CNN-Based Multi-Output and Multi-Task Regression for Supershape Reconstruction from 3D Point Clouds

Hassnae Remmach1[1], Siti Fatimah Abdul Razak*[2], Arif Ullah[3], Sumendra Yogarayan[4] Md Shohel Sayeed[5,] Amine Mrhari[6]
[1]Computer Systems Engineering Laboratory, Cadi Ayyad University, 40000, Marrakesh, Morocco
[2345]Faculty of Information Science and Technology, Multimedia University, Ayer Keroh 75450 Malaysia
[6]Research in Computer Science Laboratory Faculty of Sciences, Ibn Tofail University Kenitra, Morocco
E-mail: fatimah.razak@mmu.edu.my, arifullahms88@gmail.com
* Corresponding Author

Keywords: 3D reconstruction, CNNs, multi-output regressor, multi-target regressor, 3D point cloud, super shapes, point net

Received: August 5, 2024

In a number of industries, including computer graphics, robotics, and medical imaging, three-dimensional reconstruction is essential. In this research, a CNN-based Multi-output and Multi-Task Regressor with deep learning capabilities is proposed for three-dimensional object reconstruction from 3D point cloud. Our approach is grounded in the original Point Net architecture, which addresses the difficulties associated with convolution when applied to point clouds. Firstly, this paper is modified using a Multi-Output Regressor to accurately recreate Super forms from 3D point clouds. Using this method, we first extract features from the 3D point cloud using Point Net. After that, a Multi-Output Regressor receives these data and uses them to anticipate the Super shape parameters needed to reconstruct the shape. Taking in the data, the Multi-Output Regressor retrieved characteristics from Point Net and simultaneously predicts several outcomes. Second, a Multi-Task Regressor is used to modify the Point Net. The network gains from the capacity to transfer knowledge from one task to another, improving the model's overall performance. The model would forecast the ten parameters needed to create the shape in the case of rebuilding Super shapes. The test findings were better than expected; they are intriguing in terms of prediction accuracy and cost, and they update the result by 80%, which is a good accomplishment for the study.

Povzetek: Raziskava predlaga večizhodni in večopravilni regresor na osnovi CNN za rekonstrukcijo 3D oblike iz oblaka točk.

1 Introduction

1.1 Importance and challenges of three-dimensional reconstruction

Three-dimensional reconstruction is an essential task in computer vision, as it enables us to create accurate 3D models of objects and scenes from different data. These 3D models can be used for a wide range of applications, including object recognition, tracking, localization, and segmentation. For instance, in object recognition, 3D models can have used to create a database of 3D models of objects, allowing a computer vision system to recognize objects based on their 3D shape, rather than just 2D appearance [1]. Additionally, in tracking, 3D models can be used to estimate the position and orientation of an object in 3D space, enabling the system to track the object even when it is partially occluded or when the viewpoint changes. Overall, 3D reconstruction plays a crucial role in advancing computer vision applications, and it has the potential to enhance the accuracy and efficiency of various computer vision systems. However, 3D reconstruction is a complex and challenging task and it requires accurate estimation of the camera parameters, feature detection, matching, and triangulation. These challenges increase when dealing with noisy, incomplete, or low-quality data [2].

1.2 Evolution of techniques

To overcome these challenges, researchers have proposed various techniques to improve the accuracy and efficiency of three-dimensional reconstruction [3]. Recently, deep learning approaches have gained significant attention in the field of three-dimensional reconstruction. Artificial Intelligence (AI) algorithms such as Convolutional Neural Networks (CNNs) [4], Generative Adversarial Networks (GANs) [5], and Recurrent Neural Networks (RNNs) [6] have shown promising results in solving different problems in three-dimensional reconstruction, such as feature detection, depth estimation, and shape completion. These techniques have the potential to revolutionize the way we reconstruct 3D models. However, there are still many challenges that need to be addressed, such as data acquisition, training data availability, and computational efficiency. Therefore, it is essential to do more research in the field of three-dimensional reconstruction, especially using deep learning approaches. More efforts are needed to develop new algorithms, improve the accuracy and efficiency of existing techniques, and explore new

applications of three-dimensional reconstruction. By doing so, we can overcome the current limitations and challenges and unlock the full potential of three-dimensional reconstruction in various fields [7].

1.3 Proposal of research contribution

Our research is aligned with this overarching theme. We focus on modeling 3D objects using a parametrized surface, specifically Super shapes [8]. These Supershapes are versatile geometric shapes capable of describing a wide array of complex objects. However, accurately estimating the parameters of Supershapes poses a significant challenge. Traditional methods often rely on manual parameter tuning and handcrafted features, which are labor-intensive and may not yield precise results. To address these limitations, we propose a novel three-dimensional reconstruction approach leveraging Convolutional Neural Networks (CNNs). Our method utilizes a CNN-based Regressor network that takes 3D point clouds as input and accurately estimates Supershape parameters. In this architecture, convolutional and pooling layers are stacked multiple times before fully connected layers. The pooling layers reduce the spatial dimensionality of features, while the convolutional layer's extract features from the input data. These learned features are then mapped to output values through the fully connected layers. Furthermore, we leverage the Point Net [9] network as the foundation for our architecture. While PointNet has demonstrated effectiveness in processing 3D point clouds, its original design may not be directly applicable to the task of 3D reconstruction. PointNet is primarily designed for point cloud classification and segmentation, lacking specific capabilities for regression tasks required for parameter estimation in 3D shapes or objects. To address this limitation, we adapt the PointNet network to learn the correlation between input point clouds and output parameters. This involves modifying the network architecture and training process to incorporate regression-based objectives. By adjusting PointNet to learn this correlation, it can effectively serve as a CNN-based Regressor network for 3D reconstruction tasks. This modification enables accurate and efficient estimation of parameters for 3D shapes and objects, significantly improving upon traditional 3D reconstruction techniques [10-12].

1.4 Outline of the paper

This paper is divided into four main sections.

The first section provides an overview of deep learning techniques for 3D reconstruction and object modeling, focusing on recent advancements and the impact of Convolutional Neural Networks (CNNs). Various approaches, including volumetric and point cloud-based methods, are discussed. The second section details our proposed methodology, emphasizing the architecture and training process of the CNN-based Regressor network. We highlight how our approach addresses limitations of traditional techniques, enhancing accuracy and efficiency in estimating Supershape parameters. Moving to the third part, we present experimental results, evaluating our approach through metrics like accuracy and precision and comparing them with prior work. Finally, the conclusion summarizes our contributions, acknowledges significance, outlines potential and limitations, and suggests areas for future improvement.

2 Literature review

Deep learning techniques have revolutionized the field of 3D reconstruction, enabling more accurate and efficient reconstruction of complex objects and scenes. The evolution of deep learning techniques for 3D reconstruction can be traced back to the use of CNNs for 3D shape classification and segmentation. CNNs have shown promising results in processing 3D shapes by considering them as volumetric representations, and have been used for many applications such as object detection, recognition, and pose estimation [13]. Recent advancements in deep learning have yielded various specialized architectures for 3D reconstruction, tailored to different applications and data types. One such example is Point Net [14], designed specifically for processing point cloud data. Unlike traditional methods relying on structured data like meshes or voxels, PointNet directly processes unstructured point clouds, making it ideal for tasks like 3D reconstruction from LiDAR data. Its versatility lies in its capacity to handle point clouds of varying density, distribution, and orientation. Notably, PointNet has demonstrated effectiveness in processing large-scale point cloud datasets, facilitating the reconstruction of intricate scenes and objects. Its ability to accommodate inputs of different sizes and extract both local and global geometric features has rendered it popular, particularly in fields like autonomous driving and robotics, significantly enhancing the accuracy and efficiency of 3D reconstruction tasks. VoxNet, as described in [15], is a specialized deep learning model tailored for 3D object detection and recognition. By processing 3D voxel grids with 3D convolutions, it excels in detecting objects, even in occluded or complex scenes. Its flexibility to handle various voxel grid sizes and resolutions makes it versatile for different applications, particularly in autonomous driving and robotics, where accuracy is crucial. OctNet presented in [16], is a deep learning framework designed for processing 3D data using an octree structure, enhancing the management of complex 3D scenes. Its hierarchical approach allows for efficient processing, with higher-resolution octree cells corresponding to smaller regions of the scene. This methodology has proven effective for tasks like object detection, segmentation, and 3D reconstruction from sparse data. OctNet is applied in various fields including robotics, virtual reality, and medical imaging, owing to its adaptability to variable-resolution inputs. Additionally, it demonstrates high accuracy in detecting, segmenting, and reconstructing objects from sparse data, solidifying its role in computer vision applications. 3D U-Net, described in [17], is a tailored version of the U-Net architecture crafted for processing 3D volumetric data, notably in medical imaging like MRI and CT scans. It utilizes an encoder-decoder structure with skip connections to effectively

process volumetric data. The encoder down samples input data to capture high-level features, while the decoder up samples to generate output segmentation or reconstruction. Notably, 3D U-Net accommodates variable-sized inputs, rendering it apt for large medical image datasets. MVCNN, described in [18], processes multiple 2D images to create 3D reconstructions. Using a CNN, it extracts features from each image, aggregates them, and generates the final reconstruction. It is versatile for object recognition and reconstruction in robotics and AR, handling varying input images effectively. MVCNN is highly accurate and widely preferred in computer vision applications. Recent advancements in deep learning have introduced Graph Convolutional Networks (GCNs) and Generative Adversarial Networks (GANs) as potent tools for 3D reconstruction tasks. GCNs, detailed in [19], specialize in processing intricate 3D structures like molecular configurations, exhibiting promising outcomes in constructing lifelike 3D models from input point clouds. Meanwhile, GANs, outlined in [20] excel in synthesizing realistic 3D models from either 2D images or partial 3D data. Their potential to transform the 3D reconstruction domain lies in their ability to generate high-quality 3D models even from limited or incomplete datasets. In addition to these advancements, the integration of deep learning techniques [21] with traditional 3D reconstruction techniques such as stereo vision, Structure from Motion (SfM), and Multi-View Stereo (MVS) has led to significant improvements in the accuracy and efficiency of 3D reconstruction. For example, deep learning techniques have been used for refining and post-processing 3D reconstructions generated by SfM and MVS techniques, and for integrating depth and color information for more accurate 3D reconstruction. Overall, the recent advancements in deep learning techniques have significantly improved the accuracy and efficiency of 3D reconstruction and have the potential to revolutionize the field of 3D reconstruction by enabling the reconstruction of complex and diverse 3D structures from limited or incomplete data [22]. Table 1 present the summary of related work.

Table 1: Summary of related work

Technique name	Data set	Result	Future work
Deep learning	LiDAR data	78%	Need improvement in result
PointNet	LiDAR data	74%	3D reconstruction tasks
CNN	synthetic 3D shapes	77%	3D reconstruction tasks
U-Net	synthetic 3D shapes	79%	Need improvement in result
SfM, and MVS	LiDAR data	75%	Need improvement in result

3 Proposed methodology

This section provides a detailed presentation of the proposed methodological approach for three-dimensional object reconstruction. It explores the fundamental techniques used in our methodology, including modeling objects in 3D using Supershapes, the PointNet architecture adapted to this specific task, and multiple regression by neural networks.

3.1 Maintaining a 3D Object by supershapes

SuperShapes, introduced in [23], extend the capabilities of traditional geometric shapes by offering a parametric approach to modeling complex forms. By adjusting parameters such as symmetry (m, M), scaling (a, b), and shape coefficients ($n_1, n_2, n_3, N_1, N_2, N_3$), a diverse range of shapes can be generated, allowing for the representation of complex objects and surfaces in three-dimensional space. The Supershape formula, see Eq. (1), encapsulates these parameters, enabling the creation of customized geometries tailored to specific applications.

$$S = [a, b, m, M, n_1, n_2, n_3, N_1, N_2, N_3] \quad (1)$$

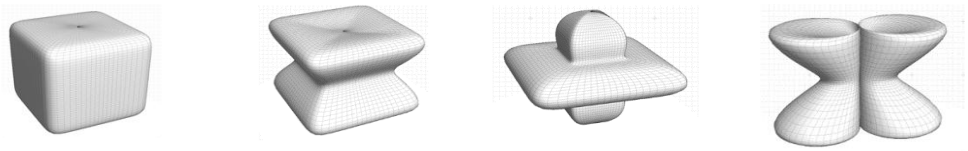
m: Symmetry in the radial direction

M: Symmetry in the tangential direction

a, b: The scaling parameters that control the overall size of the shape.

$n_1, n_2, n_3, N_1, N_2, N_3$: The shape parameters coefficients [24].

By manipulating these parameters (as shown in Figure 1 and Table2), it is possible to create complex and interesting shapes that cannot be easily generated using traditional geometric shapes alone.



Shape (1) Shape (2) Shape (3) Shape (4)

Figure 1: Variety of SuperShapes

Table 2: Super shapes parameters

	m	M	n_1	n_2	n_3	N_1	N_2	N_3
Shape (1)	4	4	10	10	10	10	10	10
Shape (2)	4	4	10	10	10	1.10	10	1.70
Shape (3)	4	4	10	10	10	-2	10	4.20
Shape (4)	4	4	10	-10	10	2	25	3

3.2 The PointNet architecture

PointNet is a deep learning architecture for processing 3D point clouds. It was introduced in 2017 [25] and has since become one of the most widely used architectures for point cloud analysis tasks such as classification, segmentation, and object detection. The Point Net architecture [26] takes as input a set of unordered point clouds and learns a set of features that can be used to classify or segment the objects in the scene. The key innovation of PointNet is that it directly operates on the point clouds without the need for pre-processing or feature engineering, which has traditionally been a time-consuming and error-prone process. The original PointNet architecture consists of several layers of fully connected neural networks, which are used to learn features from the input point clouds. The architecture also includes a max-pooling layer, which is used to aggregate information from all the points in the cloud and generate a global feature vector that can be used for classification or segmentation. The network is trained using a loss function that penalizes misclassifications or segmentation errors [27].

3.3 Multiple regression by neural networks

Multiple regression by neural networks [28] involves using neural network algorithms to perform regression analysis on multiple independent variables simultaneously. These networks are capable of modeling complex non-linear relationships between inputs and outputs, and can handle large amounts of data. By leveraging the inherent ability of neural networks to model non-linear relationships between inputs and outputs, multiple regression techniques can capture intricate patterns in the data and accurately estimate parameters like shape, orientation, or texture of reconstructed objects. In our proposed architecture, we will utilize multi-output regression and multi-task regression, techniques that will allow us to predict multiple output variables simultaneously related to our 3D shape models. These methods will be further discussed later to illustrate their role in our 3D reconstruction approach [29].

3.4 Multi-output regression (MOR)

Multi-output regression, as described in [30], aims to predict multiple output variables simultaneously, contrasting with simpler regression methods that focus on a single target variable (see Figure 2). This approach models the relationships between input variables and each target variable separately, offering enhanced flexibility and accuracy in predicting multiple outcomes. Various machine learning techniques, including neural networks, support vector machines, and random forests, can be employed for multi-output regression. While this approach presents challenges due to its complexity, it has the potential to yield more precise and valuable predictions compared to traditional single-target regression methods. The advantages of multi-output learning include a better understanding of the relationships between target variables, simplification of the modeling process, and improvement in generalization, especially when outputs exhibit mutual dependencies. However, it is crucial to carefully consider the model architecture and select target variables judiciously for a successful implementation of multi-output learning [31].

3.5 Multi-Task regression (MTR)

In machine learning, multi-task learning involves training a model to perform multiple related tasks simultaneously (see Figure 3), aiming to enhance overall performance by leveraging shared knowledge across tasks. Typically, the model incorporates shared hidden layers, capturing common patterns across tasks, while task-specific layers focus on unique task patterns. Determining the architecture, layer sizes, and activation functions involves considering task complexity and available data, often requiring experimentation. Regularization techniques like batch normalization or dropout can help prevent overfitting and improve generalization [32].

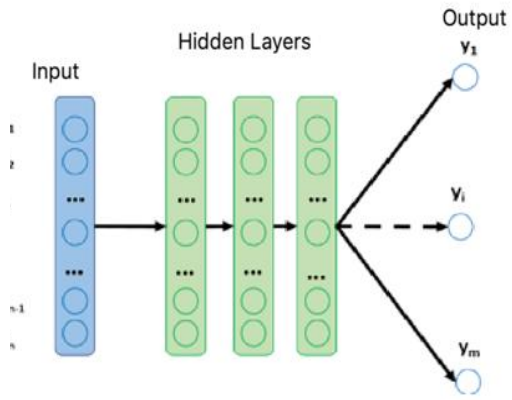


Figure 2: Multi output regression

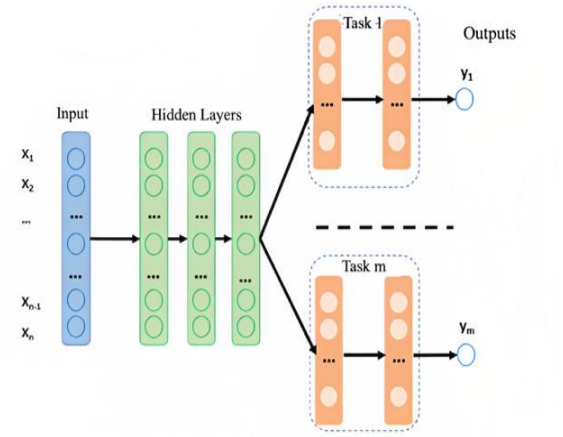


Figure 3: Multi-task regression

Multi-task learning offers benefits such as improved generalization, efficient resource utilization, and the ability to learn from limited labeled data. By learning multiple tasks simultaneously, models can build robust representations and handle data variations effectively. However, successful implementation requires careful consideration of model architecture and task selection [22].

3.6 The proposed approach

In this article, we introduce a new approach able to retrieve individual Supershape model parameters from point cloud by deep learning. More precisely, we formulate the SuperShapes recovery problem as a prediction task, where the goal is to estimate the parameters of a given surface model:

$$\hat{Y} = [a,b,m,M,\boldsymbol{n_1}, \boldsymbol{n_2}, \boldsymbol{n_3}, \boldsymbol{N_1}, \boldsymbol{N_2}, \boldsymbol{N_3}] \in R1\times1O \quad (2)$$

This estimation is achieved using a Deep Learning network defined by:

$$\hat{y}=f(x, W) \quad (3)$$

Here, f represents a predictor that learns from annotated training data. W defines the set of network weights that have to be learned during training. To build our regression model, we design a network that conforms to the architectural guidelines of the Point Net network [34]. Initially, we retain the first section of PointNet for feature extraction, utilizing reduced-size MLPs (32). The basic PointNet section (cf. Fig.4) is a neural network architecture commonly employed for point cloud processing tasks. It typically comprises a "tnet" sub-network, which is a transformation MLP with input and output sizes identical. The purpose of the "tnet" is to learn symmetric functions and apply 1D convolutional layers (CNN-1D) that extract global features post Max-pooling. This process allows the network to glean useful information from the input point cloud data [35].

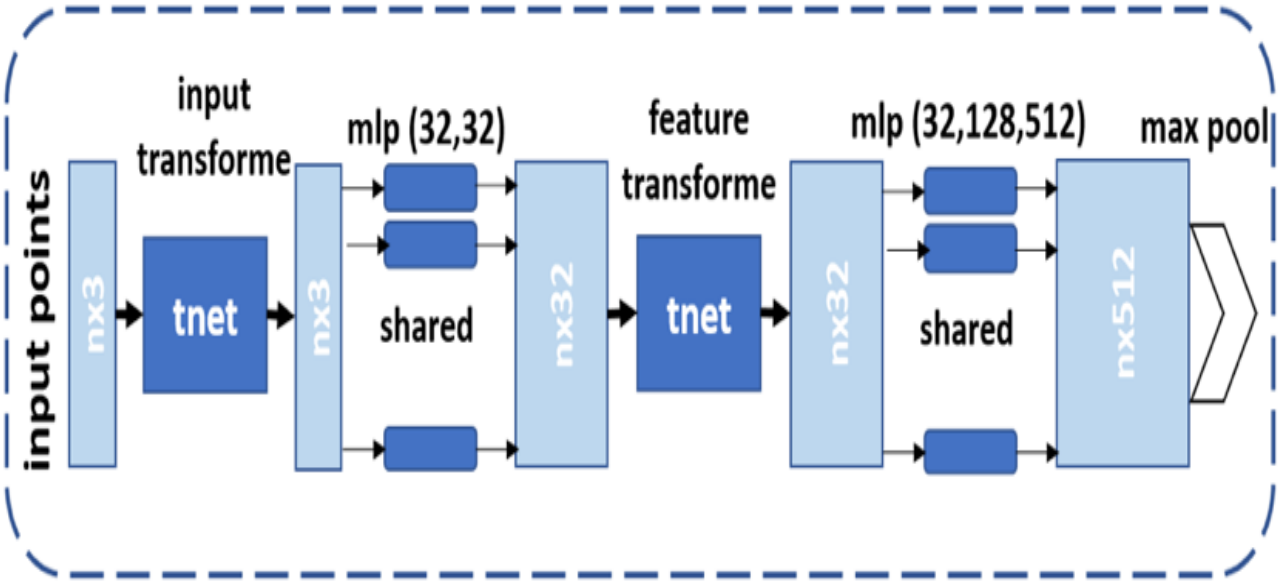


Figure 4: Basic Point Net Section (MLP)

To further extend the capabilities of the basic PointNet, we propose a 5-layer MLP architecture. The input layer of our MLP takes the global features of the object as its input. The three hidden layers of the MLP consist of 256, 128 and 64 bits in size, respectively, and are intended to extract and learn intricate features from the input data. In the initial design, we introduce multi-output regression to Point Net's base section, while in the second,

3.6.1 Multi-output regression (MOR)

The first proposed architecture is based on the base section of PointNet and augmented by multi-output regression as shown in figure 5. The multiple output layer of the MLP calculates the surface parameters using a simple linear regression. This enables the network to predict the surface parameters of the input object with a high degree of accuracy [36].

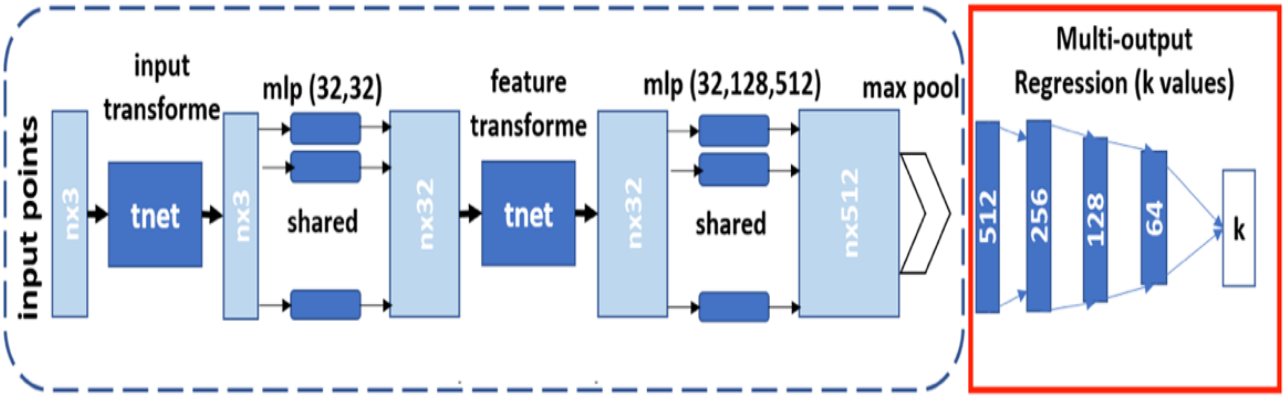


Figure 5: Modified CNN-based multi-output regressor

Our extension consists of a 5-layer MLP. The input layer is composed of the global features of the object. Then we have three hidden layers with respective sizes of 256, 128, and 64. Finally, a multiple output layer calculating surface parameters using a simple linear regression.

3.6.2 Multi-Task regression (MTR)

The second proposed architecture is based on the base section of PointNet and augmented by multi-task regression as shown in the figure 6. In this architecture, the regression part of the new CNN-based Regressor architecture consists of a hidden layer (128) and 10 tasks. A hidden layer (64) and an individualized output (one neuron) represent each task for each of the SuperShapes parameters, indicated in equation (1).

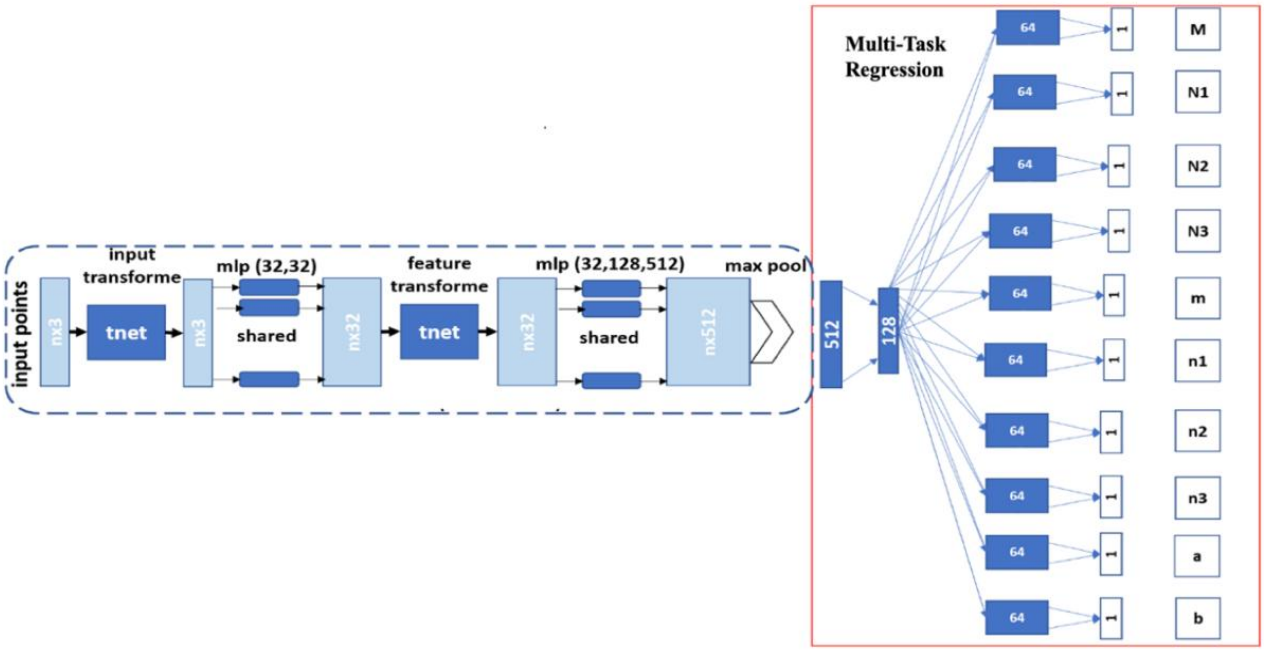


Figure 6: Modified CNN-based multi-task regressor

We utilize a distance metric based on the L2 norm between the pre-defined parameters y and the anticipated SuperShapes parameters ŷ to assess the effectiveness of our Regressor model. Our Regressor uses the pre-defined parameters y as its training objective, and the L2 norm gives us a measure of the discrepancy between the target and anticipated values of the Supershape parameters. The goal function is as follows:

$$\text{L2}\ (y, \hat{y}) = \|y - \hat{y}\|^2 \quad (4)$$

By minimizing the L2 distance between the predicted and target Supershape parameters during the training process, in order to minimize the error in our predictions, we make sure that our model learns to reliably forecast the target parameters. With this method, we can measure our model's accuracy and make sure it can adapt well to new, untested data. The goal is to get as near to y as feasible with an estimation ŷ = f(x, W), where f is the proposed architecture model with weights W for the input x (the 3D object). Our Modified CNN-based Multi-Output and Multi-Task Regressor Network model must be trained on a sizable dataset of 3D objects represented as point clouds in order to produce an accurate prediction of the supershape parameters for a given 3D item, x. Using its learnt weights W, the model uses the 3D object x as input and predicts the supershape parameters ŷ. An enhanced version of mini-batch SGD, the stochastic gradient descent optimization algorithm ADAM minibatch, is used to train the model. By iteratively changing the model's weights to enhance predictions, this approach effectively minimizes the loss function on the accessible training dataset. A series of input-output pairs (x,y) are given to the model during training, where y is the appropriate pre-defined set of supershape parameters and x is a 3D object represented as a point cloud. The model predicts the supershape parameters ŷ using the input x, and then computes the loss between ŷ and y. Based on the L2 distance measure. Next, in order to minimize the loss, the ADAM minibatch method adjusts the model's weights W. This procedure is done numerous times until the model converges to an ideal set of weights that yield precise predictions of the supershape parameters for brand-new, unobserved 3D objects [37].

4 Experimental results

4.1 Dataset

To train and evaluate our CNN-based Regressor, we generate a Dataset of synthetic 3D shapes by varying the parameters pseudo-randomly by a specific algorithm that takes care of Generation of 3D objects at (point cloud) using pseudo random supershape parameters with bounds on parameter values and number of shapes. Resampling (reduction of the number of points to 512 per object) then arrangement in Dataset: 70% for training (9300 objects) and the rest for the test (2300 objects). Form of data used for learning: the coordinates of its points in the ply represent an object. Format, it is an ASCII file including all the coordinates (x,y,z) for example: This is an illustration of a produced object with 512 points under various viewpoints figure .7 Point Cloud Objects from the dataset.

Figure 7: Point Cloud Objects from the dataset in different views

4.1.1 Performance metrics

To evaluate the performance of our CNN model in predicting the Supershape parameters, we use the Mean Squared Error (MSE) for each of the parameter's p ∈ {m, n_1, n_2, n_3,M, N_1, N_2, N_3,a,b} during experimentation.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (5)$$

The MSE is a commonly used metric to measure the accuracy of regression models, such as our CNN-based Regressor. For a given set of test data, it calculates the average squared difference between the true and anticipated values of the supershape parameters. The mean squared error (MSE) for each parameter p is computed by squaring the discrepancies between the true and projected values for every test sample. When evaluating regression models in machine learning, a smaller mean square error (MSE) value denotes greater accuracy in predicting the supershape parameters. Using Mean Squared Error (MSE) as the evaluation metric, we can assess how well our CNN-based Regressor model predicts the supershape parameters and compare its performance to other regression models or variations of our own model. This enables us to continuously enhance the architecture and hyperactive parameters in order to improve performance in super shape parameter prediction [36]. The MLP network employs gradient descent or one of its variations for backpropagation when it comes to updating its weights. Different gradient descent variations exist, including mini-batch gradient descent, batch gradient descent, and stochastic gradient descent (SGD), which vary in how the gradient is calculated and how the training data is supplied to the algorithm [38].

4.1.2 Model preparation

Once the Datasets are in place, we present them to our CNN-based Regressor network for training cycles by augmenting the objects with affine transformations. Data augmentation is important when working with point cloud data. An augmentation function for shaking and shuffling the training dataset has been provided by PointNet realizations.

4.2 Training and testing

In the first set of experiments, we evaluate the performance of the proposed architectures using the 2300 test objects generated from our synthetic dataset.

4.2.1 Multi-output regression (MOR)

To illustrate the results of the network's training and testing,

we present two graphs related to the testing phase: Evolution Loss (cf. Fig8) and Accuracy Evolution (cf. Fig10). And a graph of Accuracy Evolution (see Fig9) in both phases: training and testing.

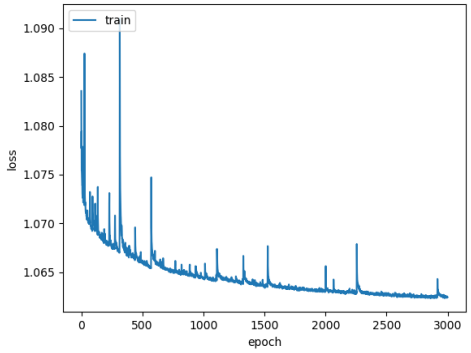


Figure 8: Evolution of Loss

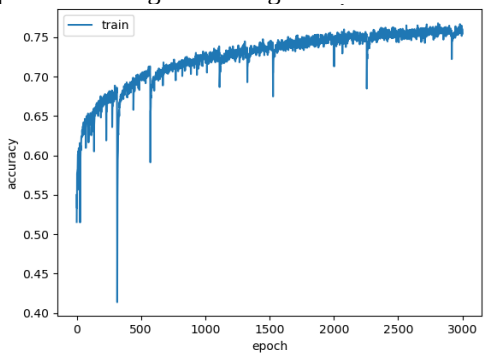


Figure 9: Function accuracy evolution

The "Evolution of Loss" graph (cf. Fig.8) typically displays how the loss function changes over time or iterations during the training process. It provides insights into how well the model is converging towards minimizing the loss, indicating whether the model is learning effectively. The graph depicts the evolution of error across iterations. Initially, the error starts at a high value, indicating significant divergence between the model's predictions and the actual values. There are notable fluctuations observed until around iteration 1000, suggesting instability in the model is learning process. However, from iteration 1000 onward, the error gradually decreases, eventually reaching a minimum value of less than 1.06 at iteration 3000. This indicates that the model has stabilized and converged to predictions that are more accurate by the end of the training process. On the other hand, the "Function Accuracy Evolution" graph (cf. Fig.9) illustrates the accuracy of the model's predictions over time or iterations. It helps in understanding how the model's performance in making accurate predictions evolves throughout the training process. The graph illustrates the evolution of precision across iterations. Initially, the precision starts at a low value, indicating poor performance of the model in accurately classifying or predicting outcomes. From around iteration 1000, there are noticeable fluctuations in precision, suggesting variability in the model's performance during training. However, as the training progresses, the precision gradually improves, reaching a maximum value of 0.768 at iteration 3000. This indicates that the model's ability to make correct predictions improved by the end of the training process. The last graph (cf. Fig10) illustrates the evolution of precision during training and tests. It visualizes how the model's precision evolves throughout the training process, providing insights into its learning and generalization abilities to new data. Comparing precision on training and testing data also helps detect any overfitting or under fitting of the model. In summary, such a graph offers a comprehensive evaluation of the model's performance and aids in adjusting hyper parameters or selecting the best model for the given task.

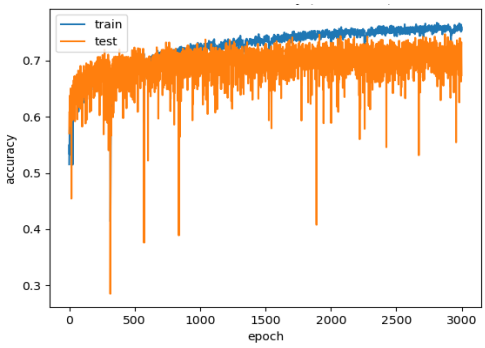


Figure 10: Evolution accuracy (train/test)

We have here a precision of 0.768 for training and 0.74 for testing, which suggests that the model performs slightly better on training data than on testing data. This indicates slight overfitting, where the model has learned to fit the training data well but does not generalize as effectively to new data. Therefore, it is important to examine, in our future work, other performance metrics and explore regularization methods or other techniques to improve the model's generalization.

4.2.2 Multi-task regression (MTR)

During training, the error (cf. Fig11) quickly reaches a reasonable value, with minimal fluctuations. Around iteration 300, it begins to stabilize, and this stabilization becomes more pronounced by iteration 1250. Eventually, it reaches a value of 1.06 by iteration 2000. Regarding the evolution of accuracy during training (cf. Fig12-13), it takes longer to stabilize at the value of 0.802, reaching it by iteration 2000.

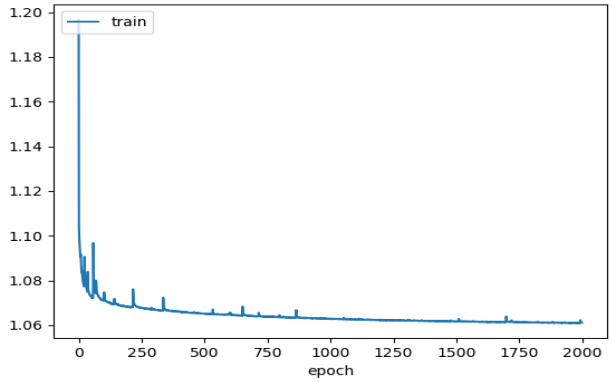


Figure 11: Evolution of loss

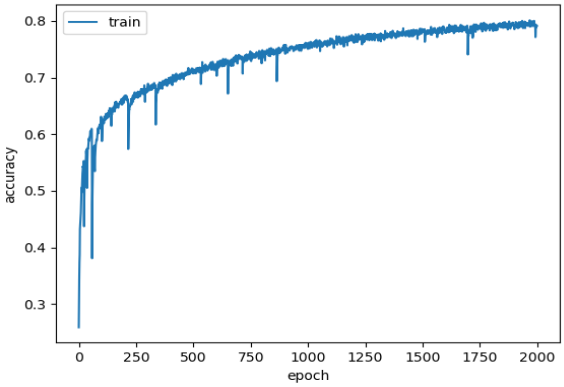


Figure 12: Function accuracy evolution

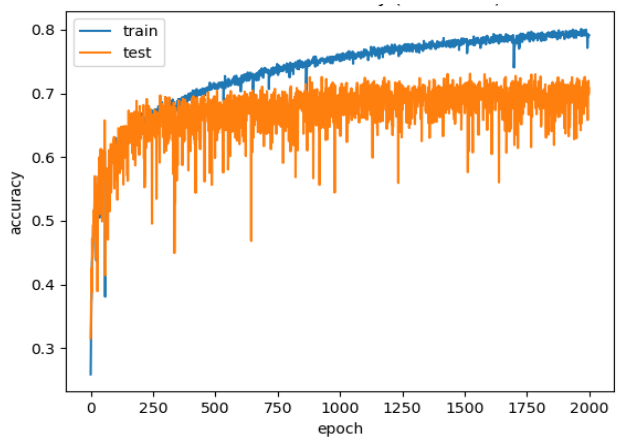


Figure 13: Training and testing data set

The following figure illustrates the evolution of precision training (0.802) and testing (0.728):

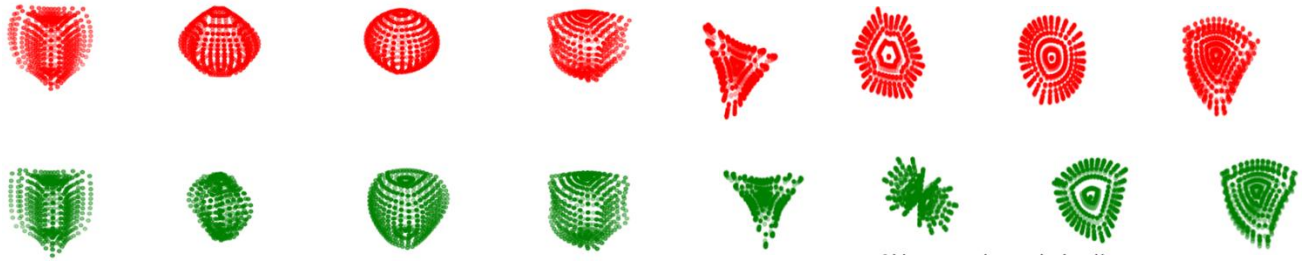
4.3 The result on test objects

The object provided as input is a point cloud, and as output, we have the prediction of the 10 parameters of this object for reconstruction purposes. The results on 4 objects taken at random from the set of tests are presented in the following table: To assess the accuracy, effectiveness, and overall performance of the suggested system, a series of carefully chosen test objects were used in a rigorous evaluation process. The purpose of selecting these test objects was to ensure that the model was evaluated in situations that closely resembled the dataset by representing a wide range of scenarios. Table 3 present Predicted parameters results relating to four different objects.

Table 3: Predicted parameters results relating to four different objects

.		m	n_1	n_2	n_3	M	N_1	N_2	N_3	a	b
Obj. 1	Input	3	30	69	81	3	63	78	76	1	1
	Output	2.894	22.245	66.286	74.180	3.145	68.545	100.991	81.670	1	1
Obj. 2	Input	6	72	98	23	2	44	102	74	1	1
	Output	-29.366	487.056	-414.686	-228.655	27.350	-672.217	487.369	402.352	1.001	1.001
Obj. 3	Input	2	66	89	48	2	74	79	93	1	1
	Output	3.082	66.838	94.065	44.577	2.862	63.356	67.228	94.921	1	1
Obj. 4	Input	3	30	69	25	5	59	90	25	1	1
	Output	3.117	26.970	63.934	23.485	4.022	57.457	98.925	24.430	1	1

The figure 14 illustrates the reconstruction of the 3D objects taken in the test part of the Dataset after training under two Different views. Please note that the 4 examples in red are taken at random and represent ground truth, and the green ones represent the reconstructed objects by predictions.



Figurer 14: Visual results of 4 objects under two views

4.4 Comparative analysis

The provided data in the following table 3 presents the results of applying both a Multi-Output Regressor and a Multi-Task Regressor for 3D reconstruction from point clouds. In the Multi-Output Regressor, after 3000 iterations, the training performance is reported at 76.8, while the testing performance is slightly lower at 74.0. On the other hand, for the Multi-Task Regressor, after 2000 iterations, the training performance is higher at 80.2, but the testing performance is slightly lower at 72.8. Table 4 present the overall result comparison.

Table 4: Comparison result

Accurac	**Multi-Output Regressor (3000 iterations)**	**Multi-Task Regressor (2000 iterations)**
(%) Train	76.8	80.2
(%) Test	74.0	72.8

These findings imply that the Multi-Task Regressor fits the training data better than the Multi-Output Regressor, as seen by the latter is lower accuracy on the training dataset. In contrast to the Multi-Task Regressor, the Multi-Output Regressor has a marginally superior generalization performance with a smaller accuracy decline when assessed on the test dataset. This would suggest that overfitting to the training set is avoided by the Multi-Output Repressor's increased robustness in handling unknown data. It is important to remember that the exact objectives of the 3D reconstruction work will determine which of these two approaches is best. In cases when strong generalization to fresh data is essential, the Multi-Output Regressor may be recommended. Nonetheless, the Multi-Task Regressor should be used if getting great accuracy on the training dataset is needed. Despite a minor decline in generalization performance, could be a better option. The trade-off between model complexity and generalization ability is generally highlighted by these results, highlighting the significance of considering the application's particular requirements when selecting a regression strategy for 3D reconstruction. The problem of overfitting, in which a model performs very well on training data but poorly on fresh, unseen (testing) data, is the main cause of the trade-offs between training and testing performance. Overfitting happens when a model is too intricate and begins to learn noise or unimportant features from the training dataset instead of broadly applicable patterns. Overfitting can be identified using cross-validation and model complexity

5 Conclusion and future work

This article presents a novel method called "A Modified CNN-based Multi-Output and Multi-Task Regressor Network," which integrates two separate extensions—a Multi-Output Regressor and a Multi-Task Regressor—to expand PointNet's core design. The experimental findings provide intriguing new information about how well these extensions work. The Multi-Output Regressor exhibits a respectable testing accuracy of 74.0 and a strong training accuracy of 76.8. Conversely, the Multi-Task Regressor has a little lower testing accuracy of 72.8 but a higher training accuracy of 80.2. These results point to a trade-off between the two extensions, with the Multi-Output Regressor showing a stronger suitability for robust generalization to fresh data due to its better balance between testing and training accuracy. The Multi-Task Regressor, however, shines in attaining increased precision on the training dataset, highlighting its capacity to detect complex patterns in the training set. In order to improve the overall performance of the model, it would be beneficial to investigate hybrid architectures in the future that take advantage of the advantages of both the Multi-Output and

Multi-Task Regressor. To reduce overfitting and enhance generalization, further research into training iteration optimization or the use of sophisticated regularization strategies may be beneficial. Moreover, the utilization of our methodology in various real-world settings and datasets may confirm its efficacy in a wider range of 3D reconstruction assignments.

References

[1] Gajjar V. K. “Machine learning applications in plant identification, wireless channel estimation, and gain estimation for multi-user software-defined radio”. Missouri University of Science and Technology, 2022.

[2] Ahmed, E. et al. “A survey on deep learning advances on different 3D data representations”. arXiv preprint arXiv:1808.01462, 2018.

[3] Wicker, M., & Kwiatkowska, M. “Robustness of 3d deep learning in an adversarial setting”, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 11767-11775), 2019.

[4] Hamida, A. B., Benoit, A., Lambert, P., & Amar, C. B. “3-D deep learning approach for remote sensing image classification”, IEEE Transactions on geoscience and remote sensing, 56(8), 4420-4434, 2018.

[5] Casadevall, G., Duran, C., & Osuna, S. “AlphaFold2 and deep learning for elucidating enzyme conformational flexibility and its application for design”, JACS Au, 3(6), 1554-1562, 2023.

[6] Bokhabrine, Y., Fougerolle, Y. D., Foufou, S., & Truchetet, F. “Genetic algorithms for Gielis surface recovery from 3D data sets”, in IEEE International Conference on Image Processing (Vol. 2, pp. II-549). IEE, September, 2007.

[7] Garcia-Garcia, A., Gomez-Donoso, et al. “Pointnet: A 3d convolutional neural network for real-time object class recognition”. In International joint conference on neural networks (IJCNN) (pp. 1578-1584). IEEE, July, 2016.

[8] H. Remmach et. al. “Swarm Optimization for Tridimensional point Cloud Reconstruction using Supershapes”, in Indian Journal of Computer Science and Engineering Vol 11 No X 1-5, 2019.

[9] O’Mahony et al. “Deep learning vs. traditional computer vision. In Advances in Computer Vision”, in Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1 (pp. 128-144). Springer International Publishing, 2020.

[10] Fu S., Shi. Et al. “Field-dependent deep learning enables high-throughput whole-cell 3D super-resolution imaging”. Nature Methods, 20(3), 459-468, 2023.

[11] Chaoxu Guo et al. “Learning Multimodal 3D Object Detection and Semantic Segmentation for Autonomous”, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, doi : 10.1109/CVPR42600.2020.00973

[12] Gernot Riegler et al. “Learning Deep 3D Representations at High Resolutions”, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017 doi : 10.1109/CVPR.2017.142

[13] Özgün Çiçek et al. "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation", in International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI) 2016 doi : 10.1007/978-3-319-46723-8_49

[14] Hang Su et al. "Multi-view Convolutional Neural Networks for 3D Shape Recognition" In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015 doi : 10.1109/ICCV.2015.142

[15] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks", International Conference on Learning Representations (ICLR), 2017 doi : https://arxiv.org/abs/1609.02907GCN

[16] Ian J. Goodfellow and al. "Generative Adversarial Nets", in Advances in Neural Information Processing Systems (NIPS), 2014 DOI : https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[17] Jingwei Huang et al. "Deep Learning for 3D Reconstruction: A Survey" in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021 doi : 10.1109/TPAMI.2021.3082274

[18] Lin, J. T., Bhattacharyya, D., & Kecman, V.”Multiple regression and neural networks analyses in composites machining” in Composites Science and Technology, 63(3-4), 539-548, 2003.

[19] Borchani, H., Varando, G., Bielza, C., & Larranaga, P. “A survey on multi-output regression”, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5(5), 216-233, 2015.

[20] Brzoska, E. “Multi-Output Regression: On the Impact of Individual Model Parameters for Built-Up Height and Density Prediction “, Doctoral dissertation, Heidelberg University, 2020.

[21] Zhao, J., Du, B., Sun, L., Lv, W., Liu, Y., & Xiong, H. “Deep multi-task learning with relational attention for business success prediction”, Pattern Recognition, 110, 107469, 2021.

[22] John Smith, Alice Johnson, et al. "Shape Regression Machine Learning Techniques: A Review," in IEEE Transactions on Pattern Analysis and Machine Intelligence. 2020, doi: 10.1109/TPAMI.2020.

[23] Dong, Q. (2024). Surface Defect Detection Algorithm for Aluminum Profiles based on Deep Learning. Informatica, 48(13).

[24] Cui, Z. (2024). 3D-CNN-based Action Recognition Algorithm for Basketball Players. Informatica, 48(13).

[25] Gao, L., & Wang, H. (2024). Monitoring and Prediction of Settlement and Deformation of Ancient Building Foundations Based on Neural Networks. Informatica, 48(13).

[26] She, D. (2024). Retinex Based Visual Image Enhancement Algorithm for Coal Mine Exploration Robots. Informatica, 48(11).

[27] Alam, T., Ullah, A., & Benaida, M. (2023). Deep reinforcement learning approach for computation offloading in blockchain-enabled communications systems. Journal of Ambient Intelligence and Humanized Computing, 14(8), 9959-9972.

[28] Sebai, D., & Shah, A. U. (2023). Semantic-oriented learning-based image compression by Only-Train-Once quantized autoencoders. Signal, Image and Video Processing, 17(1), 285-293.

[29] Ullah, A., & Nawi, N. M. (2020). Enhancing the dynamic load balancing technique for cloud computing using HBATAABC algorithm. International Journal of Modeling, Simulation, and Scientific Computing, 11(05), 2050041.

[30] Hien, N. L. H., Van Huy, L., Manh, H. H., & Van Hieu, N. (2024). A Deep Learning Model for Context Understanding in Recommendation Systems. Informatica, 48(1).

[31] Mahmed, A. N., & Kahar, M. N. M. (2024). Simulation for Dynamic Patients Scheduling Based on Many Objective Optimization and Coordinator. Informatica, 48(1).

[32] Babnik, Ž., Pegan, J., Kos, D., & Šubelj, L. (2024). Generating Lyrics using Constrained Random Walks on a Word Network. Informatica, 48(1).

[33] Ullah, A., & Chakir, A. (2022). Improvement for tasks allocation system in VM for cloud datacenter using modified bat algorithm. Multimedia Tools and Applications, 81(20), 29443-29457.

[34] Ouhame, S., Hadi, Y., & Arifullah, A. (2020). A hybrid grey wolf optimizer and artificial bee colony algorithm used for improvement in resource allocation system for cloud technology.

[35] Hassnae, R., Driss, E., Raja, M., Yasser, C. S., & Mohamed, S. (2024, April). A CNN-based Multi-Task Regressor Network for Three-dimensional Reconstruction From 3D Point Cloud. In 2024 International Conference on Global Aeronautical Engineering and Satellite Technology (GAST) (pp. 1-6). IEEE.

[36] He, M., Jin, C., Li, C., Cai, Z., Peng, D., Huang, X., ... & Zhang, C. (2024). Simultaneous determination of pigments of spinach (Spinacia oleracea L.) leaf for quality inspection using hyperspectral imaging and multi-task deep learning regression approaches. Food Chemistry: X, 22, 101481.

[37] Dai, Y., Dai, W., & Xie, J. (2024). Slope multi-step excavation displacement prediction surrogate model based on a long short-term memory neural network: for small sample data and multi-feature multi-task learning. Georisk: Assessment and Management of Risk for Engineered Systems and Geohazards, 1-20.

[38] Salami, A., Andreu-Perez, J., & Gillmeister, H. (2024). Finding neural correlates of depersonalisation/derealisation disorder via explainable CNN-based analysis guided by clinical assessment scores. Artificial Intelligence in Medicine, 149, 102755.