Fusing separated representation into an autoencoder for magnetic materials outlier detection

Ying Cao\textsuperscript{a,b} and SunWoo Ko\textsuperscript{b}

\textsuperscript{a}Information Centre, Jiangxi University of Science and Technology, Ganzhou, People’s Republic of China; \textsuperscript{b}Department of Artificial Intelligence, Jeonju University, Jeonju, Republic of Korea

\textbf{ABSTRACT}

In materials science, an outlier may be due to variability in measurement, or it may indicate experimental errors. In this paper, we used an unsupervised method to remove outliers before further data-driven material analysis. Recently, autoencoder networks have achieved excellent results by minimizing reconstruction error. However, autoencoders do not promote the separation between outliers and inliers. The proposed SRAE model integrates latent representation to optimize the reconstruction error and ensures that outliers always deviate from the dataset in the compressed representation space. Experiments on the Nd-Fe-B magnetic materials dataset also show that after removing outliers with the proposed method, the prediction result of material property is significantly improved, indicating that the outlier detection effect is excellent.

\textbf{ARTICLE HISTORY}

Received 20 September 2021
Accepted 9 March 2022

\textbf{KEYWORDS}

Autoencoder; outlier detection; latent representation; outlier score; magnetic materials

1. Introduction

In materials science, data-driven machine learning has gradually become an important research method. Raccuglia et al. (2016) pointed out that today’s material research is slowly moving from the manual era of computational material science to the industrialization stage, and there will be more materials predicted by data-driven technology in the next decade. The materials science that combines computers and artificial intelligence is called materials informatics. In recent years, it has been successfully applied in new material discovery (Sendek et al., 2016), material design (Butler et al., 2016), and properties prediction (Katsikas et al., 2021).

The research of machine learning in magnetic materials is still being explored. Hosokawa et al. (2021) predict the magnetic properties of SmFeN through neural network regression by chemical composition, process parameters, and heat treatment parameters. Park et al. (2021) predict reliable values of coercivity and maximum magnetic energy product of granular Nd-Fe-B magnets according to their microstructural attributes (e.g. inter-grain decoupling, average grain size, and misalignment of easy axes) based on numerical datasets obtained from micromagnetic simulations. Nelson and Sanvito (2019) introduce a range of machine-learning models to predict the Curie temperature of ferromagnets. Studies have shown that experimental exploration plus machine learning can accelerate the speed and efficiency of discovery of new materials.

Since the machine learning method depends on the data to be analyzed, the results are strongly affected by the size, distribution, and quality of the dataset. In this regard, unreliable, inaccurate, noisy data often hinder learning efficiency and, in extreme cases, even mislead the learning process and lead to biased predictions.

Material engineering is experimental. However, in actual engineering experiments, scientific measurements are never perfect. In addition to random errors (Barnett & Lewis, 1994), there are often inherent variability, measurement errors, and execution errors caused by various potential sources. These errors may be a row of data (outliers) or an instance property (noise).

This is a severely problematic issue since the datasets for material engineering applications are based on experimental observations. The data quality determines the upper limit of model performance, and the algorithms only approach the upper limit indefinitely. When further data analysis and mining are to be carried out, the influence of the data should be handled efficiently. Since experimental material data errors are inevitable, detecting outliers will be an important part of data preprocessing. In this paper, we will conduct property prediction on the dataset of rare earth magnetic materials, and outlier elimination will be used as part of data
preprocessing before further analysis such as regression-based prediction.

Outlier detection is referred to as the process of detecting data instances that significantly deviate from the majority of data (Pang et al., 2021). Usually, outliers are caused by different mechanisms which induce deviations from other samples. The task of outlier detection has been applied to many fields, such as credit card fraud detection, loan approval, e-commerce, network intrusion, and weather forecasting. Most outlier detection methods evaluate outliers by quantifying outlier scores.

The deep learning technique is explored to address the outlier detection, for instance through the autoencoder networks which are especially suitable for unsupervised tasks (Cheng et al., 2021; Zhou & Paffenroth, 2017). The autoencoder network uses the reconstruction error as the objective function and achieves competitive results.

In this paper, we propose an improved method for outlier detection based on autoencoder, which makes use of the autoencoder network structural characteristic and uses the compressed latent representation to facilitate the separation between outliers and inliers, that is better tailored for outlier detection task. The main contributions of this study are as follows.

- For the first time, a data-driven research application is carried out on the real actual experimental data of Nd-Fe-B materials. To improve the quality of material property prediction, an improved outlier detection algorithm is proposed.
- The proposed method, called SRAE, is designed for outlier detection tasks based on autoencoders, which unify the two independent tasks of low-dimensional representation learning and reconstruction error learning. A proposal is made to fuse the separation loss into reconstruction error to guide the optimized learning that is conducive to our outlier detection task.
- We Conduct extensive experiments that show that our approach is competitive with respect to the state-of-the-art methods. The results show that the improved algorithm compared with the autoencoder baseline increased by 2.34% in terms of $R^2$, compared with the classic algorithms Local Outlier Factor, Isolation Forest, Elliptic Envelope increased by 7.02%, 14.7%, 2.9%, respectively. In addition, Experiments also show that the proposed method has good stability and robustness to the structural changes of the automatic encoder network.

2. Related work

Outlier detection is the process of identifying the observations that deviate substantially from the majority of data. The outlier removal procedure for experimental material data discussed in this paper involves data that does not have labels for training, so the following discussions are all unsupervised outlier detection models. Unsupervised outlier detection aims to automatically find abnormal samples based on the intrinsic properties of the datasets without any manually labelled data.

Various outlier detection techniques have been reported in the literature. For instance, neighbour-based methods (Knorr & Ng, 1999; Pang et al., 2015; Sugiyama & Borgwardt, 2013) assume that positive data have close neighbours, while outliers are far from each other. While the disadvantage is its sensitivity to the nearest neighbours’ number, which is challenging to specify a priori, subspace-based methods (Keller et al., 2012; Pevný, 2016) define outliers using a set of relevant feature subspaces to avoid the curse of dimensionality. Outlying feature selection retains a feature subset relevant to outlier detection. The limitation of this method is that blind adaptations of dimension selection methods from earlier subspace clustering methods that are unaware of the nuances of subspace analysis principles across different problems, may sometimes miss significant outliers.

As a powerful learning tool, neural networks have also been used for outlier detection, and autoencoders (Kramer, 1991) are one of the fundamental architectures being deployed. Autoencoder is a unique neural network architecture with the same input and outputs. The autoencoder consists of two parts: an encoder network is responsible for mapping the original data to a low-dimensional feature space, a decoder network is responsible for reconstructing the low-latitude information back to a high-dimensional data expression. The autoencoder is reconstructed by minimizing an error objective function and iteratively learning the parameters of these two networks. $Z$ is the potential feature space, which is a low-dimensional representation of the original data. To minimize the reconstruction error, the feature representation trained by autoencoder networks must be able to represent the statistical characteristics of the entire dataset, as closely as possible. To retain and dominate example related information from normal samples as much as possible, the inliers (also called normal examples) tend to have smaller reconstruction errors than the outliers (Sakurai & Yairi, 2014). Based on this, one can conveniently identify the samples with significant reconstruction errors as outliers. Therefore, the data reconstruction errors can be directly used as outlier scores (An & Cho, 2015). Many researchers (Morales-Forero & Bassetto, 2019; Zhou &
Paffenroth, 2017) have proposed improved models based on autoencoders, applied them to abnormal point detection in various scenarios, and achieved good results.

In this paper, we propose an outlier detection method based on an autoencoder while paying more attention to the degree of separation of outliers. The improved autoencoder model is denoted as SRAE. To test the data cleaning method, we compare the regression accuracy of the basic machine learning model before and after data cleaning and evaluate the generalization ability of the detector. We also use the most advanced outlier detection methods, such as elliptic envelope (Rousseeuw & Van Driessen, 1999) isolation forest (Liu et al., 2008) and local outlier factor abbreviated as LOF (Breunig et al., 2000) to compare with the proposed methods.

3. The proposed method

The objective function of the data reconstruction in autoencoder (Hinton & Salakhutdinov, 2006) is designed for dimension reduction or data compression, rather than outlier detection. As a result, the resulting representations are a generic summarization of underlying regularities, which are not optimized for differentiating outliers from the dataset.

We proposed an improved autoencoder outlier detection algorithm called the Separated Representation Autoencoder (SRAE), which integrates latent representation features into the learning stage to optimize the reconstruction error. First, the latent representation is encoded, the outlier and inlier are prelabelled, and then the decoder is updated by minimizing the inlier reconstruction error. These two steps, encoding labelling and reconstruction learning, are performed alternatively, network parameters are updated gradually and data labels are refined step by step.

We can train the network in two stages:

- Representation learning: Update the encoder to learn the representation to facilitate the separation between outliers and inliers.
- Reconstruction learning: Update the decoder to reduce reconstruction error while making it more distinguishing for outliers.

3.1. Baseline autoencoder for outlier detection

As an effective outlier detection model, an autoencoder can reconstruct normal samples, but it cannot restore outliers that differ from the normal distribution, resulting in large reconstruction errors. Given the dataset $X$, we first apply an autoencoder to compress $X$ into a low-dimensional intermediate representation $Z$ and then inverse-map it back to a reconstructed $\hat{X}$. The encoder function is $z_i = f(x_i)$ with nonlinear neurons, and the decoder function is $\hat{x}_i = g(z_i)$ with the antisymmetric shape of the encoder. The reconstruction error of $x_i$ is equal to the mean squared error (Kramer, 1991):

$$error_i = \frac{1}{2}||x_i - \hat{x}_i||^2$$

(1)

The autoencoder network is learned by minimizing the reconstruction loss:

$$loss = \frac{1}{2|X|} \sum_{x_i \in X} ||x_i - g(f(x_i))||^2$$

(2)

where $error_i$ is a good indicator of whether a sample $x_i$ is an inlier or outlier, $error_i$ of outliers is expected to be large.

3.2. Improved autoencoder method

3.2.1. Separation learning

Studies have shown that mapping data to low-dimensional space to detect outliers is an effective way (Pang et al., 2018), because whatever dimension you project the data into, it reserves variance and intrinsic local structure in the data.

When training an autoencoder network, it generates a latent low-dimensional representation of the original data, inspired by Caberoa et al. (2021) who combine projections into relevant subspaces with a nearest-neighbour algorithm. We define an outlier weight for each data point as the sum of its distances from its $k$ nearest neighbours. Outliers are those data points having the largest values of outlier weight (Sugiyama & Borgwardt, 2013).

$$\text{ow}(x_i) = \sum_{j=1}^{k} \text{dist}(x_i, nn_j(f(x_i)))$$

(3)

$$\text{dist}(x_i, x_j) = \sqrt{|f(x_i) - f(x_j)|^2}$$

(4)

$$X^+ = \arg \min_{|X|-r} (\text{ow}(x_i))$$

(5)

$$X^- = \arg \max_{|X|-r} (\text{ow}(x_i))$$

(6)

Given a data sample $x_i$, define its outlierness as $\text{ow}(x_i)$, the larger the value is, the more likely it is to be an outlier. The K nearest neighbours set of $x_i$ is $\text{nn}_k(x_i)$, $\text{dist}(x_i, x_j)$ is the Euclidean Distance (D’Agostino & Dardanoni, 2009) between $x_i$ and $x_j$ in the latent space, and $f$ is the encoder function. Outlier ratio is the ratio, $|X|$ represents the size of the dataset, and $r$ represents the number of outliers, which is equal to the ratio $r/|X|$. $X^+$ is the pseudo inlier candidate set, constructed by selecting $r$ data samples...
with the bottom outlierness, and obviously, $X^-$ is pseudo-outlier candidate set constructed by selecting $r$ data samples with the top outlierness. All operations here are performed in the latent feature space $Z$.

In the learning procedure of the encoder network, the latent representation is constantly optimized, the inliers’ deviation is expected to be smaller than the outliers’ deviation, and they are expected to separate as much as possible. First, define the deviation for inliers

$$\text{dev}_{X^+} = \frac{1}{|X^+|} \sum_{x_m \in X^+} \omega(x_m)$$

(7)

then the deviation for outliers

$$\text{dev}_{X^-} = \frac{1}{|X^-|} \sum_{x_n \in X^-} \omega(x_n)$$

(8)

our goal is that the learned latent feature space can meet this requirement

$$\text{dev}_{X^+} < \text{dev}_{X^-}$$

(9)

We use the hinge loss function (Rosasco et al., 2004) to encourage the separation between $X^+$ and $X^-$ to guide the encoder network learning. Hinge loss is used for ‘maximum-margin’ most notably for SVM; hence, we define separation loss to maximize the margin between outliers and inliers,

$$L_s(\text{dev}_{X^+}, \text{dev}_{X^-}) = \max\{0, c \text{ + dev}_{X^+} - \text{dev}_{X^-}\}$$

(10)

where $c$ is the margin parameter, which controls the expected smallest interval between the two deviation degrees $\text{dev}_{X^+}$ and $\text{dev}_{X^-}$. The hinge loss is a convex function, which is used to penalize the violation of the requirement in Equation (9). If the margin between outliers and inliers is less than the margin parameter $c$, penalty will be effective.

### 3.2.2. Reconstruction learning

Although the reconstruction error can lead to competitive results, the autoencoder network uses the sample reconstruction error as the objective function for training, it treats the outliers and inliers equally, allowing the error distributions of outliers and inliers to overlapped.

In fact, whether the sample is an outlier or an inlier, as the training process proceeds, the model can remember enough information for constructing both inliers and outliers well, and the errors are all reduced. Xia et al. (2015) found that when an autoencoder updates its parameters, the gradient is averaged over all training data. This means that an autoencoder attempts to reduce the overall error not the error of every single datum. To generate a more distinguishing reconstructed error, we should put more effort into reducing the errors of inliers. So, if we obtain the label with a-priori knowledge in advance, the autoencoder will be prone to reconstruct the positive data better.

Hence, we defined the reconstruction loss by considering only positive data,

$$L_r(\text{error}_{X^+}) = \frac{1}{2|X^+|} \sum_{x \in X^+} (x - \hat{x})^2$$

(11)

where $X^+$ is the inlier pseudo set. Given the unsupervised task, there is no prior knowledge to indicate which data points are inlier, so to generate more discriminative errors, the premise is to prelabel the dataset.

#### 3.2.3. Loss function and optimization

We aim to find a good encoder $f$ that compresses the data into a latent representation space in which the outliers and inliers are more separated, and to find a good decoder $g$ that generates a more discriminative reconstruction error. According to formulas (10) and (11), we define the loss function as follows:

$$\text{LOSS} = \lambda L_s(\text{dev}_{X^+}, \text{dev}_{X^-}) + L_r(\text{error}_{X^+})$$

(12)

Both terms are important. The first term is separation loss, which is responsible for generating the latent space that is more suitable for outlier detection tasks, as well as trying to maximize the separation margin between outliers and inliers. The second term is the reconstruction loss, which is responsible for data reconstruction to minimize the errors. $\lambda$ controls the trade-off between these two losses. If $\lambda = 0$, only the reconstruction error loss is considered, but it is still different from the baseline autoencoder.

We train SRAE using backpropagation with gradient descent, iteratively updating the network parameters. Given the learning rate $\eta = 0.001$.

The encoder’s weights are updated by

$$w_f = w_f - \eta \left( \frac{\partial L_s}{\partial w_f} + \frac{\partial L_r}{\partial w_f} \right)$$

(13)

The decoder network is designed to recover input from a compressed representation, which is not responsible for the generation of latent representation, so there is one less constraint than the encoder network in parameters updating formula. So, the decoder’s weights are updated by

$$w_g = w_g - \eta \frac{\partial L_r}{\partial w_g}$$

(14)

#### 3.3. Outlier scoring function

The outlier detection algorithm will eventually compute an outlier score for each sample. Other literature directly
uses the reconstruction error value as the outlier score after sorting the abnormality scores and sets the cut-off to determine the outliers. Some researchers use the variance of the reconstruction error to determine the outliers. Inspired by (Leys et al., 2013), we define the outlier scoring function by the MAD of the reconstruction error. Absolute deviation from the median was (re)discovered and popularized by Hampel (1974). The median (M) is, similar to the mean, a measure of central tendency but offers the advantage of being very insensitive to the presence of outliers. The MAD is immune to the sample size. Therefore we define the mad score as

\[ \text{score}_i = b \times \text{median}(|\text{error}_i - \text{median}(|\text{error}|)) \] (15)

where \( \text{score}_i \) is the MAD score of \( x_i \), and \( b \) is set to 1.4826, which is a constant related to the assumption of normality of the data, disregarding the abnormality induced by outliers. Since score\(_i\) is scalar, eliminating outliers can be transformed into sorting the MAD score and determining the optimal threshold. Samples that have a MAD score larger than the threshold are identified as outlier candidates.

\[ \text{outlier} = \{ x_i | \text{score}_i > \text{threshold} \} \] (16)

### 3.4. The algorithm and time complexity

Algorithm 1 describes the procedure of SRAE. Step 1 uses random weight initialization. The epochs of network iterations in step 2 is automatically adjusted according to the convergence situation, where the maximum value is set to 1000. In each iteration, first, the encoder passes through some dense layers and activation function ReLU to generate low-dimensional feature representation \( z \). The size of the \( z \) dimension is tuned as a super parameter. Then, distance-based methods are used to generate inlier and outlier candidate sets \( X^+ \) and \( X^- \) in step 4. The nearest neighbour distance involved in Equation (4) is initialized with the default value \( k = 10 \), and Euclidean distance calculation is used. \( X^+ \) and \( X^- \) set sizes are hyperparameters. We found that when the outlier ratio = 0.1 is used, the performance is stable and effective.

Step 5. Use the decoder to inverse compression features by nonlinear transformation to restore the original data. According to the reconstruction output calculated in step 5 and the pseudo inlier set generated in step 4, the inlier error loss is calculated. In step 7, based on the outlier pseudo labels and inlier pseudo labels, respectively calculate their deviation in the feature space \( z \), and set \( c = 5 \) to promote greater separation between them. Step 8 defines the objective function for SRAE. Both items are designed to promote the separation between outliers and inliers. Step 9 minimizes the objective function, backpropagates by calculating the partial derivative, and uses Adam to optimize and perform gradient descent. SRAE iterative learning updates \( w_f \) and \( w_g \) in steps 3-9.

Finally, step 11 uses the trained encoder and decoder parameters to perform the last reconstruction based on the autoencoder network and calculates the outlier score based on the reconstruction error.

Since the time complexity of the neural network algorithm depends on the network layer with the maximum number of neurons, so autoencoder requires \( O(\text{epochs} \times |X| \times D^2) \), where \( |X| \) is dataset size, \( D \) is the dimension of the dataset. Step 4 requires \( O(\text{epochs} \times |X| \times d) \) to obtain the outlierness for all data samples based on \( k \) the nearest neighbour distance and require \( O(\text{epochs} \times (|X|\log|X|)) \) scan over the outlierness list to produce the inlier and outlier pseudo set, where \( d \) is the latent dimension size (\( d \) is much smaller than the original data dimension size \( D \)). Overall, the time complexity is \( O(\text{epochs} \times |X| \times D^2) \).

### 4. Experiments

#### 4.1. Dataset

The performance of the SRAE model is evaluated using regression prediction of magnetic material performance improvement. The data are experimental data on the magnetic performance improvement of Nd-Fe-B rare earth magnetic materials in various compounds, processes, and environmental processing conditions. The dataset was provided by Rare Earth Magnetic Materials and Devices of Jiangxi University of Science and Technology. The dataset is denoted as MMdata for short, and it contains 214 data with 41 original features.
Before machine learning analyses, the data collected from the real-world experimental application differs from the publicly available dataset. Preprocessing is a vital part. Data preprocessing includes the steps of removing outliers, filling in missing values, and feature selection. The method discussed in this paper belongs to the outlier detection component, and missing values are filled with median values of the corresponding attribute value. The proposed method in this paper is applied during data preprocessing of the regression task of magnetic material performance prediction. Since there is no outlier label to promote learning, the proportion of outliers is initially set by a practical value, such as 10%.

4.2. Autoencoder network architecture and evaluation

Due to the small amount of data, the network design is relatively made simple to avoid overfitting. In addition to the input layer and output layer, there are three hidden layers, as shown in Figure 1, the number of neurons in each layer is 40, 20, z, 20, 40, and z is the number of neurons in the bottleneck layer, which is finally optimized as 4 through experiments.

We applied ReLU activation function and batch normalization to all layers, except for the last layer. The Adam optimization algorithm was used to select the best model with the lowest loss. The training epochs is set to 1000, to iteratively minimize the objective function and stop training the SRAE network if the loss value does not decrease for 10 consecutive epochs.

Since the MMdata has no outlier labels, the effectiveness of the proposed outlier model is measured by regression prediction results after outlier removal. To date, several representative regression models with excellent performance have been selected for experimental comparison, including neural network models, random forests, and gradient boosted regression trees (GBRT). For each algorithm, the default parameter values are used, except for neural networks, whose settings are dynamically adjusted to achieve optimal performance.

The performance improvement of regression prediction with removing outliers and without removing outliers is compared, and the difference in regression performance with different outlier detection algorithms is compared. Regression performance evaluation uses traditional indicators such as MSE, MAE, and $R^2$. The final evaluation value is the average of the 10-fold cross-validation results of each regression model.

4.3. Experimental result

Machine learning modelling and predictive analytics are implemented in Python 3.6 on Google Colab platform, and the algorithms are implemented with Keras and ScikitLearn open-source package. To evaluate the performance of the proposed method, first, refer to Table 1, the observed regression results obtained by applying our proposed method denoted as SRAE. The traditional reconstruction-based autoencoder method is denoted as TAE with the same architecture and the same hyperparameter settings. BASE represents the results of not using any outlier removal algorithm.

As shown in Table 1, the performance of SRAE and TAE is significantly improved compared to BASE, i.e. the case without outlier removal. Here we take the best performing neural network regression algorithm as an example. In the experiments, a simple three-layer network structure is used, where the hidden nodes are 64, 16, and 1, ReLU is the activation function, Adam is the parameters optimization method, and 0.005 is the learning rate. SRAE outperforms BASE in measuring $R^2$ with performance improvements of 29.8% and 28.2%, respectively, which shows that unsupervised outlier removal using autoencoder is effective and necessary. We can also observe that the SRAE proposed in this paper performs better than the TAE on the three regression algorithms. Incorporating the separation information can adjust the reconstruction error and further separate the outliers.

Table 2 refers to the regression results obtained by applying the other three classical outlier detection algorithms provided by the scikit-learn library. We compare the proposed approach with Elliptic Envelope, Isolation Forest, and Local Outlier Factor (LOF). Overall, SRAE has the best performance, with the best results obtained for all three evaluations. Among the three regression algorithms, Neural Network and GBRT perform better.

4.4. Discussion

In addition to the parameters of the neural network, the weights and biases that are updated through backpropagation, SRAE also has three types of hyperparameters.
Table 1. Comparisons of SRAE, TAE and BASE (without outlier removal).

| Algorithm      | SRAE     | TAE       | BASE       |
|----------------|----------|-----------|------------|
|                | $R^2$    | MSE       | MAE        | $R^2$    | MSE       | MAE        | $R^2$    | MSE       | MAE        |
| Random forest  | 0.8632   | 2.6342    | 1.1797     | 0.8451   | 2.9686    | 1.2462     | 0.6996   | 4.9368    | 1.5076     |
| GBRT           | 0.9021   | 1.8271    | 1.0867     | 0.8718   | 2.4273    | 1.1957     | 0.7976   | 3.1126    | 1.2504     |
| Neural network | **0.9417** | **1.1783** | **0.8694** | 0.9302   | 1.3679    | 0.8851     | 0.7252   | 3.5158    | 1.1591     |

Table 2. Comparisons of classic outlier detection algorithms.

| Algorithm         | SRAE     | Elliptic envelope | Isolation forest | Local outlier factor |
|-------------------|----------|-------------------|------------------|----------------------|
|                   | $R^2$    | MSE               | MAE              | $R^2$    | MSE       | MAE        | $R^2$    | MSE       | MAE        |
| Random forest     | 0.8632   | 2.6342            | 1.1797           | 0.8127   | 3.7424    | 1.3312     | 0.8527   | 2.9424    | 1.2059     |
| GBRT              | 0.9021   | 1.8271            | 1.0867           | 0.8703   | 2.4771    | 1.1291     | 0.8803   | 2.8391    | 1.1134     |
| Neural network    | **0.9417** | **1.1783** | **0.8694** | 0.9143   | 1.7544    | 0.9609     | 0.8209   | 3.1671    | 1.1464     |

Figure 2. Influence of $\lambda$ on the regression’s MAE.

Figure 3. Influence of $z$ on the regression’s MAE in different dimensions.

Figure 4. The influence of ratio $r$ in terms of regression MAE.

4.4.1. Sensitivity of loss function adjustment factor $\lambda$

$\lambda$ is the first hyperparameter to be tuned. The greedy method is used to fix other hyper-parameters to find the optimal value. The number of network bottleneck nodes is initially set by the number of principal components that explain at least 90% of the variance of the data set. For MMdata, the compressed dimension size $z$ is set to 15, and the ratio of outliers, $r$ is set to 0.1 empirically. When $\lambda = 0$, the method becomes the traditional autoencoder method that does not perform separation with regularization, but only accounts for the reconstruction error. When the value of $\lambda$ ranges from 0 to 0.1, the interval is 0.01; when it ranges from 0.1 to 0.5, and the interval is 0.1. Figure 2 shows that $\lambda = 0.08$ produces the lowest MAE in the regression task. Compared to TAE ($\lambda = 0$), the performance is increased by 5.68%. However, when the MAE value for $\lambda \geq 0.2$ decreases, separation regularization may destroy the reconstruction mechanism.

4.4.2. Sensitivity of the latent representation dimension size $z$

To investigate the influence of the number of bottleneck nodes on the outlier detection performance, we train
SRAE with various intermediate dimensions and list the MAE for each regression algorithm, as shown in Figure 3.

We find that setting a different number of neurons in the bottleneck layer has little effect on the final regression performance. The results of MAE on different regression algorithms do not vary much, and the performance of different dimensions does not show regularity. This phenomenon may be due to two main reasons. On the one hand, the change of the compressed space dimension has an impact on the reconstruction error, but on the other hand, it does not have much impact on the outlier score ranking, and the obtained outlier set is basically stable.

The compressed representation of the bottleneck layer is not used to extract features, but to detect outliers. In the case of low dimensions in the latent space, the autoencoder also has enough learning ability to keep outliers away from the normal sample. To reduce the computational effort, we set $z$ to 4 in the following experiment.

### 4.4.3. Sensitivity of the outlier ratio $r$

Since the dataset has no outlier labels, we fine tune the outlier ratio through the optimal performance value of the regression. When the network structure is fixed, and the network parameters as $\lambda = 0.08$, and $z = 4$, can finally be found through the grid search. The minimum value of MAE is the optimal parameter. When $r = 0.1$, the neural network obtains the minimum MAE value 0.08694. In other words, after removing 10% of samples as outliers, the neural network achieves the best regression results (Figure 4).

### 4.4.4. The evolution of reconstruction errors in the training procedure

As shown in Figure 5, in the network update process, the reconstruction error decreases.

Figure 6 presents the KDE (kernel density estimation) reconstruction error distribution for outliers and inliers during training, where orange represents the inlier error...
distribution, and the blue shade representing the outlier error distribution. The distribution is estimated by the Gaussian kernel density function. From Figure 6, we can first observe that the estimated curve of the outlier is always on the right side of the inlier. The reconstruction error will be more significant. Overall, as the training progresses, the reconstruction error of the inner and the outlier shows a decreasing trend, and the inner and the outlier can be further separated in the density distributions. Starting from epoch 50, inliers and outliers have similar distributions, and the overlap area is large. Until epoch $= 500$, inliers can almost be separated. The inlier distribution is more concentrated, and the error range is smaller. Outliers are more diverse and have a more significant error span.

4.4.5. Data distributions in latent space at the learning procedure of SRAE

To verify the impact of the improved algorithm on the latent compressed representation, we set the compression dimension to 2, so that we can visually observe the distribution of outliers by drawing a scatter plot.

Figure 7 displays the outlier distribution over training stages at epochs 50, 100, 200, and 500. When epoch $= 50$, these data points are mixed. The outliers indicated by red dots are mixed with the inlier points indicated by blue dots. When epoch $= 100$, most of the outliers are separated from the inlier points. At 200 epochs, the outliers are almost out of the data centre compared to epoch 100. When epoch $= 500$, the outliers are completely separated from the inliers. The inliers are more concentrated and the outliers are completely at the edges of the dataset, indicating that the compressed features can learn more important information that facilitates data separation through training. The separation learning part updates the network to better separate outliers and inliers.

5. Conclusion

This paper focuses on the outlier removal for experimental material data, which are irregular and unlabelled. Outlier detection, is a part of data preprocessing, and it is particularly important for data mining in real-world material science. We have proposed an unsupervised outlier detection method based on an autoencoder network.

In the task of predicting the properties of magnetic materials, the regression results show that the use of SRAE is always better than other outlier removal algorithms. The proposed SRAE model can simultaneously generate...
a more separable latent representation and a more discriminative error.

In addition, there is an open issue that further needs to be addressed: It is worth searching for a more interpretable deviation degree of outliers in the compressed feature space.

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**Data availability statement**

The data that support the findings of this study are available from the corresponding author, Y C, upon reasonable request.

**References**

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE, 2*(1), 1–18. http://dm.snu.ac.kr/static/docs/TR/SNUDM-TR-2015-03.pdf.

Barnett, V., & Lewis, T. (1994). Outliers in statistical data (3rd ed.). John Wiley & Sons.

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data* (pp. 93–104). ACM. https://doi.org/10.1145/342009.335388

Butler, K. T., Frost, J. M., Skelton, J. M., Svanea, K. L., & Walsh, A. (2016). Computational materials design of crystalline solids. *Chemical Society Reviews, 45*(22), 6138–6146. https://doi.org/10.1039/C5CS00841G

Cabero, I., Epifanio, I., Piérola, A., & Ballester, A. (2021). Archetype analysis: A new subspace outlier detection approach. *Knowledge-Based Systems, 217*, Article 106830. https://doi.org/10.1016/j.knosys.2021.106830

Cheng, Z., Zhu, E., Wang, S., Zhang, P., & Li, W. (2021). Unsupervised outlier detection via transformation invariant autoencoder. *IEEE Access, 9*, 43991–44002. https://doi.org/10.1109/ACCESS.2021.3065838

D’Agostino, M., & Dardanoni, C. (2009). Whats so special about Euclidean distance. *Social Choice and Welfare, 33*(2), 211–233. https://doi.org/10.1007/s00355-008-0353-5

Hampel, F. R. (1974). The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69(346), 383–393. https://doi.org/10.1080/01621459.1974.10482962

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504–507. https://doi.org/10.1126/science.1127647

Hosokawa, H., Calvert, E. L., & Shimojima, K. (2021). Machine learning prediction for magnetic properties of Sm-Fe-N-based alloys produced by melt spinning. *Journal of Magnetism and Magnetic Materials, 526*, Article 167651. https://doi.org/10.1016/j.jmmm.2020.167651

Katsikas, G., Sarafidis, C., & Kioseoglou, J. (2021). Machine learning in magnetic materials. *Physica Status Solidi (b), 258*(8). https://doi.org/10.1002/pssb.202000600

Keller, F., Muller, E., & Bohm, K. (2012). HiCS: High contrast subspaces for density-based outlier ranking. In *IEEE 28th International Conference on Data Engineering* (pp. 1037–1048). IEEE.

Knorr, E. M., & Ng, R. T. (1999). Finding intensional knowledge of distance-based outliers. In *Proceedings of the 25th VLDB Conference* (Vol. 99, pp. 211–222).

Kramer, M. A. (1991). Nonlinear principal component analysis using auto associative neural networks. *AIChE Journal, 37*(2), 233–243. https://doi.org/10.1002/aic.690370209

Leys, C., Ley, C., Klein, O., Bernard, P., & Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology, 49*(4), 764–766. https://doi.org/10.1016/j.jesp.2013.03.013

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *Eighth IEEE International Conference on Data Mining*. IEEE.

Morales-Forero, A., & Bassetto, S. (2019). Case study: A semi-supervised methodology for anomaly detection and diagnosis. In *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 1031–1037). https://doi.org/10.1109/IEEM44572.2019.8978509

Nelson, J., & Sanvito, S. (2019). Predicting the curie temperature of ferromagnets using machine learning. *Physical Review Materials, 3*(3), 104405. https://doi.org/10.1103/PhysRevMaterials.3.104405

Pang, G., Chen, L., Cao, L., & Liu, H. (2018). Learning representations of ultrahigh-dimensional data for random distance-based outlier detection. In *KDD 2018 – Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 2041–2050). Association for Computing Machinery.

Pang, G., Shen, C., Cao, L., & Hengel, A. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys, 54*(2), 1–38. https://doi.org/10.1145/3439950

Pang, G., Ting, K. M., & Albrecht, D. (2015). LeSiNN: Detecting anomalies by identifying least similar nearest neighbours. In *ICDM Workshop* (pp. 623–630). IEEE.

Park, H. K., Lee, J. H., Lee, J., & Kim, S. K. (2021). Optimizing machine learning models for granular NdFeB magnets by very fast simulated annealing. *Scientific Reports, 11*(1), 3792. https://doi.org/10.1038/s41598-021-83315-9

Pevný, T. (2016). Loda: Lightweight on-line detector of anomalies. *Machine Learning, 102*(2), 275–304. https://doi.org/10.1007/s10994-015-5521-0

Raccuglia, P., Elbert, K., Adler, P., Falk, C., Wenny, M. B., Mollo, A., Zeller, M., Friedler, S. A., Schrier, J., & Norquist, A. J. (2016). Machine-learning-assisted materials discovery using failed experiments. *Nature, 533*(7601), 73–76. https://doi.org/10.1038/nature17439

Rosasco, L., Vito, E. D., Caponnetto, A., Piana, M., & Verri, A. (2004). Are loss functions all the same? *Neural Computation, 16*(5), 1063–1076. https://doi.org/10.1162/089976604773135104

Rousseeuw, P. J., & Van Driessen, K. (2000). A fast algorithm for the minimum covariance determinant estimator. *Technometrics, 41*(3), 212–223. https://doi.org/10.1080/00401706.199.10485670

Sakurada, M., & Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis* (p. 4). ACM.

Sendek, A., Yang, Q., Cubuk, E., Duerloo, K.-A., Cui, Y., & Reed, E. (2016). Holistic computational structure screening of...
more than 12,000 candidates for solid lithium-ion conductor materials. *Energy & Environmental Science, 10*(1), 306–320. https://doi.org/10.1039/C6EE02697D

Sugiyama, M., & Borgwardt, K. (2013). Rapid distance-based outlier detection via sampling. In *NeurIPS* (467–475).

Xia, Y., Cao, X., Wen, F., Hua, G., & Sun, J. (2015). Learning Discriminative Reconstructions for Unsupervised Outlier Removal. In *2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1511–1519). https://doi.org/10.1109/ICCV.2015.177

Zhou, C., & Paffenroth, R. C. (2017). Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 665–674). ACM.