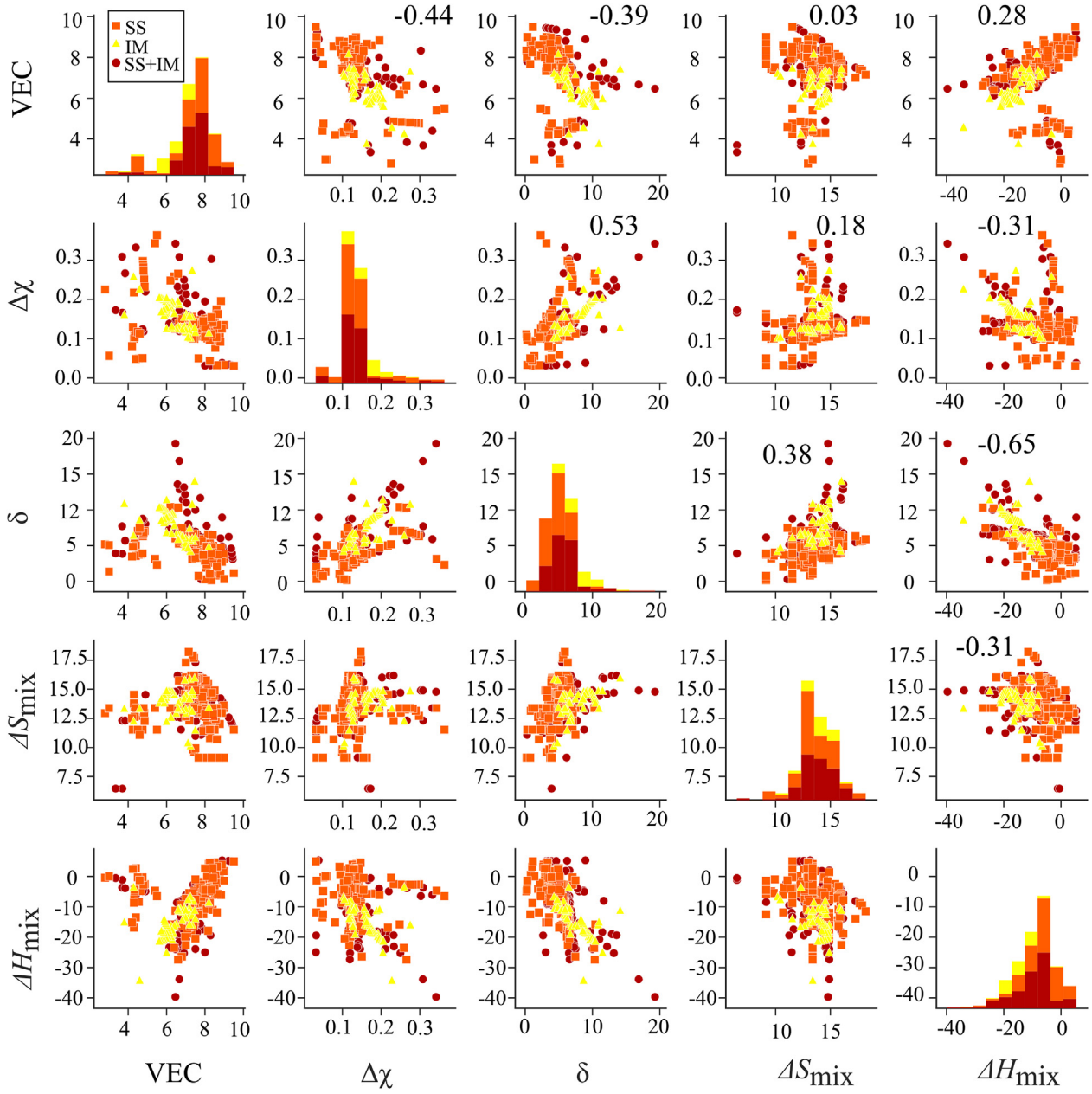

Fig. 1. Diagonal panels: Histograms showing the distributions of the SS, IM, and SS + IM phases at different values of one of the five features. Off-diagonal panels: Scatter plots showing the distributions of the three phases as a function of a pair of the five features. The numbers at the top right of the upper triangular panels are the calculated Pearson correlation coefficients between each pair of features.

training data represented by vectors in a n-dimensional space ($n = 5$ here due to the five input features) are assigned specific labels. During the testing stage, a testing data (vector) is added into the space and the distance between this testing vector and training vectors are computed using Euclidean distance algorithm [42],

$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}. \quad (6)$$

where $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ represents the testing vector and $\mathbf{q} = (q_1, q_2, \ldots q_n)$ is one of the KNN training vectors.

With a pre-set k value, the k-nearest training samples to a testing data are selected, and the dominant label among these training vectors is assigned to the testing data [43]. For example, if $k = 1$, the testing vector has the same category as its nearest-neighbor training vector. If $k = 3$, the first three closest training samples to the query (testing) sample are chosen to define its label, and the query sample is assigned to the same class which appears more than twice in the three chosen samples. Therefore, for a binary classification problem, k is empirically set to odd numbers (such as 1, 3, 5, …) and for a ternary classification problem, k is often set to one or some even numbers (such as 4, 6, 8 …). There are six types of KNN implemented in the Matlab ML package [44,45]. We find only two of them—Fine and Weighted KNNs—result in a training accuracy of nearly 100%. We therefore select these two KNNs to perform a four-fold cross validation to train and test the data set. We adjust the k values from 1 to 10 to explore how they affect the testing accuracy values.

Support vector machines (SVMs) are typical supervised learning methods, which have been reported to possess an advantage of adopting theoretical concepts from computational learning theory to achieve a good performance in various problems, especially binary classification [46]. The gist of SVMs contains four main components: (1) SV estimation function, (2) learning theory, (3) optimal hyper-plane algorithm, and (4) kernel function [47,48].

The training set—N-dimensional (R^N) pattern x_i and its label y_i, form a estimation function $f(x_i) = y_i$, which tends to conform the distribution of training data [49]. For new samples (testing set), the estimation function generated from the training data is used to calculate the same underlying probability distribution $P(\mathbf{x}, \mathbf{y})$ and then to assign labels to new samples, which will be assigned to the class having a higher probability. But the learning theory proposes that, if there is no restraint to the estimation function, even though there exists a function behaving well in the training data, the function will not generalize unknown data (testing data) [50]. This is because ML will not perform a task that is not assigned to. A ML algorithm only needs to generate the ideal result for training set and its mission is accomplished, giving rise to the above-mentioned overfitting problem. We therefore must not let the training data spontaneously form the estimation function, but we use a kernel function as the restriction [51]. The nonlinear training data in an input space can be mapped into a feature space via an algorithm Θ, which constructs a hyper-plane Γ (usually the dimension of Γ is R^{N-1}), with a maximum margin between two classes (see Fig. 2) [52]. $\Theta(x_i)$ then substitutes each training pattern x_i and performs the optimal hyper-plane algorithm in the feature space. A kernel function can be defined as,

$$k(\mathbf{x}, \mathbf{y}) = (\Theta(\mathbf{x}) \cdot \Theta(\mathbf{y})). \tag{7}$$

We learn from Eq. (7) that the higher dimension of the feature space corresponds to the more complex kernel function. Selecting different kernel functions leads to different algorithms such as polynomial, radial basis function, and Gaussian [53].

The Gaussian kernel is a typical kernel method often generating a fine and accurate tradeoff between fitting and smoothing the data. We therefore adopt this kernel in this work. In one-dimensional, two-dimensional and N-dimensional spaces, the Gaussian kernel function can be defined as [54]

$$G_{1D}(x, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2} \right), \tag{8}$$

$$G_{2D}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2} \right), \tag{9}$$

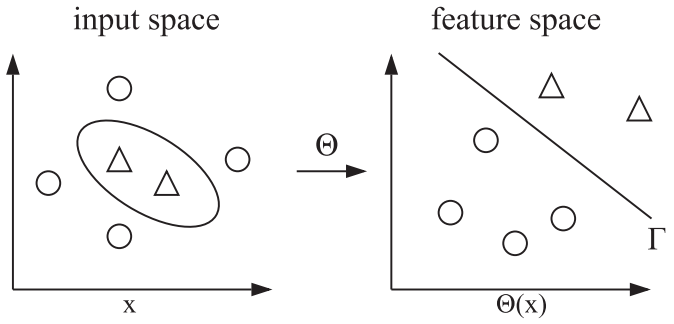

Fig. 2. A diagram showing the nonlinearity of a classification problem using support vector machines is transformed to a linear problem through a kernel function Θ, leading to a plane Γ called a hyper-plane in the feature space.

and

$$G_{ND}(\mathbf{x}, \sigma) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^N} \exp\left(-\frac{\left|\left|\mathbf{x}\right|\right|^2}{2\sigma^2} \right), \tag{10}$$

respectively, where σ, the standard deviation, determines the width of the Gaussian function and σ^2 is called the variance.

2.3. ANN

Artificial neural network (ANN) is constructed by a series of interconnected artificial neurons to simulate the function of a biological learning system like a human brain [55]. In the ANN family, there are several different models, each of which has its own characteristics and applications. These models can be defined as supervised or unsupervised learning according to whether target labels are provided [56]. Similar to the other ML algorithms, ANN possesses an autonomous learning ability and is capable of recognizing the underlying data pattern through its own statistical models. In this work, we employ two main types of ANN models: the unsupervised self-organizing maps (SOMs) and the supervised multi-layer feed-forward neural network (MLFFNN).

2.3.1. SOM

The Kohonen SOM method is a typical unsupervised machine learning neural network method, where the network learns to distinguish groups without the need of knowing their labels [44]. The SOM follows the topology competitive principle among neurons [57,58]. In such a network, neurons are distributed in a grid to form a so-called map in a n-dimensional space. Here, n is 5 due to the five input features. Fig. 3(a) illustrates an example of the SOM model. The Euclidean distance between an input vector, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and a weight vector $\mathbf{w} = (w_{j1} w_{j2}, \ldots w_{jn})$ [58,59], for each neuron $j = (1, 2, \ldots, N$; N: total number of neurons in the network) can be written as

$$d_j(x) = \sqrt{\sum_{i=1}^{N} (x_i - w_{ji})^2}. \tag{11}$$

The neuron in the map that is closest to the input vector is called the "winner" [60]. In the learning process, not only the "winner" tends to move toward the input vector $\mathbf{x}$, but also its neighbours all move closer to $\mathbf{x}$, although not as much as the "winner" moves, resulting in neighboring neurons that have similar weight vectors. In this way, similar input vectors can be clustered by a group of neurons with similar weight vectors and the clustering of neurons approximates the categorical distribution of the input data solving the classification problem (see Fig. 3(b)). During the moving process, the learning weights of all the neurons are updated by the following equation [61]:

$$w_{ji}^{t+1} = w_{ji}^{t} + \eta h(j, k)\left(x_i^t - w_{ji}^t\right), 0 \le i \le n, \tag{12}$$

where η is the learning rate and t represents the current epoch of iterations [62]. k denotes the "winner" and $h(j, k)$ is a function dependent on the distance between neuron j and the "winner" k. $h(j, k)$ is equal to one if $j = k$, and it decreases as neuron j is far away from the "winner".

Fig. 3(b) assumes an ideal outcome after convergence. Three groups (SS, IM, SS + IM) are clustered separately due to different properties and the map uses three different groups organized by neurons to enclose the target categories. The residual neurons denoted by 1, 2, …, and 9 are far away from the three groups

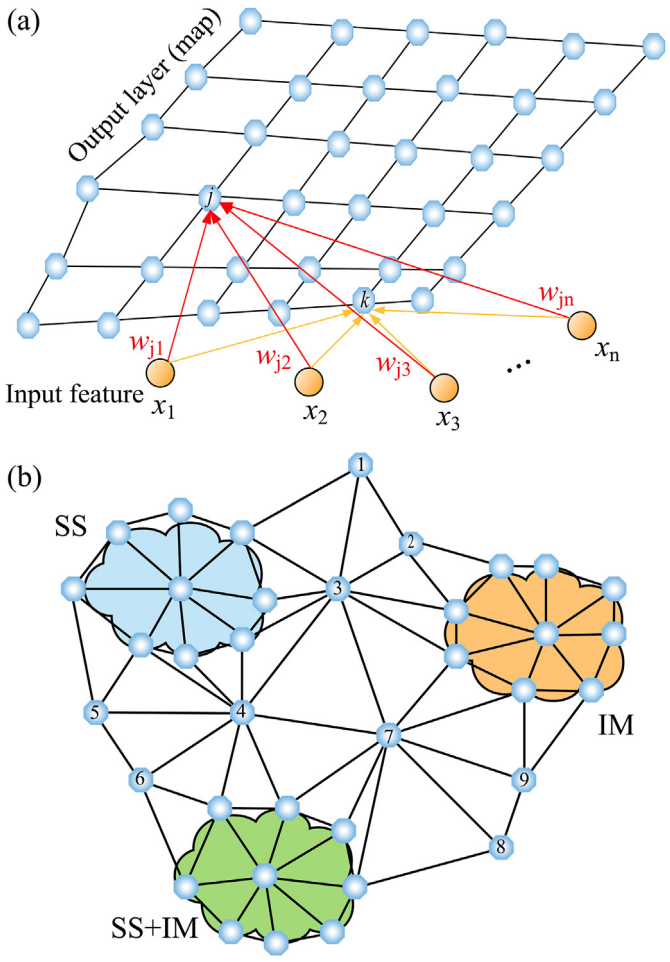

Fig. 3. An example of the self-organizing map model adopted from Ref. [63]. (a) The orange circles represent input features. The octagons denote neurons that are organized into a map, representing an unsupervised learning neural network. (b) An ideal situation that three groups have clear boundary in space and the "map" tends to use groups of neurons to embrace each category respectively, while residual neurons form the isolation band among three groups. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

forming boundaries among the groups.

Note that because SOM is an unsupervised ML method, we can only obtain the information of clustering rather than the specific category each cluster belongs to. The obtained n-dimensional weight vectors can be visualized through the projection onto a two-dimensional space. In this work, we use the SOM method as implemented in MATLAB with a 10 × 10 grid of neurons to obtain two-dimensional weight planes.

2.3.2. MLFFNN

Similar to any other ML algorithm, the general routine of ANN-based ML comprises two steps: training and testing. For a MLFFNN framework [64], the neurons are forced to connect in the forward direction. Namely, the transmission of signals is consecutive and there is no recurrent or backward connection between neurons.

Architecture of the neural network. The first step to construct the MLFFNN is to define the number of the hidden layers. Generally, increasing the number of hidden layers is associated with longer computational time and larger storage of training parameters. We find that the neuron network architecture with only two hidden layers is not capable of solving the complex ternary classification problem, i.e., the resulting testing accuracy is expectedly low. We also test the MLFFNN models with four or more hidden layers, and the accuracy is nearly the same but associated with significant more computational efforts. Considering the trade-off between testing accuracy and computational cost, we focus on using three hidden layers in our work. Fig. 4 illustrates the MLFFNN architecture employed in this work. Therefore, the architecture consists of an input layer, three hidden layers and one output layer. In the input layer, all information is received from outside and no more processing is required in this layer. The input vector of features undergoes a linear operation by a matrix called the weight matrix. The resulting new vector together with a bias vector leads to another vector to be processed by an activation function at each neuron of the first hidden layer. This process continues until the activation of the last hidden layer to create the output layer. The hidden layers, providing the capability of learning the nonlinearity of a problem and of solving it, are therefore the key components of the whole network. High training and testing accuracy can be achieved by modifying, for example, training functions, activation functions, and learning rates.

Activation and error functions and learning rate. We use the Sigmoid function [65] as the activation function to transform the input variables to values ranging from 0 to 1. We use the mean-squared error (MSE) as a criterion to measure the performance of training a neural network [66]. During training, the network compares the actual and expected outputs using a MSE algorithm after each epoch. The network then obtains the feedback from the calculated MSE, thus components of the weight matrix and bias vector are adjusted to decrease the MSE until it converges. This method is called the back propagation algorithm [67]. A suitable choice of the learning rate can not only improve the efficiency of training but also ensure reaching the minimum error during training. We adopt 0.01 as the learning rate through trial and error.

Number of neurons. Choosing an optimal number of neurons for each hidden layer is critical to the performance of a neural network. Too few neurons and too many neurons lead to underfitting and overfitting issues, respectively. We therefore propose a cyclical approach to determine the number of neurons n_1, n_2, and n_3 in the first, second, and third hidden layers, respectively. We define one combination of n_1, n_2, and n_3 as a cycle. For the ternary problem of directly classifying three phases and the binary classification of the SS and SS + IM phases, we set a higher upper limit of 50 for n_1, n_2, or n_3 and their initial value is set to 10. This leads to 729 (9^3) cycles

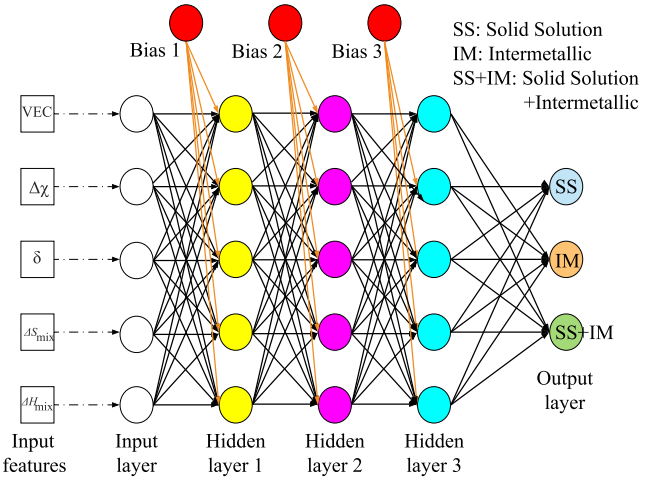

Fig. 4. An architecture of a multi-layer feed-forward artificial neural network consisting of an input layer, three hidden layers, and one output layer. The five input features and three bias nodes are also sketched.

in total. A lower limit of 30 is used for the relatively straightforward binary problems of classifying the SS and IM phases, and of classifying the SS + IM and IM phases; the initial value of n_1, n_2, and n_3 is also set to a smaller number (5), resulting in 216 (6^3) cycles. Note that choosing the initial values and upper and lower limits is purely empirical as suggested in the literature using multiples of the number of input features as the number of neurons in hidden layers [68]. We shall see that our selected ranges (10~50 and 5~30) are sufficiently wide to lead to a suitable combination of n_1, n_2, and n_3.

Overfitting and cross validation. The overfitting problem still occurs if too many iterations (epochs) are used associated with low MSE during the training process in our work [69,70]. In addition to the four-cross validation method we applied, we therefore constantly change the number of training epochs attempting to avoid overfitting. The final determined number of epochs is 1000 by trial and error [38].

3. Results and discussion

3.1. KNN and SVM

Table 1 reports the testing accuracy resulted from using Fine and Weighted KNNs. As can be seen, neither KNN can lead to the testing accuracy higher than 70%, independent of the choices of the k values. This low testing accuracy is due to a common drawback of applying KNNs to classification problems—when the class distribution is imbalanced, the more frequent categories in the training data tend to dominate the prediction of the classes of the testing data. This problem becomes more serious as the k values increase. As a result, Table 1 also shows that the KNN accuracy is lower when the k values are larger. In short, our calculations indicate the KNN methods are not the best ML algorithm for classifying the phases in HEAs due to the imbalanced distribution of 54 IM phases in contrast to 174 SS and 173 SS + IM phases.

SVM is widely accepted for performing well in binary classification problems. Indeed, our work confirms that it is not suitable for the ternary classification (SS, IM, and SS + IM), whose testing accuracy cannot exceed 64.3%. We therefore apply SVM to binary classifications (*i.e.*, classifying SS and IM, SS + IM and IM, as well as SS and SS + IM). As we mentioned before, Gaussian kernel function as training function is applied to SVMs. In the MATLAB software, there are Fine Gaussian, Medium Gaussian and Coarse Gaussian SVMs, and the differences among them lie in the flexibility of response function and kernel scale (the kernel scale is for scaling all features to comparable ranges; the lower kernel scale corresponds to the better performance). Fine Gaussian SVM bears the lowest kernel scale and the highest flexibility to response function and allows the rapid variation during training so that Fine Gaussian SVM performs the best among all the three SVMs. After four-fold cross validation, we obtain the average testing accuracy values for the SS and IM phases, the SS + IM and IM phases, and the SS and SS + IM phases, which are 86.3%, 91.2%, and 73.2%, respectively. The results seem to be a significant improvement compared to previous one. Note that the accuracy improved by using SVM is not necessarily because SVM is handing the imbalanced data approximately. Instead, SVM has been shown to be more powerful than KNN [71]. In other words, the imbalance in data remains unaddressed because of the limit in the available data.

Fig. 5 shows a scatter plot based on the VEC-δ coordinate system displaying the distribution and wrong labels of the SS and SS + IM phases. The result is consistent with the subplot in Fig. 1 using the same coordinate system.

3.2. ANN

3.2.1. SOM

The most valuable role of SOM method is reflected on using a two-dimensional map (see Fig. 6(a)) to visualize the distribution of 401 alloys in a five-dimensional space. The colors in the region containing the red line indicate the distance between neurons in the five-dimensional space of training data of the alloys. For example, a darker color represents a larger distance than a lighter one. The lower-right region marked using a black curve manually is approximately clustered as a group. By observing the so-called Hits map as illustrated in Fig. 6(b) (the black curve encloses the corresponding area in Fig. 6(a)), one is able to learn how many samples are associated with each neuron. We find this group contains nearly 50 members, corresponding to the number of the IM phases. Thus, the distribution of the IM phases in the five-dimensional space is relatively concentrated with a clear boundary. Furthermore, Fig. 7 shows the contribution of each variable for deciding the position of neurons in the five-dimensional space. From the perspective of the coordinate system, each graph represents the value of each neuron under certain coordinate axis. For instance, the most upper-left neuron has larger distances for the $\Delta\chi$ and δ variables and smaller distances for the other three variables. Therefore, if the weight planes of two variables are similar, these two variables are highly correlated. In our results, we can see that the weight plane for each variable is different from the others so that the five input features are independent.

Although the SOM method cannot provide a very precise clustering result, it shows the approximate distribution of 401 alloys in the five-dimensional space and the correlations between each variables. From the results, we can identify that the core of this

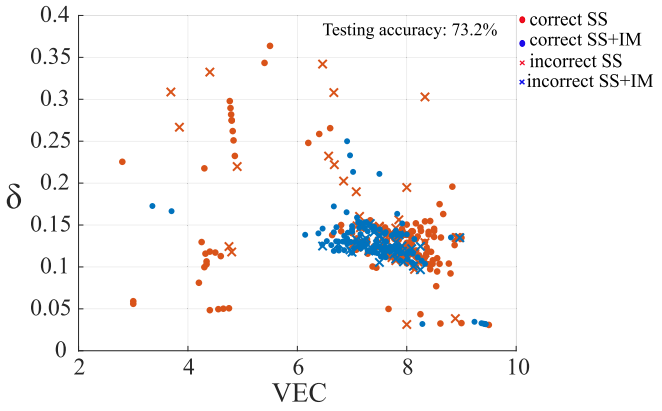

Fig. 5. A scatter plot showing the distribution of two classes when they are projected into VEC-δ coordinate system. There are other 24 figures besides this one similar to the 25 subfigures shown in Fig. 1. However, the plot here also indicates which samples are misclassified and the testing accuracy.

Table 1
Testing accuracy resulting from the calculations of two types of KNN with different k values ranging from 1 to 10.

	1	2	3	4	5	6	7	8	9	10
Fine KNN	65.8%	66.3%	65.6%	62.3%	64.8%	59.9%	61.6%	61.1%	61.6%	61.3%
Weighted KNN	65.8%	65.8%	68.6%	68.3%	68.3%	66.8%	67.3%	66.6%	67.6%	67.1%

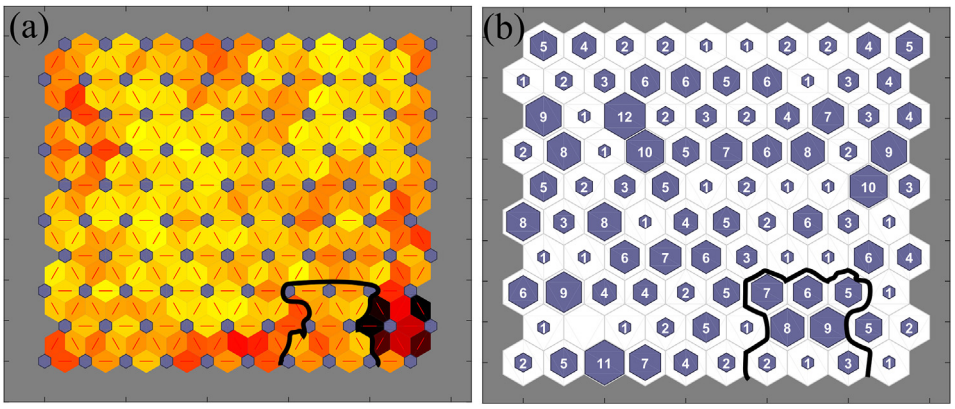

Fig. 6. (a) Neighboring weight distances in a self-organizing map with 10 × 10 neurons. The blue hexagons represent the neurons, and the red lines connect neighboring neurons. The color intensities in the diamond-shaped areas containing the red lines indicate the distances between two neurons. The darker color corresponds to the larger distance. (b) A Hits map corresponding to the neighboring weight distances in (a), showing how many samples are associated with each neuron. For example, the neuron at the upper-left corner denoted by the number 5 shows that there are five data concentrated in this neuron. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

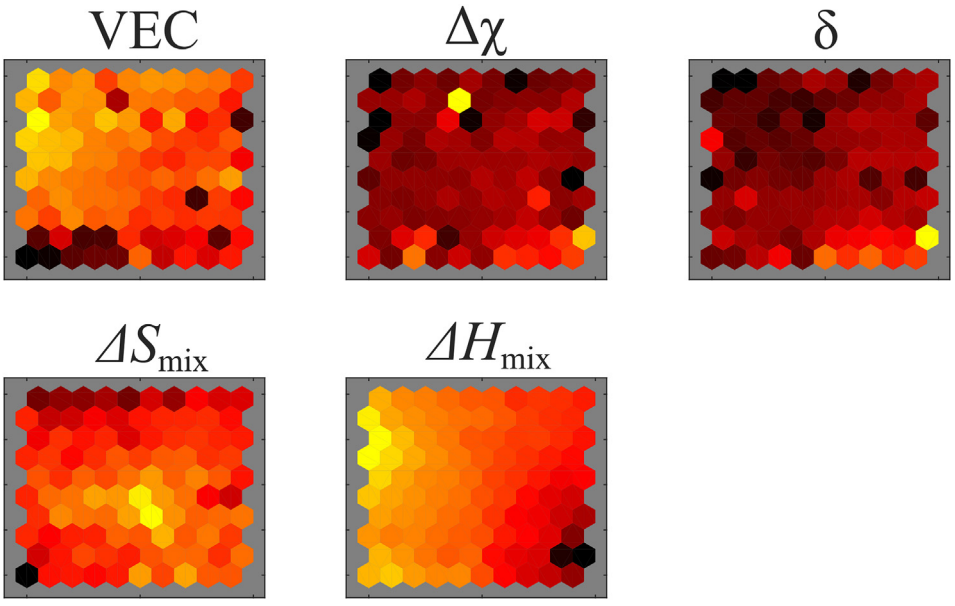

Fig. 7. Weight planes showing the weight distributions of 10 × 10 neurons for each of the five input features.

classification problem is how to separate the SS and SS + IM phases rather than to distinguish the IM phase, because of the relatively clear boundary of the IM phase from the other two phases.

3.2.2. MLFFNN

We now set to focus on applying the MLFFNN algorithm to the full data set. We again use two steps: training and a four-fold cross validation. During the training process, we achieve high accuracy values if there are sufficient number layers and neurons in the network. But the testing accuracy is much lower than the training accuracy due to overfitting. Our goal is therefore to reduce the degree of overfitting to improve the testing accuracy.

We attempt to reach a trade-off between the bias and variance in the data, so we constantly change the number of neurons in the three hidden layers to seek the optimal testing accuracy. For each combination (cycle) of n_1, n_2, and n_3 neurons in the hidden layers, we perform a four-fold validation to obtain the maximum, average, and minimum accuracy values. Fig. 8 shows a scatter plot of the average accuracy obtained from the 729 cycles. The error-bar plot in Fig. 9 shows the three accuracy values for each cycle.

In the total 729 cycles, we find that the combination of $n_1 = 20$, $n_2 = 15$, and $n_3 = 25$ results in the highest average testing accuracy (74.3%) in classifying all the three phases. We therefore further evaluate the performance of the network based on this combination of neurons in the hidden layers. We compute the MSE and the linear regression coefficient R, which are displayed in Fig. 10. We observe from Fig. 10(a) that the MSE decreases as the training progresses and converges to the "Best" value of 0.023. Although R shown in Fig. 10(b) is high ($> \sim 0.95$), the MSE is not sufficiently small, causing the low average testing accuracy and implying the complexity of classifying three phases in HEAs.

To evaluate the relative importance of the five input features, we remove one of the features in succession and retrain the neural network to study how this removal affects the testing accuracy. Fig. 11 (a) shows the accuracy decrease due to the removal of the corresponding features. We observe all of the accuracy decrease is

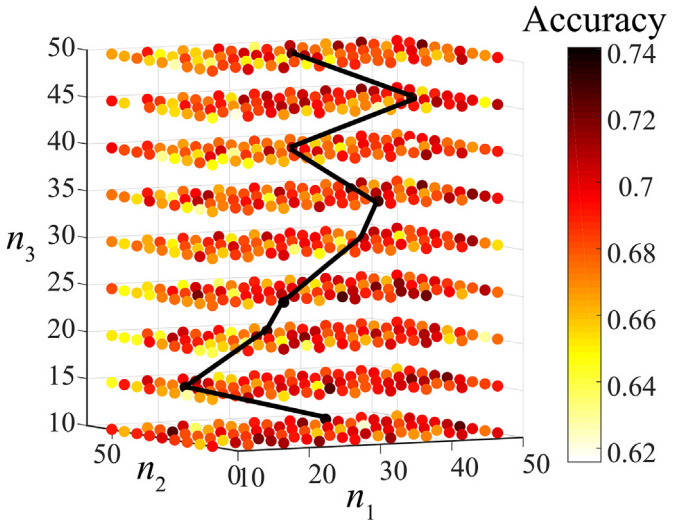

Fig. 8. A 3D scatter plot showing how the average testing accuracy of classifying the SS, IM and SS + IM phases in high-entropy alloys varies with the numbers of neurons in each hidden layer, denoted by n_1, n_2, and n_3, respectively. The darker color corresponds to the higher accuracy value. The black lines connect the dots representing the highest accuracy value for a fixed n_3, when n_1 and n_2 are changed from 10 to 50 at a step of 5. The viewpoint of this plot is parallel to the n_1-n_2 plane. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

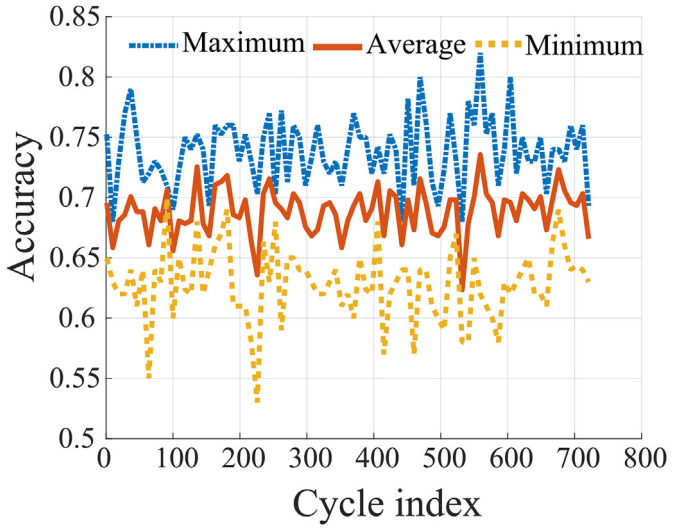

Fig. 9. An Error-Bar chart showing the maximum, average, and minimum testing accuracies resulting from the four-fold cross validation. The Cycle index refers to a different combination of the number of neurons in the three hidden layers. The top curve shows the maximum testing accuracy, while the bottom curve represents the lowest accuracy of the four-fold cross validation. Maximum and minimum accuracies correspond to overfitting and underfitting situations, respectively. The average accuracy is reported to remedy these two extreme scenarios.

positive, indicating that removing any of the five feature decreases the testing accuracy. In addition, the accuracy decrease follows the order: δ > VEC > ΔS_{mix} > ΔH_{mix} > $\Delta\chi$. Our previous study showed that ΔS_{mix} trivially affects the classification of the SS, IM, and amorphous phases [38]. According to our current calculations, we consistently observe that the testing accuracy only decreases by 1~2% after removing the $\Delta\chi$, ΔS_{mix}, or ΔH_{mix} feature. By contrast, the δ and VEC features seem to play the most important roles in determining the phase selection, providing guidance for designing HEAs by preferably considering these two features.

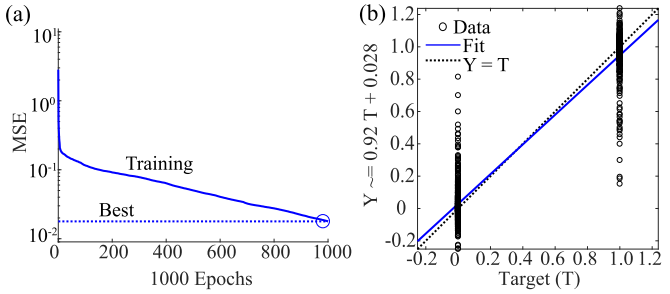

Fig. 10. (a) A mean-squared error (MSE) plot. The dotted line is the minimum (Best) MSE the network can reach after 1000 iterations and the solid curve shows the variation of the MSE during the training. (b) A regression plot. The coefficient of determination, "R^2", (0.92 in the regression plot) is considered to display the degree of linear correlation between the target and the output value. The "Fit" line shows the fitted Y-T (target) relationship and the dash grey "Y-T" line displays the ideal fit of $R^2 = 1$. The three phases are mathematically represented by three vectors: (1,0,0), (0,1,0) and (0,0,1) in the network. But in a two-dimensional regression plot, they can only be denoted as "0" and "1" and the "Data" points indicate groups "0" or "1".

The above results show that directly classifying the three phases in HEAs is quite challenging. The boundary between the SS and SS + IM phases represented by any pair of the five features is blurry, and some categories even overlap when projected into a plane. Note that it is also challenging to distinguish these two phases in experiments. For example, for the same HEA, AlCoCrCuFeNi, one experiment reports that the phase is SS [72], while another experiment claims the phase to be SS + IM [19].

Given the complexity of classifying the three phases simultaneously, referring to the binary classification of SVMs, we transfer the ternary problem into three binary classification problems: the SS and IM phases, the SS + IM and IM phases, and the SS and SS + IM phases. For all these three classification processes, Fig. 12 shows the scatter plots of the average testing accuracy values for various combinations of neurons in the three hidden layers. Fig. 13 depicts an Error-Bar plot displaying the maximum, average, and minimum accuracy values for the three classification problems. The MSE and linear regression coefficient R are displayed in Fig. 14 as a criterion to evaluate the performance of our network. We find that the average testing accuracy values in classifying the SS and IM phase, the SS + IM and IM phases, and the SS and SS + IM phases can reach 86.7%, 94.3%, and 78.9%, respectively. The maximum testing values are 89.3%, 94.7%, and 81.6% respectively. We find that high average testing accuracy values can be achieved under different combinations of neurons in the hidden layers. During the testing process, it is straightforward to differentiate the SS and IM phases, and the SS + IM and IM phases with much less neurons in the network. By contrast, it remains challenging to distinguish the SS and SS + IM phases. The high testing accuracy in the binary classifications of these cases enables to perform an indirect classification of the three phases. For example, one can apply our trained models in a two-step process: the first to distinguish the SS of SS+IM from IM phases and the second to discern the SS + IM from IM phases.

Because classifying the SS and IM phases and the SS+IM and IM phases results in high testing accuracy, we also assess the relative importance of the five input features and the results are shown in Fig. 11(b) and (c), respectively. We observe that, the order of importance of the features in classifying the SS + IM and IM phases follows the same trend as in classifying all the three phase simultaneously. Although ΔS_{mix} and ΔH_{mix} start to play more important roles in affecting the testing accuracy for the case of classifying the SS and IM phases, the trend of relative importance is different from the other two cases, indicating that the role of ΔS_{mix} remains insignificant.

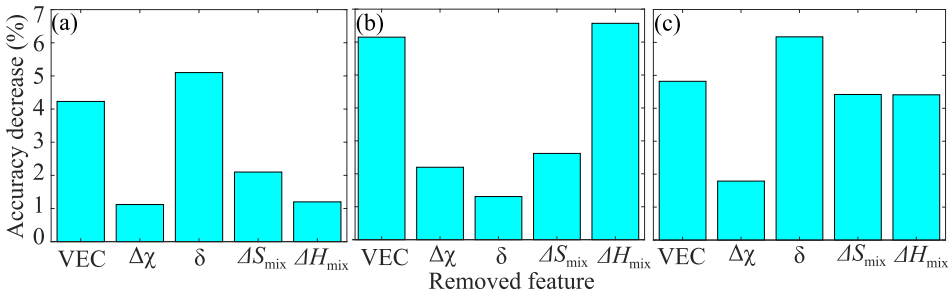

Fig. 11. Effect of removing one of the five features on the original testing accuracy when classifying (a) all the three phases, (b) the SS and IM phases and (c) the SS + IM and IM phases with the multi-layer feed-forward neural network method. The larger the accuracy decreases indicates the more important this feature for classifying the phases in high-entropy alloys.

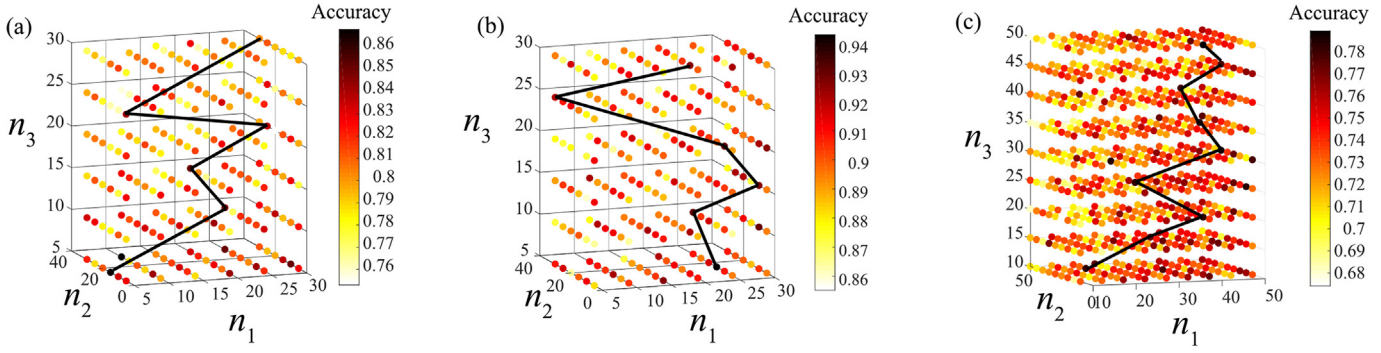

Fig. 12. Scatter plots showing the average testing accuracy of classifying two phases varies with the number of neurons in each hidden layer, denoted as n_1, n_2, and n_3, respectively: (a) SS and IM; (b) SS + IM and IM; (c) SS and SS + IM. The solid black lines connect the data points that correspond to the highest testing accuracy for a fixed n_3 when n_1 and n_2 are changed from 5 to 30 in (a) and (b), and from 10 to 50 in (c). The viewpoint of these plots is parallel to the n_1-n_2 plane.

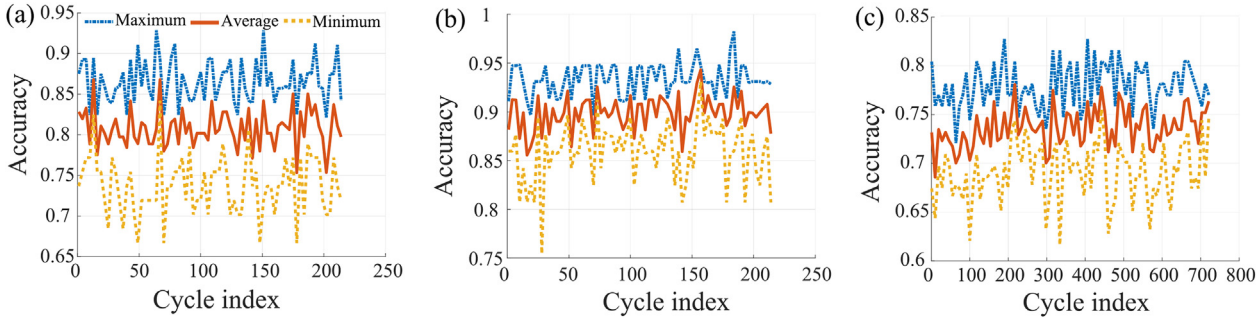

Fig. 13. Error-Bar plots showing the maximum, average, and minimum testing accuracies of classifying (a) the SS and IM phases, (b) the SS + IM and IM phases, and (c) the SS and SS + IM phases. The cycle index refers to a combination of different numbers of neurons from the three hidden layers. The top, middle, and bottom curves in each panel display the maximum, average, and minimum accuracy, respectively, resulting from a four-fold cross validation. Maximum and minimum accuracies correspond to overfitting and underfitting situations, respectively. The average accuracy is reported to remedy these two extreme scenarios.

Fig. 1 shows that there is a significant overlap under a two-dimensional projection regardless of any combination of two features, alluding to the factor that causes the problem of classification: these five features in our work may not be the most decisive factors that determine the phase selection of HEAs. Instead, there may be other more effective elemental properties that are more helpful to discern the phases in HEAs. The polarizability of the elements forming a metal alloy is proposed in the literature—as another factor in addition to the Hume-Rothery rules—to exhibit a correlation with the crystal structure of the alloy [73]. We therefore include the average polarizability calculated in a similar way as that for $\overline{\chi}$ or $\overline{r}$ as the sixth feature to classify the most challenging SS and SS + IM phases using the MLFFNN model that achieves the highest accuracy (78.9%) with five input features. The corresponding testing accuracy values obtained from the four-fold cross validation are 80.5%, 80.5%, 82.6%, and 86.1%, respectively, and the average accuracy is thus 82.1%, which is 3.2% higher than that from the MLFFNN with five input features. We therefore suggest that an improved testing accuracy could be achieved by using more relevant features.

To show the applications of our machine learning models (e.g., SVM or MLFFNN), we compare the accuracy levels obtained in this work to that from using parametric methods. For example, ΔH_{mix} and δ have been used to distinguish the SS and SS + IM phases. It is found that if a HEA has δ smaller than 6.7% and ΔH_{mix} larger than −11.66 kJ/mol, the phase is designed as the SS phase [74–76].

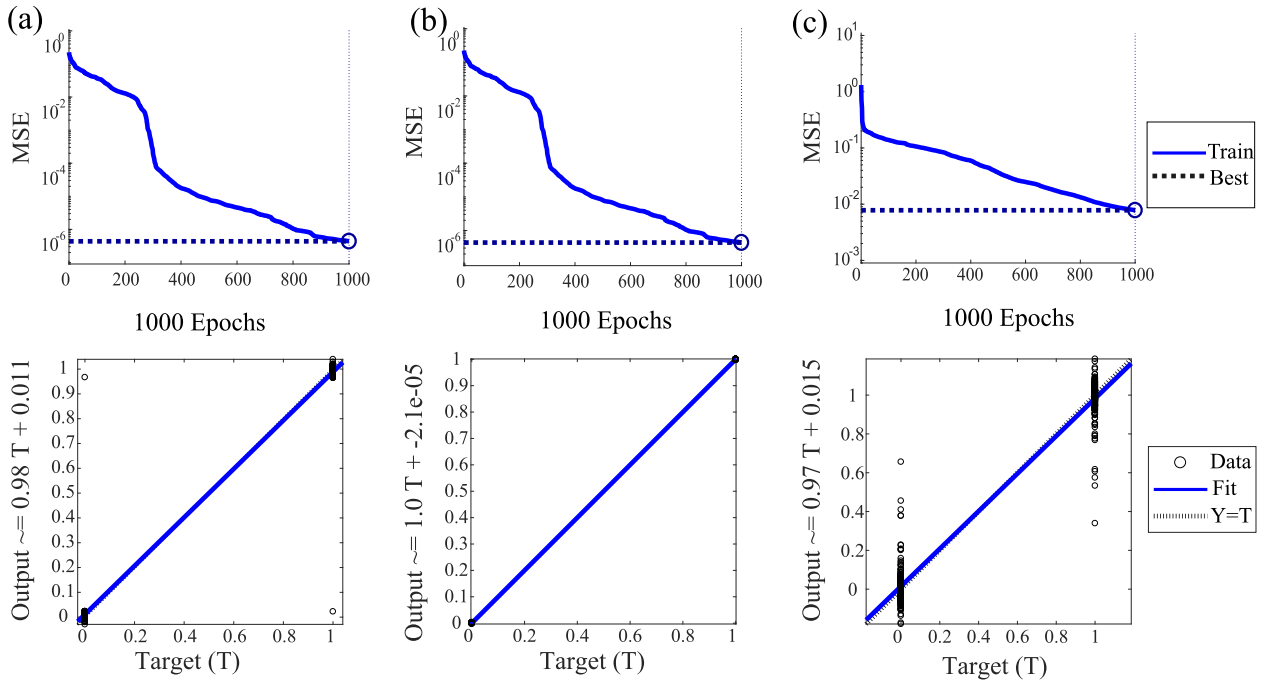

Fig. 14. Top panels: mean-squared error (MSE) during training. Bottom panels: the maximum degree of regression the multi-layer feed-forward neural network can reach for classifying (a) the SS and IM phases, (b) the SS + IM and IM phases, and (c) the SS and SS + IM phases.

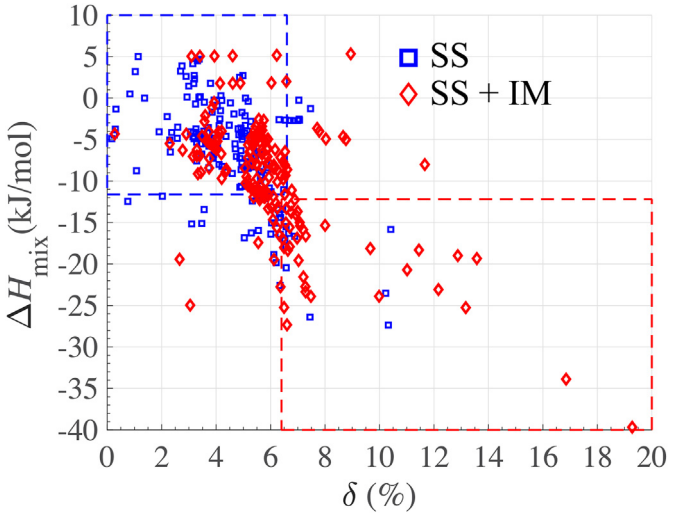

Fig. 15. A δ-ΔH_{mix} plot showing the labels of the phases in 347 high-entropy alloys.

We apply these two parameters to our 347 samples (174 SS and 173 SS + IM phases). Fig. 15 shows that 134 samples are mislabeled (including 13 mislabeled SS and 121 mislabeled SS + IM phases), i.e., the final accuracy for this method is only 57.3%, much smaller than the accuracy achieved from our machine learning calculations.

4. Conclusions

In conclusion, we used three different ML algorithms: KNN—Fine and Weight KNNs, SVM and ANN—the unsupervised learning algorithm SOM and supervised learning MLFFNN to determine the phase selection using a dataset of 401 HEAs. We find that the training accuracy can always be improved to a high level by adjusting the hyperparameters involved in the training process, independent of any particular ML algorithm. But the suitability of the selected model is highly determined by its ability to perform well on unseen data(testing data). From this perspective, the KNN methods lead to a lower testing accuracy of 68.6%, largely because of the imbalance of dataset. Considering SVM and ANN method, the SOM model provides the evidence that all the five input features are nearly independent from each other, which is consistent with the result from Fig. 1, and it consolidates the premise of our classification problem. Ternary classification of directly differentiating the SS, IM, and SS + IM phases using the SVM and MLFFNN, the testing accuracy can only reach 64.3% and 74.3%, respectively. The main reason is because of the blurry boundary between the SS and SS + IM phases. We therefore use the same SVM and MLFFNN framework to do binary classifications composed of pairs of the three phases: the SS and IM phases, the SS + IM and IM phases, as well as the SS and SS + IM phases through the four-fold cross validation. The average testing accuracy values of MLFFNN reach 86.7%, 94.3%, and 78.9%, respectively, all of which are higher than those from using SVM. We can therefore conclude that the MLFFNN is the best ML model for our classification problem.

We also evaluated the relative importance of the five input features in affecting the testing accuracy. We found that the δ and VEC features are more crucial for the phase selection than the ΔH_{mix}, ΔS_{mix}, and $\Delta\chi$ features. In the future work, we suggest that more relevant input features such as the atomic polarizability should be included as additional features to improve the accuracy of predicting the phases in HEAs. Alternatively, optimizing the number of hidden layers and neurons along with adjusting the internal parameters such as the training function is also expected to help improve the testing accuracy. Overall, we expect this work to provide guidance on the design and phase prediction of new HEAs using ML methods.

Acknowledgements

We thank the start-up funds from Arizona State University. This work used computational resources of the Texas Advanced Computing Center under Contract No. TG-DMR170070.

References

[1] Y. Ye, Q. Wang, J. Lu, C. Liu, Y. Yang, High-entropy alloy: challenges and prospects, Mater. Today 19 (2016) 349–362.
[2] M. Peters, J. Kumpfert, C.H. Ward, C. Leyens, Titanium alloys for aerospace applications, Adv. Eng. Mater. 5 (2003) 419–427.
[3] K.U. Kainer, Metal Matrix Composites: Custom-Made Materials for Automotive and Aerospace Engineering, John Wiley & Sons, 2006.
[4] R. Montanari, B. Riccardi, R. Volterri, L. Bertamini, Characterisation of plasma sprayed w coatings on a cucrzr alloy for nuclear fusion reactor applications, Mater. Lett. 52 (2002) 100–105.
[5] G. He, J. Eckert, Q. Dai, M. Sui, W. Löser, M. Hagiwara, E. Ma, Nanostructured ti-based multi-component alloys with potential for biomedical applications, Biomaterials 24 (2003) 5115–5120.
[6] M.-H. Tsai, J.-W. Yeh, High-entropy alloys: a critical review, Mater. Res. Lett. 2 (2014) 107–123.
[7] J.-W. Yeh, S.-K. Chen, S.-J. Lin, J.-Y. Gan, T.-S. Chin, T.-T. Shun, C.-H. Tsau, S.-Y. Chang, Nanostructured high-entropy alloys with multiple principal elements: novel alloy design concepts and outcomes, Adv. Eng. Mater. 6 (2004) 299–303.
[8] B. Cantor, I. Chang, P. Knight, A. Vincent, Microstructural development in equiatomic multicomponent alloys, Mater. Sci. Eng., A 375 (2004) 213–218.
[9] B. Cantor, F. Audebert, M. Galano, K. Kim, I. Stone, P.J. Warren, Novel multicomponent alloys 24 (2005) 1–6.
[10] C.-J. Tong, M.-R. Chen, J.-W. Yeh, S.-J. Lin, S.-K. Chen, T.-T. Shun, S.-Y. Chang, Mechanical performance of the $Al_xCoCrCuFeNi$ high-entropy alloy system with multiprincipal elements, Metall. Mater. Trans. 36 (2005) 1263–1271.
[11] Y. Zhang, Y.J. Zhou, Solid solution formation criteria for high entropy alloys 561 (2007) 1337–1339.
[12] C. Li, J. Li, M. Zhao, Q. Jiang, Effect of alloying elements on microstructure and properties of multiprincipal elements high-entropy alloys, J. Alloy. Comp. 475 (2009) 752–757.
[13] Y. Zhang, Mechanical Properties and Structures of High Entropy Alloys and Bulk Metallic Glasses Composites, vol. 654, 2010, pp. 1058–1061.
[14] H. Diao, R. Feng, K. Dahmen, P. Liaw, Fundamental deformation behavior in high-entropy alloys: an overview, Curr. Opin. Solid State Mater. Sci. 21 (5) (2017) 252–266.
[15] Y. Zhou, Y. Zhang, Y. Wang, G. Chen, Solid solution alloys of Al-Co-Cr-Fe-Ni-Ti_x with excellent room-temperature mechanical properties, Appl. Phys. Lett. 90 (2007) 181904.
[16] L. Wen, H. Kou, J. Li, H. Chang, X. Xue, L. Zhou, Effect of aging temperature on microstructure and properties of alcocrcufeni high-entropy alloy, Intermetallics 17 (2009) 266–269.
[17] P.-K. Huang, J.-W. Yeh, T.-T. Shun, S.-K. Chen, Multi-principal-element alloys with improved oxidation and wear resistance for thermal spray coating, Adv. Eng. Mater. 6 (2004) 74–78.
[18] C.-M. Lin, H.-L. Tsai, Evolution of microstructure, hardness, and corrosion properties of high-entropy $Al_{0.5}CoCrFeNi$ alloy, Intermetallics 19 (2011) 288–294.
[19] M.-H. Tsai, Physical properties of high entropy alloys, Entropy 15 (2013) 5338–5345.
[20] D. King, S. Middleburgh, A. McGregor, M. Cortie, Predicting the formation and stability of single phase high-entropy alloys, Acta Mater. 104 (2016) 172–179.
[21] M.C. Gao, J.-W. Yeh, P.K. Liaw, Y. Zhang, High-Entropy Alloys, Springer, 2016.
[22] W.P. Huhn, M. Widom, Prediction of A2 to B2 phase transition in the high-entropy alloy Mo-Nb-Ta-W, J. Occup. Med. 65 (2013) 1772–1779.
[23] R. Feng, P.K. Liaw, M.C. Gao, M. Widom, First-principles prediction of high-entropy-alloy stability, npj Comput. Mater. 3 (1) (2017) 50.
[24] Y. Zhang, Y.J. Zhou, J.P. Lin, G.L. Chen, P.K. Liaw, Solid-solution phase formation rules for multi-component alloys, Adv. Eng. Mater. 10 (2008) 534–538.
[25] Y. Zhang, S. Yang, J. Evans, Revisiting hume-rotherys rules with artificial neural networks, Acta Mater. 56 (2008) 1094–1105.
[26] M. Rupp, A. Tkatchenko, K.-R. Müller, O.A. Von Lilienfeld, Fast and accurate modeling of molecular atomization energies with machine learning, Phys. Rev. Lett. 108 (2012) 058301.
[27] B. Meredig, C. Wolverton, A hybrid computational–experimental approach for automated crystal structure solution, Nat. Mater. 12 (2013) 123.
[28] L. Ward, A. Agrawal, A. Choudhary, C. Wolverton, A general-purpose machine learning framework for predicting properties of inorganic materials, NPJ Computational Materials 2 (2016) 16028.
[29] Y. Lin, J. Zhang, J. Zhong, Application of neural networks to predict the elevated temperature flow behavior of a low alloy steel, Comput. Mater. Sci. 43 (2008) 752–758.
[30] W. Hume-Rothery, B. Coles, Atomic Theory for Students of Metallurgy, Book (Institute of Metals), Institute of Metals, 1988.
[31] C.-H. Zhang, M.-H. Lin, B. Wu, G.-X. Ye, L.-K. Zhang, T. Chen, W.-J. Zhang, Z.-H. Zheng, Q. Li, Y.-Q. Shao, Explore the possibility of forming fcc high entropy alloys in equal-atomic systems CoFeMnNiM and CoFeMnNiSmM, J. Shanghai Jiaot. Univ. 16 (2011) 173.
[32] D. Miracle, O. Senkov, A critical review of high entropy alloys and related concepts, Acta Mater. 122 (2017) 448–511.
[33] C. Liu, Physical metallurgy and mechanical properties of ductile ordered alloys $(Fe, Co, Ni)_3V$, Intl. Metals Rev. 29 (1984) 168–194.
[34] J. Zhu, P. Liaw, C. Liu, Effect of electron concentration on the phase stability of $NbCr_2$-based laves phase alloys, Mater. Sci. Eng., A 239 (1997) 260–264.
[35] S. Fang, X. Xiao, L. Xia, W. Li, Y. Dong, Relationship between the widths of supercooled liquid regions and bond parameters of Mg-based bulk metallic glasses, J. Non-Cryst. Solids 321 (2003) 120–125.
[36] L. Pauling, The Nature of the Chemical Bond, vol. 260, Cornell university press Ithaca, NY, 1960.
[37] A. Takeuchi, A. Inoue, Classification of bulk metallic glasses by atomic size difference, heat of mixing and period of constituent elements and its application to characterization of the main alloying element, Mater. Trans. 46 (2005) 2817–2829.
[38] N. Islam, W. Huang, H.L. Zhuang, Machine learning for phase selection in multi-principal element alloys, Comput. Mater. Sci. 150 (2018) 230–235.
[39] C.J. Stone, Consistent nonparametric regression, Ann. Stat. (1977) 595–620.
[40] N.S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, Am. Statistician 46 (1992) 175–185.
[41] A. Mucherino, P.J. Papajorgji, P.M. Pardalos, K-nearest neighbor classification, in: Data Mining in Agriculture, Springer, 2009, pp. 83–106.
[42] P.-E. Danielsson, Euclidean distance mapping, Comput. Graph. Image Process. 14 (1980) 227–248.
[43] P. Hall, B.U. Park, R.J. Samworth, et al., Choice of neighbor order in nearest-neighbor classification, Ann. Stat. 36 (2008) 2135–2152.
[44] S. Sivanandam, S. Deepa, Introduction to Neural Networks Using Matlab 6.0, Tata McGraw-Hill Education, 2006.
[45] S.B. Kotsiantis, I. Zaharakis, P. Pintelas, Supervised machine learning: a review of classification techniques, Emerging artificial intelligence applications in computer engineering 160 (2007) 3–24.
[46] I. Steinwart, A. Christmann, Support Vector Machines, Springer Science & Business Media, 2008.
[47] S. Amari, S. Wu, Improving support vector machine classifiers by modifying kernel functions, Neural Network. 12 (1999) 783–789.
[48] W. Wang, Z. Xu, W. Lu, X. Zhang, Determination of the spread parameter in the Gaussian kernel for classification and regression, Neurocomputing 55 (2003) 643–663.
[49] B. Schölkopf, A. Smola, Support vector machines, The Handbook of Brain Theory and Neural Networks (2003) 1119–1125.
[50] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, B. Scholkopf, Support vector machines, IEEE Intell. Syst. Their Appl. 13 (1998) 18–28.
[51] B. Schölkopf, A. Smola, Kernel Methods and Support Vector Machines, 2003.
[52] A. Shmilovici, Support vector machines, in: Data Mining and Knowledge Discovery Handbook, Springer, 2009, pp. 231–247.
[53] L. Zhang, W. Zhou, L. Jiao, Wavelet support vector machine, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 34 (2004) 34–39.
[54] B.M. ter Haar Romeny, The Gaussian kernel, front-end vision and multi-scale image analysis: multi-scale computer vision theory and applications, written in Mathematics (2003) 37–51.
[55] E. Byvatov, U. Fechner, J. Sadowski, G. Schneider, Comparison of support vector machine and artificial neural network systems for drug/nondrug classification, J. Chem. Inf. Comput. Sci. 43 (2003) 1882–1889.
[56] R.S. Michalski, J.G. Carbonell, T.M. Mitchell, Machine Learning: an Artificial Intelligence Approach, Springer Science & Business Media, 2013.
[57] J. A. Bullinaria, Introduction to Neural Networks, School of Computer Science, The University of Birmingham, Birmingham, UK.
[58] T. Kohonen, The self-organizing map, Proc. IEEE 78 (1990) 1464–1480.
[59] T. Kohonen, T. Honkela, Kohonen network, Scholarpedia 2 (2007) 1568.
[60] J. Vesanto, E. Alhoniemi, Clustering of the self-organizing map, IEEE Trans. Neural Netw. 11 (2000) 586–600.
[61] J.E. Houlahan, S.T. McKinney, T.M. Anderson, B.J. McGill, The priority of prediction in ecological understanding, Oikos 126 (2017) 1–7.
[62] T. Heskes, Self-organizing maps, vector quantization, and mixture modeling, IEEE Trans. Neural Netw. 12 (2001) 1299–1305.
[63] A.F. Taktak, A.C. Fisher, Outcome Prediction in Cancer, Elsevier, 2006.
[64] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Network. 4 (1991) 251–257.
[65] A.K. Jain, J. Mao, K.M. Mohiuddin, Artificial neural networks: a tutorial, Computer 29 (1996) 31–44.
[66] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the marquardt algorithm, IEEE Trans. Neural Netw. 5 (1994) 989–993.
[67] J. Li, J.-H. Cheng, J.-Y. Shi, F. Huang, Brief introduction of back propagation (BP) neural network algorithm and its improvement, in: Advances in Computer Science and Information Engineering, Springer, 2012, pp. 553–558.
[68] F. Chollet, Deep Learning with python, Manning Publications Co., 2017.
[69] T. Dietterich, Overfitting and undercomputing in machine learning, ACM Comput. Surv. 27 (1995) 326–327.
[70] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
[71] I. Hmeidi, B. Hawashin, E. El-Qawasmeh, Performance of knn and svm classifiers on full word Arabic articles, Adv. Eng. Inf. 22 (1) (2008) 106–111.

[72] J.-M. Wu, S.-J. Lin, J.-W. Yeh, S.-K. Chen, Y.-S. Huang, H.-C. Chen, Adhesive wear behavior of alxcocrcufeni high-entropy alloys as a function of aluminum content, Wear 261 (2006) 513–519.
[73] L. Darken, R. Gurry, Physical Chemistry of Metals, Metallurgy and Metallurgical Engineering Series, McGraw-Hill, 1953.
[74] E. Pickering, N.G. Jones, High-entropy alloys: a critical assessment of their founding principles and future prospects, Int. Mater. Rev. 61 (2016) 183–202.
[75] Y. Ye, Q. Wang, J. Lu, C. Liu, Y. Yang, High-entropy alloy: challenges and prospects, Mater. Today 19 (6) (2016) 349–362.
[76] S. Guo, Q. Hu, C. Ng, C. Liu, More than entropy in high-entropy alloys: forming solid solutions or amorphous phase, Intermetallics 41 (2013) 96–103.