Journal of Computer Science

Original Research Paper

Automated Terrain Classification with a Bayesian Hyperparameter Optimized Deep Supervised Autoencoder Model

Tiny Du Toit, Hennie Kruger and Annette Van Der Merwe

School of Computer Science and Information Systems, Faculty of Natural and Agricultural Sciences, North-West University, Private Bag X6001, Potchefstroom, 2520, South Africa

Article history
Received: 22-04-2023
Revised: 28-06-2023
Accepted: 07-07-2023

Corresponding Author:
Tiny Du Toit
School of Computer Science and Information Systems, Faculty of Natural and Agricultural Sciences, North-West University, Private Bag X6001, Potchefstroom, 2520, South Africa
Email: Tiny.DuToit@nwu.ac.za

Abstract: Terrain classification according to specific terrain attributes, has become increasingly important in certain decision-making scenarios. Automated robots are often utilized to traverse a specific surface to collect data that can be used in classification models to identify a specific terrain. In this study, a supervised autoencoder model (i.e., an autoencoder combined with a supervised learner such as a multilayer perceptron) is proposed to perform the classification of different terrains. Furthermore, a Bayes hyperparameter optimization approach is employed to determine optimum hyperparameter values. The dataset used for model building and training was obtained by driving a Lego Mindstorm EV3 mobile robot, fitted with a Raspberry Pi computer and a Sense HAT inertial measurement unit over six different terrain surfaces, i.e., asphalt, dirt, epoxy, grass, paving, and stone surfaces. The final dataset contains 281 232 data points which were used for model building. The results of the proposed supervised autoencoder were compared and contextualized with three other models, i.e., an SVM model, a logistic regression model, and an XGBoost model. Results indicate that it is not only feasible but also desirable to consider the use of a supervised autoencoder model when there is a need for terrain classifications.

Keywords: Bayesian Optimization, Hyperparameter Optimization, Supervised Autoencoder, Terrain Classification

Introduction

Autoencoders (AEs) are simple unsupervised learning-based models which transform model inputs into outputs with the least possible distortion of the inputs (Eddahmani *et al.*, 2023). This type of model is essential in machine learning, although it has a conceptually simple structure. According to Xu and Ren (2022), AEs were first created by Rumelhart *et al.* (1986) to perform backpropagation with the inputs used for supervision and are classified as one of the fundamental paradigms of unsupervised learning. They comprise a central building block of many deep learning approaches which train stacked AEs in an unsupervised bottom-up manner (d'Avila Garcez and Lamb, 2020). To enhance the performance of neural network modeling, a supervised learning architecture is often employed in combination with an AE which is then used to train the final layers and fine-tune the complete neural network architecture (Le *et al.*, 2018). The lower part of the AE is task-agnostic and may also be used in transfer learning approaches. In this study, the relatively new approach of a supervised autoencoder (Jafarzadeh *et al.*, 2021) will be applied to perform terrain classification using a specialized robot built for this purpose. Combining a supervised learner such as a multilayer perceptron and an AE to simultaneously predict inputs and outputs, is a methodology that has been applied by a few researchers (Hanakata *et al.*, 2020; Yang *et al.*, 2021). However, the application of an SAE approach for terrain classification is a fairly new concept with little evidence of these types of applications in the literature.

Intelligent outdoor robots are increasingly used in environments that can be hazardous to human beings, like military reconnaissance, disaster management, or even remote medical examinations (Concon *et al.*, 2021). For obvious reasons, an autonomous robot must adapt to the traversed domain to optimize its planned operation. This will be possible only if a robot is equipped with the necessary tools to perceive its environment and optimally adapt to it. Classification techniques are generally vision, reaction, or hybrid-based, depending on the application for which it is required (Fritz *et al.*, 2023).

Vision-based approaches are typically employed when data collection is done either on-board utilizing cameras or laser range finders, or with high-performance electronic devices such as synthetic aperture radar imaging technology that uses microwave sensing and is not adversely affected by extreme weather and light conditions (Jia *et al.*, 2019). Chavez-Garcia *et al.* (2017) implemented a supervised training approach in a Convolutional Neural Network (CNN) to predict whether a robot will be able to negotiate its way across a terrain based on a single image. Improvements in accuracy were achieved using a combined approach consisting of an unsupervised stacked denoising AE feeding into a supervised learning neural network with backpropagation while optimizing the entire network through error backpropagation (Liang *et al.*, 2017). Vision-based classification often results in three-dimensional maps that indicate navigability, vegetation, etc., and are adequate for predicting specific terrains like gravel, grass, or tar. Mobile robots need to timeously maintain performance throughout their operation, necessitating advanced techniques that will enable dynamic terrain classification for more accurate domain adaptation.

Proprioceptive sensors like Inertial Measurement Units (IMUs) are critical equipment in terrain classification and can measure a vehicle's slip, sinkage, or vibrations in reaction-based terrain classification, with a high degree of accuracy (Fritz *et al.*, 2023). The motivation, collection, and data type are essential factors influencing the choice of algorithms and methods for classifying the terrain and subsequent domain adaptation. IMUs present the option of combining deep learning techniques such as Long Short-Term Memory (LSTM) and CNNs to benefit from both temporal and spatial advantages of reaction-based classification (Concon *et al.*, 2021). Ahmadi *et al.* (2021) investigated a semi-supervised model consisting of a gated Recurrent Neural Network (RNN) that used raw and variable-length time-series data to perform terrain classification.

Challenges identified when performing terrain classification have inspired hybrid-based approaches. Apart from combining different data acquisition techniques like visual and reaction-based surveillance, combinations of classification methods have been developed. Ding *et al.* (2022) mathematically modeled the interaction between a robot's movement sensor and the terrain and adopted a combination of the support vector machine, Gaussian discriminant analysis, logistic regression, and K-nearest neighbor algorithms for classification. Accuracy was, however, not very high for mixed terrain areas due to insufficient characteristic observation. An LSTM trained on time series data from an actuator that measured the difference between a robot's center-of-pressure and leg forces partially addressed this challenge (Allred *et al.*, 2021). Another mixed approach involved combining 2D images with 3D photogrammetric data in a CNN architecture supplemented with a depth pooling layer to create a simulation of the environment (Chen *et al.*, 2021). Varying approaches deliver success and, consequently, bring about some challenges but it is clear that incorporating soft computing techniques into intelligent classification strategies will only improve terrain classification procedures (Nampoothiri *et al.*, 2021).

The motivation for this study is to determine the accuracy with which a Bayesian hyperparameter-optimized deep SAE model can identify the terrain type from IMU sensor data. The main contribution of the study is the development of a supervised autoencoder (combined with a supervised learner such as a multilayer perceptron) that can be utilized for terrain classification purposes. This approach has not been used previously for terrain classification problems. The remainder of the paper is structured as follows. The next two sections present background information on AEs and the Bayesian hyperparameter optimization methodology. The material and methodology section details the data used and the experimental setup while the results section describes the result of the model-building process, the accuracies achieved, and contextualization with three other comparable modeling techniques. The penultimate section of the paper presents a discussion of the proposed model and the modeling process, as well as further insight into the modeling and classification results. The paper is then concluded in the conclusion section with some final remarks.

Autoencoders

An AE is an unsupervised artificial neural network that typically consists of three layers, i.e., an input layer, a hidden layer, and an output layer (Bao *et al.*, 2017). AEs have become a popular way of reducing the dimensionality of large datasets and are often mentioned together with Principal Component Analysis (PCA), which is another technique used for dimensionality reduction (Witanowski *et al.*, 2023). Reducing the number of features in a dataset has specific advantages such as increasing the computational efficiency of a modeling process, removing highly correlated features, eliminating noise in the data, and reducing the baseline drift (Kensert *et al.*, 2021). Both techniques may be used for dimensionality reduction, however, Witanowski *et al.* (2023) argue that although PCA is a popular way of dimensionality reduction, an AE is more effective in reducing the number of features. Abdulhammed *et al.* (2019) list some differences between the two approaches. A fundamental distinction is that an AE can model linear and nonlinear structures, while PCA can only work with linear structures. Other differences include modeling aspects such as run time (PCA has a fast run time and AE, has a slow run time), computational complexity, and memory complexity. Care should also be

taken to prohibit overfitting when making use of an AE. In this study, an AE will be implemented to reduce the dimensionality of a given dataset, therefore, a brief introduction to AE is provided in the subsequent discussion. The objective of an AE is to reconstruct an input vector *x* into an output vector *y* in such a way that the error between the input and reconstructed vectors is minimized. To do this, the AE contains two distinct parts called an encoder function and a decoder function. The encoder maps a given input vector into a compressed form, the bottleneck, in a latent space. The decoder function then maps the latent representation back onto a reconstructed output vector (Abdulhammed *et al.*, 2019). An example of a typical AE is illustrated in Fig. 1.

After the input vector is mapped to the hidden layer, it is reconstructed by mapping the hidden vector to the reconstruction or output layer. These two actions can mathematically be expressed as follows (Bao *et al.*, 2017):

$$a(x) = f\left(W_1 x + b_1\right) \tag{1}$$

$$y = f\left(W_2 a(x) + b_2\right) \tag{2}$$

where, $x, y \in \mathbb{R}^n$ (n is the dimensionality of the input and output layers). The hidden layer is denoted by $a(x)$ while W_1 and W_2 represent the weight of the hidden and output layers, respectively. The biases of these two layers are indicated by b_1 and b_2 respectively, while f is an activation function. To minimize the reconstruction error between the input vector x and the reconstructed output vector y, the following function is optimized:

$$argminW_1, W_2, b_1, b_2 [J] = argminW_1, W_2, b_1, b_2 \left[1/2 \sum_{i=1}^{n} \| x_i - y_i \| + J_{wd} + J_{sp}\right] \tag{3}$$

In this function, J is the squared reconstruction error, and x_i and y_i are the i^{th} value of x and y, respectively. A weighted decay (J_{wd}) and sparse penalty term (J_{sp}) are included in the objective function. These terms are required to avoid overfitting and ensure that the model generalizes effectively. The two terms are formulated as follows:

$$J_{wd} = (1/2)\lambda\left(\|W_1\|_F^2 + \|W_2\|_F^2\right) \tag{4}$$

$$J_{sp} = \beta \sum_{t=1}^{n} KL\left(\rho \| \hat{p}_t\right) \tag{5}$$

where, $\|.\|_f$ is the Frobenius norm. The weight decay and sparse penalty are controlled by parameters λ and β while KL (.) denotes the Kullback-Leibler divergence, a standard function to measure the difference between two distributions (Nielsen, 2022). A sparsity parameter and the average activation of the t^{th} hidden layer are denoted by ρ $\hat{p}_t$ and, respectively. The average activation $\left(\hat{p}_t\right)$ for input i is formulated as follows:

$$\hat{p}_t = (1/m)\sum_{i=1}^{k} a_t\left(x_i\right) \tag{6}$$

where, a_t (x_i) denotes the k^{th} unit of the t^{th} hidden layer.

Although the above description refers to three layers (input, hidden, and output), the AE is referred to as a single layer AE (Bao *et al.*, 2017). A sequence of single-layer AEs may be stacked to form a stacked AE (Bao *et al.*, 2017; Bengio *et al.*, 2006). In a stacked AE, the reconstructed layer of the first single-layer AE is removed and the hidden layer then acts as the input layer for the second single-layer AE. This process is repeated for each layer within the stacked AE so that each subsequent one is the hidden layer of the one preceding it. Bao *et al.* (2017) provides details of a stacked AE, where an example of a five-layer stacked AE is described. There are several other variations of AEs that do not form part of this study. Examples include a sparse AE (Hu *et al.*, 2019), deep AE (Farahnakian and Heikkonen, 2018), denoising AE (Tagawa *et al.*, 2015), under-complete AE (Buongiorno *et al.*, 2019) and variational AE (Liu *et al.*, 2020).

This research models terrain classification, a supervised learning scenario in which the goal is to learn a function for a vector of inputs $x \in \mathbb{R}^d$ to predict a vector of targets $y \in \mathbb{R}^m$ (Le *et al.*, 2018). The function is trained on a finite batch of independent and identically distributed data, (x_1, y_1), …,(x_t, y_t), to accurately predict new samples generated from the same distribution. To perform well in prediction, a typical goal is representation learning, in which the input x_i is transformed into a new representation from which a simple predictor like a linear predictor may be learned. AEs are known for their ability to perform representation learning-see, for example, Yang *et al.* (2022).

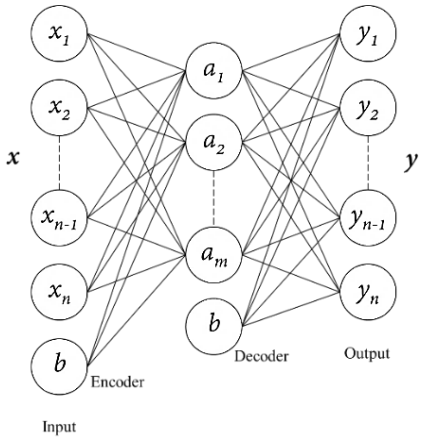

Fig. 1: Typical AE structure

An SAE, which is proposed in this study, is a kind of neural network that jointly predicts inputs (reconstruction) and targets (Le *et al.*, 2018). This essentially implies that a supervised loss is added to the output layer for a single hidden layer, i.e., the supervised loss is introduced to the innermost (lowest) layer of a deep AE. Following the AE training, the layer is typically handed off to the supervised learner. In this study, the AE and supervised learner training is performed simultaneously. By including a supervised loss in the AE, representation learning is more directed toward successful representations for the targeted tasks. In contrast, training a representation purely on supervised tasks, such as learning hidden layers in a neural network, is likely to be an under-constrained problem, yielding solutions that match the data well but do not uncover underlying patterns or generalize well. Thus, the combination of the two losses has the potential to achieve both a balance in terms of extracting underlying structure and accurate prediction performance. Furthermore, it has been demonstrated empirically (Le *et al.*, 2018) that adding the reconstruction error has no adverse effect on performance compared to the matching neural network model. In some instances, it may improve classification accuracy significantly. To be able to evaluate the results in a structured manner and draw reasonable conclusions, the results obtained by three other models are compared to the proposed SAE. The three models used for comparative purposes and which are briefly introduced in the subsequent paragraphs are a Support Vector Machine (SVM), a logistic regression model, and a gradient tree boosting model.

Support Vector Machines (SVMs) are a subset of kernel methods that have been successfully used to classify terrains (Liu *et al.*, 2022). A separating hyperplane is constructed between two classes of points so that the margin between the hyperplane and the points closest to it becomes maximal. Nonlinear classification may be accomplished by first mapping the original data in a nonlinear form to a high-dimensional feature space. Typically, this computation is performed implicitly using a kernel function that defines the dot product between points in a feature space. Allowing for a small number of training errors is also feasible via a so-called soft margin parameter which regularizes the trade-off between maximizing the margin and minimizing the training error.

Logistic regression is an efficient supervised machine learning algorithm used for binary classification problems which can also be generalized to multiclass classification problems. The technique employs a non-linear sigmoidal function and models the probability of a discrete outcome by building, what is generally known as a logit model. This type of model does not require a linear relationship between inputs and outputs and the model's range is bounded to the interval [0,1] (Subasi, 2020). Logistic regression models are widely used by decision-makers to solve classification problems and in the context of this study, the work of Wang *et al.* (2021) stands out as an example where the technique was applied to a terrain classification problem.

Tree boosting is a commonly used and very successful machine-learning method (Chen and Guestrin, 2016). Among the current machine-learning techniques, gradient tree boosting consistently produces excellent results in various applications, including terrain classification (Zhang *et al.*, 2021). Rather than parallelizing the process of decision tree construction, gradient tree boosting obtains predictions in a sequential approach in which each decision tree predicts the error of the preceding tree, therefore boosting (improving) the error (gradient) (Ayyadevara, 2018).

Bayesian Hyperparameter Optimization

Hyperparameters are internal model parameters not determined or learned by a machine learning algorithm, but rather set by the user prior to training (Stuke *et al.*, 2021). They directly influence the performance of a training algorithm and are responsible for the efficiency of machine learning models. Examples of hyperparameters include the number of hidden layers, the number of neurons per layer, the learning rate, and momentum. The best or optimal settings for hyperparameters depend on the size and type of dataset, making the model usually relevant to only one problem. Hyperparameter tuning or optimization involves finding the optimal hyperparameter set that will optimize the performance of a model (Karl *et al.*, 2022). When initializing a machine learning model, it is customary to use a rule-of-thumb approach for the initial values of hyperparameters and improve them through trial and error. However, automated optimization techniques can save a considerable amount of processing time. Popular optimization algorithms include grid search, random search, and Bayesian optimization (Masood and Sherif, 2021).

Algorithm 1: The Bayes optimization algorithm (Guo *et al.*, 2020)

Input: $f(x)$, T, $a(x, H)$
Output: H
$H \leftarrow \theta$
Random initialization of Gaussian processes, calculate $p(f(x)| x, H)$
for $t \leftarrow 1$ *to* T **do**
  $x' \leftarrow argmax_x\, a(x, H)$
  evaluate $y' = f(x')$
  $H \leftarrow H \cup (x', y')$
  Remodel Gaussian processes according to H, calculate $p(f(x)| x, H)$
end for

The grid search approach passes all combinations of hyperparameters through the model and selects the one with the best result. This is an exhaustive and time-consuming technique, especially in models with very large datasets. Random search involves the selection and use of random combinations of hyperparameter values and returning the mix that produced the best result. It is useful when the possible ranges for hyperparameters are relatively large and the method requires less time than grid search. As not all the possible combinations are tested in such a case, the resulting hyperparameter set may not be the optimal one. Both methods evaluate numerous unsuitable combinations without considering the results from previous iterations. In this study, Bayesian optimization is utilized for hyperparameter tuning. The technique is a probabilistic approach that uses Bayesian inference to model the uncertainty in the hyperparameters, rather than relying on other deterministic methods.

Bayesian optimization is an informed method that learns from previous iterations to have an improved subspace in the subsequent iterations (Owen, 2022). The technique consists of two key elements namely a probabilistic surrogate model (i.e., a regression model to create simpler objective functions) and an acquisition function that is used for updating probabilities in a step-by-step manner to gauge how a group of hyperparameters may affect their performance, considering data that has been observed in the past. New hyperparameters that can be experimented with are recommended while the acquisition function determines which subspace should be tested next. The next set of hyperparameters to evaluate is selected based on the posterior distribution, such as by sampling from it. This process is then repeated multiple times until a satisfactory set of hyperparameters is found.

Algorithm 1 presents the pseudo-code for the Bayes optimization approach and is constructed as follows (Guo *et al.*, 2020). Let $f(x)$ be a function that obeys a Gaussian process, then $|\, p\left(f\left(x \right) | x \right)$ is a normal distribution. Furthermore, assume that N experiments are performed and let $H = \left\{ x_n, y_n \right\}_{n=1}^{N}$ represent the training set consisting of the N observations of $f(x)$. The posterior distribution of $f(x)$ is then calculated as $p\left(f\left(x \right) | x, H \right)$. Following the calculation of the posterior distribution, an acquisition function $a(x, H)$ is defined to determine the next sample point by maximizing the acquisition function. These two key elements of the technique are defined in Algorithm 1 in Step 4 (the acquisition function is maximized) and Steps 5-7 (the updating of the posterior distribution).

For further details and a mathematical review of the Bayes optimization technique, the work of (Shahriari *et al.* 2015; Wu *et al*., 2019) may be consulted.

Materials and Methods

An experiment was performed to acquire representative training data and then build the SAE model to accurately classify the terrain on which a purpose-built mobile robot travels. To ensure a proper and valid data-gathering process, a Lego Mindstorm EV3 mobile robot was chosen to collect the required data. Lego Mindstorms (Valk, 2014) is a low-cost robotic platform that has been used for various real-world applications, including data logging (Abdullah *et al.*, 2014), remote control (Chin *et al.*, 2009), navigation (Kwon *et al.*, 2023) and localization (Liu *et al.*, 2018). The mobile robot shown in Fig. 2 was constructed with a Raspberry Pi computer and a Sense HAT IMU (Chatterjee and Debnath, 2018) mounted on top. This IMU included a variety of integrated circuit-based sensors that are suitable for different types of experiments and applications. The mobile robot moved on rubber wheels and is battery-operated.

A robot's functional abilities are highly dependent on its sensing capabilities (Chin *et al.*, 2009). Therefore, the Sense HAT IMU was chosen since it includes six sensors that measure the dynamic performance of the vehicle which is of particular significance in terrain classification problems. These sensors include the following:

- Accelerometer-used to detect the acceleration force acting along the *x*-, *y*- and *z*-axes in a local frame
- Barometer-a pressure sensor that also determines the altitude
- Gyroscope-this sensor detects angular velocity in a local frame along the *x*-, *y*- and *z*-axes
- Humidity-a device that detects and measures the amount of water vapor in the air
- Magnetometer-a sensor that senses the magnetic field of the earth and provides the heading of the sensor; and
- Temperature-used to determine the ambient temperature

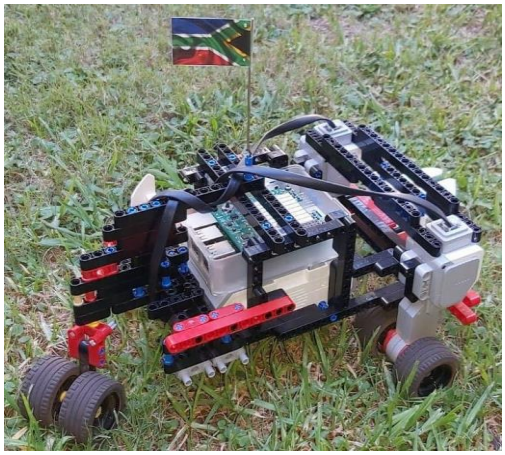

Fig. 2: Lego Mindstorms robot used in the data acquisition experiments

The sensors measured eight quantities, *viz* acceleration intensity, orientation, rotational intensity, relative humidity, humidity-based temperature, magnetic intensity, pressure, and pressure-based temperature. Since the planned data acquisition experiment was conducted on several terrains, of which each composition has different properties and characteristics, it was advantageous to include all the sensor measurements to classify the terrain. In addition, when deep learning models are developed an effective method for improving classification accuracy is to train them on more, or better data (Chollet, 2021; Hasib *et al.*, 2022; 2023).

Data

Data for the model-building experiment was obtained by driving the mobile robot on six different surfaces, i.e., asphalt, dirt, epoxy, grass, paving, and stone surfaces, respectively. These diverse surfaces were selected to improve the robustness of the proposed SAE model. The robot was driven for 30 min on two different samples of each of the six terrains while simultaneously gathering measurements from all the IMU sensors at 13.02 Hz. This driving time and sampling rate resulted in 46872 data points recorded for each terrain. According to a heuristic proposed by Goodfellow *et al.* (2016), a supervised learning algorithm will generally achieve acceptable performance with around 5000 labeled examples per category (terrain). Therefore, the assumption was made that the IMU sensor measurements represented a signature that could be used to classify the terrain accurately. The IMU sensor measurements are described in Table 1.

The complete dataset had 16 inputs (Table 1), six one-hot encoded outputs corresponding to the different terrains, and 281 232 data points with no missing values (the complete dataset is available on request from the corresponding author). An extraction of the dataset is presented in the Appendix. In many practical applications, the pre-processing technique applied to the data will critically affect the performance of the final system (Bishop, 1995). The input values differed by several orders of magnitude, not reflecting their relative importance in determining the terrain type. Linear rescaling is one of the most common forms of pre-processing and is often useful when the typical values for different inputs are significantly diverse. Consequently, all inputs were standardized to a $N(0, 1)$ distribution with like values.

The final step in the pre-processing was to split the available data into three sets: Training, validation, and test sets. This step enables the proposed SAE model to be evaluated by a simple holdout validation protocol (Chollet, 2021). Since the six terrains were equally represented in the available data, the data were randomly partitioned into training (70%), validation (20%), and test (10%) sets, as suggested by Goodfellow *et al.* (2016).

Experimental Setup

The model-building experiments were performed on an IntelR CoreTM i7-7700 CPU with a 3.60GHz processor, 32 GB of RAM, and a 64-bit Windows 10 operating system running Python 3.9.6, TensorFlow 2.5.0 and the Keras 2.5.0 API. In addition, a GeForce GTX 1080 GPU with 8 GB of frame buffering provided accelerated computing. Table 2 shows the hyperparameter lower and upper bounds of the SAE model. These bounds define a search space of 12 096 000 possible candidate architectures. The number of architectures was determined by performing preliminary experiments to determine appropriate bounds and then enumerating all the possible architectures within these bounds.

All the hidden layers used Rectified Linear Unit (ReLU) activation functions. Compared to other activation functions (e.g., sigmoid and tanh), ReLUs significantly accelerate stochastic gradient descent convergence and avoid saturation. A linear activation function was selected for the AE output layer and a softmax activation function for the output layer of the supervised learner i.e., a multilayer perceptron in this study. The softmax function's output represents the class probability associated with each surface. Default Keras parameters (kernel initializer: Glorot uniform, bias initializer: Zeros) were utilized to randomly initialize the model's weights. The Adam adaptive learning rate optimization algorithm (Kingma and Ba, 2014) was used for training the feature-based model offline, utilizing a batch size of 32. To minimize overfitting, a Keras early stopping callback was included (patience of 250 epochs). No regularization was performed as the best model discovered did not overfit due to the relatively small model size. The model-building experiment ran for 284.81 h while conducting 108 Bayesian optimization trials. The best model discovered was then further trained until the best terrain classification accuracy on the validation dataset was established. Finally, the model's performance was evaluated using the test dataset.

Along with the SAE model, an SVM was utilized to learn the separation between each terrain type and all other terrain types (one versus rest classification). Later, an unseen test pattern was assigned to the class with the most significant distance to the decision boundary. In addition to the SAE and SVM models, logistic regression and gradient tree boosting models were also used to model the data.

Table 1: IMU sensor measurements

Input	Sensor	Measuring Unit	Description
acc_x	Accelerometer	m/s^2	Acceleration intensity of the x-axis
acc_y	Accelerometer	m/s^2	Acceleration intensity of the y-axis
acc_z	Accelerometer	m/s^2	Acceleration intensity of the z-axis
pitch	Accelerometer	deg	Orientation
roll	Accelerometer	deg	Orientation
yaw	Accelerometer	deg	Orientation
gyro_x	Gyroscope	rad/s	Rotational intensity of the x-axis
gyro_y	Gyroscope	rad/s	Rotational intensity of the y-axis
gyro_z	Gyroscope	rad/s	Rotational intensity of the z-axis
humidity	Humidity	%	Relative humidity
temp_h	Humidity	°C	Humidity-based temperature
mag_x	Magnetometer	µT	Magnetic intensity of the x-axis
mag_y	Magnetometer	µT	Magnetic intensity of the y-axis
mag_z	Magnetometer	µT	Magnetic intensity of the z-axis
	Pressure	mb	Pressure
temp_p	Pressure	°C	Pressure-based temperature

Table 2: Hyperparameter lower and upper bounds

Hyperparameter	Type	Lower bound	Upper bound
Encoder layers	Integer	1	2
Encoder hidden layer nodes	Integer	1	15
Bottleneck layer nodes	Integer	2	15
Decoder layers	Integer	1	2
Decoder hidden layer nodes	Integer	1	15
MLP hidden layers	Integer	2	3
MLP hidden layer nodes	Integer	1	15
Learning rate (on a logarithmic scale)	Real	10^{-6}	10^{-1}
Epochs	Integer	64	512

Results

The architecture of the best SAE model discovered is shown in Fig. 3. This model has 867 parameters and an optimized learning rate of $8.87.10^{-4}$. Asymmetrical AE was constructed with one hidden layer in the encoder and a decoder with ten nodes each. The bottleneck of the AE has eight nodes. The lower layers of the supervised learner (multilayer perceptron) share the encoder's hidden layer and bottleneck, with 13 and 11 nodes in the two consecutive higher hidden layers.

The Bayesian optimization history in terms of the objective value (validation set accuracy) for the 108 trials is presented in Fig. 4. Trials are numbered starting from 0 and each trial is indicated by a blue dot. Considering the red line, which shows the best objective value determined so far, there was a sharp increase in the validation set accuracy from 26.59-69.32% for the model architecture at the 15^{th} trial. From thereon, three better models were determined within the 108 trials performed. However, continued training of the best model at the 108^{th} trial did not improve the validation set performance. Finally, the third-best model at the 100^{th} trial with a validation set accuracy of 78.83% performed the best with continued training.

Many machine learning methods rely critically on hyperparameter settings for the best results. However, end-users may lack insight into the relative importance of various hyperparameters and their interactions if they rely only on such methods. Hutter *et al*. (2014) described efficient methods that may be utilized to get such insight by fitting random forest models to previously collected data using Bayesian optimization. They presented a unique, linear-time algorithm for computing marginals of random forest predictions. Hutter *et al.* (2014) then demonstrated how to use these predictions inside a functional ANOVA (fANOVA) framework to quantify the importance of single hyperparameters. Figure 5, the importance of the hyperparameters towards the objective value is based on the fANOVA hyperparameter importance evaluation algorithm. The learning rate was the most critical hyperparameter by a large margin compared to the number of neurons in the second hidden layer of the multilayer perceptron (*ffnn_architecture2*). This vital importance of the learning rate is confirmed by Goodfellow *et al.* (2016). In addition, the importance of the hyperparameters regarding the duration of training was also evaluated by the fANOVA framework.

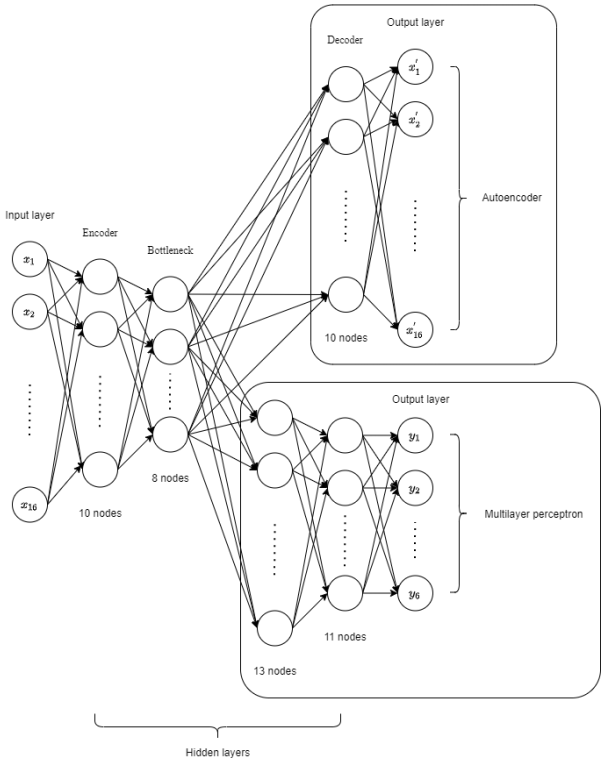

Fig. 3: The supervised autoencoder model architecture

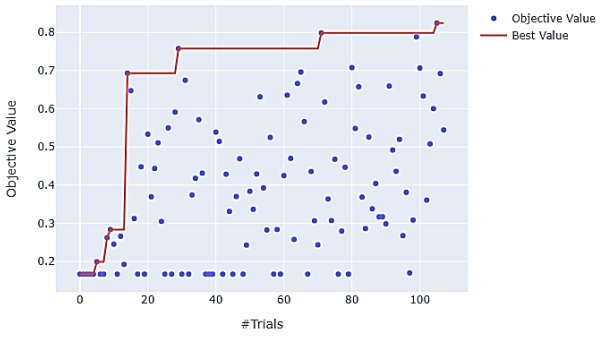

Fig. 4: Bayesian optimization history

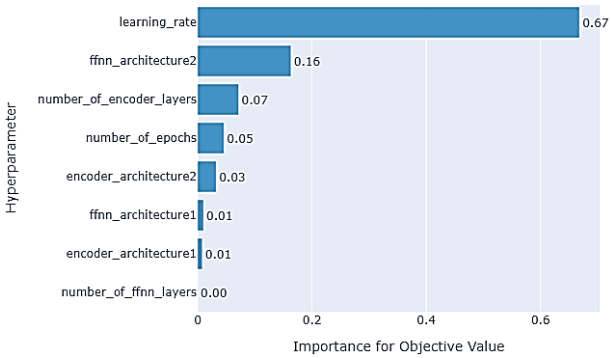

Fig. 5: Importance of the hyperparameters toward the objective value

Table 3: Average accuracy results

Model	Average Accuracy %
SVM model	43.54
Logistic regression	53.94
Supervised autoencoder model	88.62
XGBoost model	95.03

Insight into the importance of hyperparameters and duration time is used to exclude unimportant hyperparameters from optimization, which may dominate the training time. Of all the hyperparameters, it was only the number of epochs that influenced the model training time and consequently, no hyperparameters were excluded in the proposed model.

As alluded to in the section on AEs, comparative modeling results are needed to make meaningful deductions from the results of the proposed model. Three additional models were selected to compare and validate the results of the proposed SAE model. The models chosen are an SVM, a logistic regression model, and an XGBoost model. The detailed modeling process for each of these three models is omitted as it is only the average accuracy of the models that are used for comparison purposes. Table 3, the resulting average accuracy of the proposed SAE, together with the three comparative modeling techniques, is shown. The SVM model performed the worst, with the XGBoost model outperforming the SAE model by 6.41%.

Although the XGBoost model performs best with a relatively small margin, the proposed SAE model performed much better than the SVM model and the logistic regression model. The confusion matrix (to explain the actual and predicted values) for the proposed SAE model is shown in Table 4. The most accurate results of the SAE model were obtained on the paving, epoxy, dirt, and asphalt terrains. The result for the grass surface was also acceptable but for the stone surface, a significant number of predictions were incorrectly indicated as an epoxy surface.

To present a clearer picture of the proposed SAE model, the accuracy, precision, recall, and F1 scores were also calculated for each surface. These scores are summarized in Table 5. The accuracy of a classification system is a class-insensitive performance measure and is defined as the ratio of correctly classified instances to all instances:

$$accuracy = \frac{TP}{TP + FP + FN + TN'} \quad (7)$$

where, Positive (*P*) or Negative (*N*) corresponds to the class labels predicted by the model and True (*T*) or False (*F*) values indicate the model's accuracy (Rivera-Lopez *et al*., 2021).

Table 4: Confusion matrix for the supervised autoencoder model

	Predicted					
Real terrain	Paving	Epoxy	Grass	Dirt	Stone	Asphalt
Paving	4507	0	135	0	0	46
Epoxy	168	4383	52	0	82	3
Grass	226	0	4290	2	84	86
Dirt	0	0	42	4608	38	0
Stone	366	1377	0	16	2613	316
Asphalt	74	0	86	0	2	4526

Table 5: Accuracy, precision, recall, and F1 scores for the supervised autoencoder model

	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
Paving	96.14	84.38	96.14	89.88
Epoxy	93.49	76.09	93.49	83.90
Grass	91.51	93.16	91.51	92.33
Dirt	98.29	99.61	98.29	98.95
Stone	55.74	92.69	55.74	69.62
Asphalt	96.54	90.94	96.54	93.66

Accuracy is a good measure for this dataset since the dataset is balanced (both negative and positive classes have the same number of data instances). Precision and recall performance measurements are class-sensitive indicators that specify how well the evaluated classifier detects the positive class correctly. Precision is defined as the ratio of correctly classified positive cases to all positive instances:

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

The recall rate is the same as the True Positive rate (*TP*). A good classifier's precision and recall should ideally be 100% (high). Therefore, a metric is needed that considers both precision and recall. As a result, the *F*1 score is defined as the harmonic mean of precision and recall (Eq. 9) and becomes 100% only when precision and recall are both 100%:

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{9}$$

The performance metrics in Table 5 complement and confirm the initial results depicted in the confusion matrix. The four accuracy measures for the epoxy terrain are marginally lower than the others while the accuracy of the stone surface is considerably lower than the rest. The classification results computed for the dirt terrain were the most accurate. In general, the results in Tables 4-5 indicate that the implementation of an SAE does indeed contribute satisfactorily to the classification of various terrains. The use of a Bayes optimization hyperparameter technique also contributes to the good accuracy results obtained. In the next section, a more detailed discussion will be offered.

Discussion

The implementation of the proposed technique has provided new contributions and insights into both the modeling approach and the actual results obtained. In the field of terrain classification, the combination of an AE and a supervised learner such as a multilayer perceptron is a first and new attempt to improve and contribute to the classification process of different surfaces.

General Remarks Related to the Proposed Model and Modeling Process

The degree of computational complexity (amount of computing resources required) is an aspect that should be considered when an SAE is employed. The proposed model required a considerable amount of time to train and a significant number of trials were necessary to optimize the modeling process. This, however, is not a disadvantage if one considers the criticality of the application area. Classifying different terrains is not time critical and a suitable model can be developed over a longer period. Once the model is trained, inferences can be made in a very short time.

The introduction of a Bayes optimization method for hyperparameter value determination proved to be more advantageous over traditional approaches such as a grid search. The technique ensures that the "best" hyperparameter values are selected in terms of an objective function that is minimized and therefore contributes to addressing the critical aspect of finding suitable hyperparameter values. Furthermore, the technique facilitates the identification of the most important hyperparameter variables which in turn contributes to a more understandable model. In this study, the learning rate, the number of neurons in the hidden layer of the supervised learner, and the training time were

identified as critical. These findings concur with other related studies in the literature-see for example Goodfellow *et al.* (2016).

The results of the model are easily verifiable and contextualized by comparing it to other classification models. In this study, three other models were used to compare the results of the proposed SAE. An SVM model, a logistic regression model, and an XGBoost model were constructed using the same dataset. The results of these three models were then compared to the proposed SAE. The proposed model performed much better than the SVM and logistic regression model but was marginally worse than the XGBoost model. This may be attributed to the parallel processing capabilities of an XGBoost model. Despite its good performance, it should also be noted that XGBoost models are prone to overfitting, difficult to fine-tune, less interpretable, and difficult to visualize (Jafarzadeh *et al.*, 2021). Although the three models selected for comparative purposes are traditionally easy to train and implement, the SAE model was equally easy to develop and implement. However, the combination with a supervised learner (the multilayer perceptron) does need some initial understanding of the models and concepts involved.

Results of the Proposed Model

The actual results obtained from the SAE model provide further insight into the proposed methodology. To analyze the predicted cases, a confusion matrix was constructed in Table 4. In Table 6, a summary of the percentage of correct classifications for each of the six terrains is shown.

From the results depicted in Table 6, it is clear that the proposed model predicted the dirt terrain with the highest accuracy (98.3%). The worst performance was recorded for the stone terrain where a large number of predictions were incorrectly indicated as epoxy surface. The accuracy for the stone surface was only 55.7%. All the other classification predictions were higher than 91% correct. If the stone terrain is omitted, the confusion matrix shows an average of 95.2% correct classification predictions for the remaining five surfaces. It should be noted that the classification of stone surfaces is traditionally more challenging because of the complexity of a stone terrain. Stones are not similar in shape and often result in uneven surfaces that are difficult to classify. This observation is in line with findings from other studies pertaining to the classification of stone surfaces- see, for example, the work by Fredriksson (2022).

Four different accuracy measures were computed for the proposed model and were summarized in Table 5. Except for the stone surface, all accuracy measures were high with the classification of the dirt surface showing the highest accuracy for all four measures. The results from both the confusion matrix and the computed four accuracy measures are encouraging and indicate the feasibility of employing an SAE in the field of terrain classifications.

Table 6: Correct classifications according to the confusion matrix

Terrain	Number correctly classified	Percentage correctly classified
Paving	4507	96.1
Epoxy	4383	93.5
Grass	4290	91.5
Dirt	4608	98.3
Stone	2613	55.7
Asphalt	4526	96.5

Conclusion

The use of an SAE model for terrain classification, using robots, is increasingly important and may be applied in areas such as disaster management, military reconnaissance, or inspection of hazardous areas. To contribute to the area of terrain classification, an SAE model is proposed in this study. Although the technique is not entirely new, it is a first attempt at terrain classification. As part of the model-building process, a Bayes optimization technique was employed to determine hyperparameter values. The Bayes approach proves to be more advantageous than the traditional hyperparameter identification techniques. The results of the proposed SAE were compared with three other well-known models. Although one of these models has marginally outperformed the proposed model, a closer look at the results and the modeling process indicated that definite benefits can be derived by opting for an SAE model to distinguish among different surfaces. As this was the first attempt at using an SAE in terrain classification, several future research opportunities exist. One such opportunity may be to experiment with the construction of the dataset (drive the robot longer on more surfaces to generate a larger and more reliable dataset) as well as with the bounds of the hyperparameter set (relaxing some of the bounds to further improve the accuracy of the Bayes hyperparameter value optimization process).

Acknowledgment

We gratefully acknowledge the funding received from the Unit of Data Science and Computing (UDSC) at the North-West University in South Africa and the access to their equipment, which was instrumental in facilitating the experimentation and analysis conducted in this study.

Funding Information

Funding for this study was provided by the UDSC.

Author's Contributions

Tiny Du Toit: Design of research project, prepare and perform experiments, development of model architecture, written and finalized the manuscript.

Hennie Kruger: Design of research project, assist with theoretical research, assist with interpretation of results and written and finalized of manuscript.

Annette Van Der Merwe: Research of theoretical concepts and principles used in the research project, written and finalized the manuscript. Assist with language edited and grammatical issues.

Ethics

This article is original work and has not been submitted for publication elsewhere. The authors have read and approved the manuscript.

References

Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M., & Abuzneid, A. (2019). Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics*, *8*(3), 322. https://doi.org/10.3390/electronics8030322

Abdullah, M. A., Ramli, F. R., & Lim, C. S. (2014). Railway Dynamics Analysis Using Lego Mindstorms. *Applied Mechanics and Materials*, *465*, 13-17. https://doi.org/10.4028/www.scientific.net/AMM.465-466.13

Ahmadi, A., Nygaard, T., Kottege, N., Howard, D., & Hudson, N. (2021). Semi-supervised gated recurrent neural networks for robotic terrain classification. *IEEE Robotics and Automation Letters*, *6*(2), 1848-1855. https://doi.org/10.1109/LRA.2021.3060437

Allred, C., Russell, M., Harper, M., & Pusey, J. (2021, November). Improving methods for multi-terrain classification beyond visual perception. In *2021 5th IEEE International Conference on Robotic Computing (IRC)* (pp. 96-99). IEEE. https://doi.org/10.1109/IRC52146.2021.00022

Ayyadevara, V. K. (2018). Gradient boosting machine. *Pro machine learning algorithms: A hands-on approach to implementing algorithms in python and R*, 117-134. https://doi.org/10.1007/978-1-4842-3564-5_6

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS One*, *12*(7), e0180944. https://doi.org/10.1371/journal.pone.0180944

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2006). Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, *19*. https://doi.org/10.5555/2976456.2976476

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press. ISBN: 0198538642.

Buongiorno, D., Camardella, C., Cascarano, G. D., Murciego, L. P., Barsotti, M., De Feudis, I., ... & Bevilacqua, V. (2019, July). An undercomplete autoencoder to extract muscle synergies for motor intention detection. In *2019 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE. https://doi.org/10.1109/IJCNN.2019.8851975

Chatterjee, J., & Debnath, T. (2018). Environmental monitoring using sense hat based on IBM Watson IOT platform. *International Research Journal of Engineering and Technology (IRJET)*, *5*(7), 392-399. https://doi.org/10.13140/RG.2.2.30943.56484

Chavez-Garcia, R. O., Guzzi, J., Gambardella, L. M., & Giusti, A. (2017). Image classification for ground traversability estimation in robotics. In *Advanced Concepts for Intelligent Vision Systems: 18th International Conference, ACIVS 2017, Antwerp, Belgium, September 18-21, 2017, Proceedings 18* (pp. 325-336). Springer International Publishing. https://doi.org/10.1007/978-3-319-70353-4_28

Chen, M., Feng, A., Hou, Y., McCullough, K., Prasad, P. B., & Soibelman, L. (2021). Ground material classification for UAV-based photogrammetric 3D data A 2D-3D Hybrid Approach. *arXiv preprint arXiv:2109.12221*. https://doi.org/10.48550/arXiv.2109.12221

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). https://doi.org/10.1145/2939672.2939785

Chin, K. C. Y., Buhari, S. M., & Ong, W. H. (2009, February). Impact of lego sensors in remote controlled robot. In *2008 IEEE International Conference on Robotics and Biomimetics* (pp. 1777-1782). IEEE. https://doi.org/10.1109/ROBIO.2009.4913271

Chollet, F. (2021). Deep Learning with Python, 2nd Ed. ISBN: 9781617296864.

Concon, M., Wong, W. K., Juwono, F. H., & Apriono, C. (2021). Deep Learning for Terrain Surface Classification: Vibration-based Approach. In *ISIC* (pp. 237-243). https://ceur-ws.org/Vol-2786/

d'Avila Garcez, A., & Lamb, L. C. (2020). Neurosymbolic AI: The 3rd wave. *arXiv e-prints*, arXiv-2012. https://doi.org/10.48550/arXiv.2012.05876

Ding, L., Xu, P., Li, Z., Zhou, R., Gao, H., Deng, Z., & Liu, G. (2022). Pressing and rubbing: Physics-informed features facilitate haptic terrain classification for legged robots. *IEEE Robotics and Automation Letters*, *7*(3), 5990-5997. https://doi.org/10.1109/LRA.2022.3160833

Eddahmani, I., Pham, C. H., Napoléon, T., Badoc, I., Fouefack, J. R., & El-Bouz, M. (2023). Unsupervised learning of disentangled representation via auto-encoding: A survey. *Sensors*, *23*(4), 2362. https://doi.org/10.3390/s23042362

Farahnakian, F., & Heikkonen, J. (2018, February). A deep auto-encoder based approach for intrusion detection system. In *2018 20th International Conference on Advanced Communication Technology (ICACT)* (pp. 178-183). IEEE. https://doi.org/10.23919/ICACT.2018.8323688

Fredriksson, E. (2022). Classification of Terrain Roughness from Nationwide Data Sources Using Deep Learning. http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-185560

Fritz, L., Hamersma, H. A., & Botha, T. R. (2023). Off-road terrain classification. *Journal of Terramechanics*, *106*, 1-11. https://doi.org/10.1016/j.jterra.2022.11.003

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press. ISBN: 0262337371.

Guo, H., Zhuang, X., & Rabczuk, T. (2020). Stochastic analysis of heterogeneous porous material with modified neural architecture search (NAS) based physics-informed neural networks using transfer learning. *arXiv Preprint arXiv:2010.12344*. https://doi.org/10.48550/arXiv.2010.12344

Hanakata, P. Z., Cubuk, E. D., Campbell, D. K., & Park, H. S. (2020). Forward and inverse design of kirigami via supervised autoencoder. *Physical Review Research*, *2*(4), 042006. https://doi.org/10.1103/PhysRevResearch.2.042006

Hasib, K. M., Islam, M. R., Sakib, S., Akbar, M. A., Razzak, I., & Alam, M. S. (2023). Depression Detection From Social Networks Data Based on Machine Learning and Deep Learning Techniques: An Interrogative Survey. *IEEE Transactions on Computational Social Systems*. https://doi.org/10.1109/TCSS.2023.3263128

Hasib, K. M., Tanzim, A., Shin, J., Faruk, K. O., Al Mahmud, J., & Mridha, M. F. (2022). Bmnet-5: A novel approach of neural network to classify the genre of bengali music based on audio features. *IEEE Access*, *10*, 108545-108563. https://doi.org/10.1109/ACCESS.2022.3213818

Hu, C., Wu, X. J., & Shu, Z. Q. (2019). Discriminative feature learning via sparse autoencoders with label consistency constraints. *Neural Processing Letters*, *50*, 1079-1091. https://doi.org/10.1007/s11063-018-9898-1

Hutter, F., Hoos, H., & Leyton-Brown, K. (2014). An efficient approach for assessing hyperparameter importance. In *International Conference on Machine Learning* (pp. 754-762). PMLR. https://doi.org/10.5555/3044805.3044891

Jafarzadeh, H., Mahdianpari, M., Gill, E., Mohammadimanesh, F., & Homayouni, S. (2021). Bagging and boosting ensemble classifiers for classification of multispectral, hyperspectral and PolSAR data: A comparative evaluation. *Remote Sensing*, *13*(21), 4405. https://doi.org/10.3390/rs13214405

Jia, Z., Guangchang, D., Feng, C., Xiaodan, X., Chengming, Q., & Lin, L. (2019). A deep learning fusion recognition method based on SAR image data. *Procedia Computer Science*, *147*, 533-541. https://doi.org/10.1016/j.procs.2019.01.229

Karl, F., Pielok, T., Moosbauer, J., Pfisterer, F., Coors, S., Binder, M., ... & Bischl, B. (2022). Multi-Objective Hyperparameter Optimization-An Overview. *arXiv Preprint arXiv:2206.07438*. https://doi.org/10.48550/arXiv.2206.07438

Kensert, A., Collaerts, G., Efthymiadis, K., Van Broeck, P., Desmet, G., & Cabooter, D. (2021). Deep convolutional autoencoder for the simultaneous removal of baseline noise and baseline drift in chromatograms. *Journal of Chromatography A*, *1646*, 462093. https://doi.org/10.1016/j.chroma.2021.462093

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv Preprint arXiv:1412.6980*. https://doi.org/10.48550/arXiv.1412.6980

Kwon, Y., Kim, W., & Jung, I. (2023). Neural Network Models for Driving Control of Indoor Autonomous Vehicles in Mobile Edge Computing. *Sensors*, *23*(5), 2575. https://doi.org/10.3390/s23052575

Le, L., Patterson, A., & White, M. (2018). Supervised autoencoders: Improving generalization performance with unsupervised regularizers. *Advances in neural Information Processing Systems*, *31*. https://doi.org/10.5555/3326943.3326954

Liang, P., Shi, W., & Zhang, X. (2017). Remote sensing image classification based on stacked denoising autoencoder. *Remote Sensing*, *10*(1), 16. https://doi.org/10.3390/rs10010016

Liu, W., Li, R., Zheng, M., Karanam, S., Wu, Z., Bhanu, B., ... & Camps, O. (2020). Towards visually explaining variational autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8642-8651). https://doi.org/10.1109/CVPR42600.2020.00867

Liu, G., Wang, L., Liu, D., Fei, L., & Yang, J. (2022). Hyperspectral image classification based on non-parallel support vector machine. *Remote Sensing*, *14*(10), 2447. https://doi.org/10.3390/rs14102447

Liu, Y., Fan, R., Yu, B., Bocus, M. J., Liu, M., Ni, H., ... & Mao, S. (2018, December). Mobile robot localisation and navigation using lego nxt and ultrasonic sensor. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 1088-1093). IEEE. https://doi.org/10.1109/ROBIO.2018.8665350

Masood, A. (2021). *Automated Machine Learning: Hyperparameter optimization, neural architecture search, and algorithm selection with cloud platforms*. Packt Publishing Ltd. ISBN: 1800565526.

Nampoothiri, M. H., Vinayakumar, B., Sunny, Y., & Antony, R. (2021). Recent developments in terrain identification, classification, parameter estimation for the navigation of autonomous robots. *SN Applied Sciences*, *3*, 1-14. https://doi.org/10.1007/s42452-021-04453-3

Nielsen, F. (2022). The Kullback–Leibler divergence between lattice Gaussian distributions. *Journal of the Indian Institute of Science*, *102*(4), 1177-1188. https://doi.org/10.1007/s41745-021-00279-5

Owen, L. (2022). *Hyperparameter Tuning with Python: Boost Your Machine Learning Model's Performance Via Hyperparameter Tuning*. Packt Publishing Limited. ISBN: 9781803241944.

Rivera-Lopez, R., Canul-Reich, J., Mezura-Montes, E., & Cruz-Chávez, M. A. (2022). Induction of decision trees as classification models through metaheuristics. *Swarm and Evolutionary Computation*, *69*, 101006. https://doi.org/10.1016/j.swevo.2021.101006

Rumelhart, D. E., McClelland, J. L., & PDP Research Group, C. O. R. P. O. R. A. T. E. (Eds.). (1986). *Parallel distributed processing: Explorations in the microstructure of cognition, Vol. 1: Foundations*. MIT press. https://dl.acm.org/doi/abs/10.5555/104279

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. (2015). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148-175. https://doi.org/10.1109/JPROC.2015.2494218

Stuke, A., Rinke, P., & Todorović, M. (2021). Efficient hyperparameter tuning for kernel ridge regression with Bayesian optimization. *Machine Learning: Science and Technology*, *2*(3), 035022. https://doi.org/10.1088/2632-2153/abee59

Subasi, A. (2020). *Practical machine learning for data analysis using python*. Academic Press. ISBN:0128213809.

Tagawa, T., Tadokoro, Y., & Yairi, T. (2015, February). Structured denoising autoencoder for fault detection and analysis. In *Asian conference on machine learning* (pp. 96-111). PMLR. https://doi.org/10.1016/j.ress.2021.107864

Valk, L. (2014). *The lego mindstorms EV3 discovery book: A beginner's guide to building and programming robots*. No Starch Press. ISBN: 1593275323.

Wang, M., Ye, L., & Sun, X. (2021). Adaptive online terrain classification method for mobile robot based on vibration signals. *International Journal of Advanced Robotic Systems*, *18*(6), https://doi.org/10.1177/17298814211062035

Witanowski, Ł., Ziółkowski, P., Klonowicz, P., & Lampart, P. (2023). A hybrid approach to optimization of radial inflow turbine with principal component analysis. *Energy*, *272*, 127064. https://doi.org/10.1016/j.energy.2023.127064

Wu, J., Chen, X. Y., Zhang, H., Xiong, L. D., Lei, H., & Deng, S. H. (2019). Hyperparameter optimization for machine learning models based on Bayesian optimization. *Journal of Electronic Science and Technology*, *17*(1), 26-40. https://doi.org/10.11989/JEST.1674-862X.80904120

Xu, X., & Ren, W. (2022). A hybrid model of stacked autoencoder and modified particle swarm optimization for multivariate chaotic time series forecasting. *Applied Soft Computing*, *116*, 108321. https://doi.org/10.1016/j.asoc.2021.108321

Yang, S., Wang, Y., & Li, C. (2021). Wind turbine gearbox fault diagnosis based on an improved supervised autoencoder using vibration and motor current signals. *Measurement Science and Technology*, *32*(11), 114003. https://doi.org/10.1088/1361-6501/ac0741

Yang, Z., Xu, B., Luo, W., & Chen, F. (2022). Autoencoder-based representation learning and its application in intelligent fault diagnosis: A review. *Measurement*, *189*, 110460. https://doi.org/10.1016/j.measurement.2021.110460

Zhang, Y., Ma, Z., Song, X., Wu, J., Liu, S., Chen, X., & Guo, X. (2021). Road Surface Defects Detection Based on IMU Sensor. *IEEE Sensors Journal*, *22*(3), 2711-2721. https://doi.org/10.1109/JSEN.2021.3135388

Appendix

In this appendix section, the first ten data points of the training dataset are presented. The provided data points proffer a preliminary insight into the characteristics and scope of the data under analysis.

Appendix 1: Extract of first ten records of data file used for model development

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}
1	1.81E-01	5.05E-01	-6.20E-01	2.06E-01	3.42E-01	-1.29E+00	-1.30E+00	1.03E+00	-1.89E-01	4.32E-01	2.90E-01	2.22E-01	5.53E-01	-9.55E-02	7.72E+03	5.93E+02
2	-1.97E-01	1.06E-01	-1.74E-01	2.13E-01	3.42E-01	-1.34E-01	-4.17E-01	-2.58E-03	7.69E-01	-4.84E-01	-4.55E-01	-3.87E-02	1.78E-01	3.80E-01	1.59E+04	1.22E+03
3	-2.05E-01	1.40E-01	-1.62E-01	2.19E-01	3.38E-01	-1.69E-01	-4.96E-01	-2.04E-02	8.69E-01	7.01E-01	1.00E+00	-9.60E-01	-1.39E-01	-5.14E-01	1.47E+04	1.12E+03
4	2.96E-02	3.70E-02	8.47E-01	2.05E-01	3.54E-01	-1.78E+00	7.84E-01	8.73E-01	6.67E-02	1.34E-01	-6.16E-01	2.42E-01	1.85E-01	-6.00E-01	1.61E+04	1.23E+03
5	1.39E-01	3.95E-02	-2.94E-01	2.06E-01	3.47E-01	-5.74E-01	-6.32E-01	1.75E+00	-3.00E-01	3.30E-01	2.31E-01	3.91E-01	4.71E-01	-4.28E-02	1.30E+04	9.96E+02
6	1.27E-01	-7.69E-04	1.61E+00	2.24E-01	3.12E-01	-8.42E-02	1.13E+00	-4.25E-01	-1.43E+00	-1.03E+00	1.90E+00	-1.06E+00	8.08E-01	2.25E+00	3.17E+03	2.44E+02
7	2.77E-01	3.42E-01	-1.49E+00	1.96E-01	3.16E-01	1.52E+00	-7.96E-01	1.16E+00	5.11E-01	-3.33E-02	2.39E-01	4.47E-01	3.40E-02	7.44E-01	1.62E+04	1.24E+03
8	4.33E-02	7.71E-01	6.15E-01	2.25E-01	3.15E-01	6.25E-01	7.09E-01	1.96E-01	-2.22E-01	-3.43E-01	2.05E-02	1.57E+00	8.11E-01	4.08E-01	6.36E+03	4.88E+02
9	2.15E-01	4.60E-01	-4.57E-01	1.91E-01	3.27E-01	-1.34E+00	-1.27E+00	1.26E+00	-4.94E-01	2.30E-01	1.35E+00	2.35E-01	7.77E-01	7.75E-01	1.47E+04	1.12E+03
10	2.20E-02	6.01E-01	-6.68E-01	1.52E-01	2.90E-01	-5.90E-01	-4.62E-01	-4.74E-01	1.25E+00	7.63E-02	2.86E-01	4.10E-01	1.78E+00	3.11E-01	4.84E+03	3.71E+02

Appendix 2: Extract of one-hot coded terrain classification variable (six terrain surfaces)

	y_1	y_2	y_3	y_4	y_5	y_6
1	1	0	0	0	0	0
2	1	0	0	0	0	0
3	1	0	0	0	0	0
4	0	1	0	0	0	0
5	0	1	0	0	0	0
6	0	0	0	1	0	0
7	0	0	1	0	0	0
8	0	0	1	0	0	0
9	1	0	0	0	0	0
10	0	0	1	0	0	0