Data-driven 3D shape completion with product units

Ziyuan Li, Uwe Jaekel, and Babette Dellen

Department of Mathematics and Technology,
University of Applied Sciences Koblenz, 53424 Remagen, Germany
{zli,jaekel,dellen}@hs-koblenz.de

Abstract. Three-dimensional point clouds play a fundamental role in a wide array of fields, spanning from computer vision to robotics and autonomous navigation. Modeling the 3D shape of objects from these point clouds is important for various applications, including 3D shape completion and object recognition. This paper presents a complex-valued product-unit network for data-driven 3D shape completion. Using product units, sparse superpositions of complex power laws, including sparse polynomial functions, are fitted to incomplete 3D point clouds and used for extrapolating the data in the 3D space. In computer-vision applications, this task occurs frequently, e.g., when only partial views are available or occlusions hinder the acquisition of the full point cloud. We conduct a comparative analysis with a standard neural network to emphasize the superior extrapolation capabilities of product-unit networks within the 3D space. Furthermore, we present a real-world task that serves as a tangible demonstration of the proposed method's utility in the context of completing incomplete point cloud data acquired with a 3D scanner. This research contributes new insights into the field of neural network applications for 3D point cloud processing, revealing the broad potential of product-unit networks in this domain.

Keywords: Product units · 3D shape completion · Data-driven methods

1 Introduction

Shape completion is the process of inter- or extrapolating missing data that describes object shape. Incomplete shapes and point clouds are often encountered in computer-vision applications during data acquisition [2]. Often, only partial views are available or a part of the object is hidden due to self-occlusion or other objects placed in the viewing path. The missing data hinders both object recognition and point-cloud matching required for pose estimation. Hence, finding approaches to recover the complete object from partial data is a topic of high importance in current computer vision research, since it is needed for many applications, e.g., robotic grasping [1, 3, 15, 22, 23, 28].

Approaches for 3D shape completion can be distinguished in some simplification as either data driven or learning based. Learning-based approaches learn a

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

set of model shape classes from training data and match them directly to observations to perform shape completion [2, 27] or pose estimation [28]. Depending on the scenario, this may limit the method to some extent to the shape classes that have been included in the training data.

Different from that, data-driven approaches fit parametric models to partial data to derive a representation of the surface *ad hoc*. Polynomial surface models are a common choice for this task [4, 19, 20], but they also require some prior knowledge about the object shape to select a suitable model type [3, 10, 16, 17, 26, 29]. When the model type is unknown beforehand, many possibilities have to be considered to find a suitable sparse polynomial representation, which can render this task intractable.

Similar problems arise in machine-learning approaches when transforming the data into a higher-dimensional space [9, 24] or including nonlinear units representing higher-order polynomials in the network [8] to create nonlinear models (in a data-driven approach). Here, either the basis of this nonlinear space or a nonlinear kernel must be provided beforehand. Moreover, calculations in higher dimensions can be very complex and difficult [25]. The inclusion of nonlinear units in the network also allows better modeling of nonlinear relationships, but, again, the number of combinations of polynomial terms increases exponentially with the order of the polynomial terms, also leading to considerable computational complexity [7].

To address this problem, we propose complex-valued product-unit networks to generate shape models *ad hoc* from the available 3D point-cloud data. These shape models are sparse superpositions of complex power laws, including sparse polynomial functions. This is not an overly restrictive assumption on the model in many cases, since much more complex functions can often be well approximated by Taylor or fractional Taylor series expansions, i.e., linear combinations of power laws. Since the required leading terms of the sparse polynomial can be learned from the data directly, the problem of the complexity of choice is avoided.

Product-unit networks have already been shown before to have advantages towards standard neural networks in extrapolation tasks [12]. Complex-valued product units have been used previously to model the nonlinear properties of nuclear masses [11]. Standard neural network typically make piecewise, quasi-linear approximations of the functions or patterns they are meant to learn [12] which restricts the model's ability to extrapolate nonlinear relationships into regions of the feature space that the training data does not cover. In this paper, we advance the product-unit network presented in [11, 12] by adapting it to the problem of 3D shape completion and working in the complex-valued domain. We show that the extrapolation capabilities are of the complex-valued product-unit network are superior to the ones of a comparable standard neural network.

2 Methods

Briefly, our methodology involves the transformation of existing 3D point cloud data into the spherical coordinate system. Subsequently, we employ a complex-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

valued product-unit network for the purpose of predicting missing components and modeling shape as a 2D-function denoted by $r(\theta, \phi)$, where θ and ϕ represent the polar and azimuth angles, respectively, r denotes the radial distance. Here the network utilizes θ and ϕ as inputs, with r serving as output based on the training data.

2.1 Mathematical model

In contrast to a neuron in a standard neural network, the product unit operates by calculating the product of the powers of its input values [11–13, 21]. Mathematically, its output is expressed as follows:

$$y = \prod_{i=1}^{n} x_i^{\omega_i} \quad , \tag{1}$$

where x and y denote the input and output, respectively; ω represents the weight associated with each input, and n signifies the total number of input variables involved. Mathematically, the product operation can be substituted with a summation operation for computational efficiency [11–13, 21], leveraging the faster computational speed of addition over multiplication in digital systems. This substitution involves transforming the input values into their logarithmic equivalents, which are subsequently processed through a summation operation. The resulting summation output is then passed through an exponential function, enabling the neural network to produce comparable results. This transformation can be expressed mathematically as:

$$y = e^{\sum_{i=1}^{n} \omega_i log x_i} \quad . \tag{2}$$

If the input x_i is negative, the logarithm $\log_e x_i$ becomes complex, represented as $\log_e |x_i| + i\pi$ [13]. To support scenarios where the network's operations benefit from complex-valued parameters, we extend the weights and biases to the complex-valued domain, characterized by the expression $\omega = a + bi$. Here, both a and b represent real numbers, with i introducing a complex component to the weight parameter. In alignment with the complex-valued product unit, the weights and biases associated with other neurons, the summation units, within the network are configured in the corresponding complex form.

It is noteworthy that the extension of a neural network into the complex-valued space results in a doubling of both the weights and bias parameters compared to their original quantities, while the number of neural connections remains unchanged. However, when modelling real-valued functions, the imaginary parts of the weights often converge towards zero, effectively reducing the number of parameters by a factor again by a reality condition.

Instead of replacing all neurons in a standard neural network with complex-valued product units, our proposed network integrates product units to replace the standard summation units in a specific layer. This architecture harnesses the distinct strengths inherent in each type of computational unit. Conventional

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

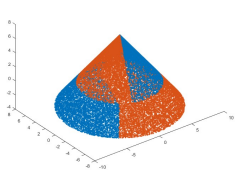
summation units within neural networks are adept at linear and simple nonlinear transformations of input data through activation functions. They specialize in learning fundamental patterns and features within the data. Complex-valued product units can capture more intricate nonlinear relationships within data, owing to their capability to compute products of powered inputs. This unique feature allows them to discern and model intricate dependencies and relationships among input variables. In the context of 3D shape completion, the architecture can be understood as a hierarchical approach to feature extraction. The standard summation units handle linear data transformations, while the complex-valued units build upon these to extract higher-level, more nuanced representations related to the geometry and structure of the 3D shapes. The complex nature of weights and biases in these complex-valued product units allows for a richer representation of data. By incorporating complex numbers, the product units are capable of processing negative inputs without necessitating the introduction of a threshold at the preceding layer.

2.2 Data sets and data acquisition

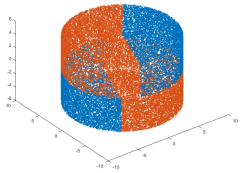
Firstly, we generated four different 3D point clouds to represent canonical geometric shapes, namely a cone, cylinder, ellipsoid, and cuboid. Each of these point clouds comprised precisely 30,000 data points, expressed in a Cartesian coordinate system defined by the coordinates x, y, and z. The representation of shapes in the spherical coordinate exhibits varying levels of complexity. For instance, this coordinate system is quite suitable to represent a cone. Conversely, accurately modeling a cuboid becomes exceedingly intricate due to the non-linear mapping of points within the spherical coordinate. To address this variability, we have categorized these shapes into three distinct levels of complexity. Shapes suitable for representation in the spherical coordinate system, such as the cone and cylinder, allocated 45% of their respective point clouds for training and interpolation, while the remaining 55% was earmarked for extrapolation. This distribution is visually depicted in Fig. 1(a) and (b). The ellipsoid, positioned at an intermediate complexity level, had an allocation of 70% and 30%, as shown in Fig. 1(c). In the case of the most challenging shape, the cuboid, 95% were used for training and interpolation, leaving 5% for extrapolation, as presented in Fig. 1(d).

Moreover, for the practical application of our methodology, we generated 3D models representing two irregularly shaped real-world objects: a bottle and a computer mouse, as illustrated in Fig. 2(a) and (c). This process involved utilizing a 3D scanner, specifically the Artec Space Spider [5], and its corresponding processing software. The object underwent an initial scanning phase using the 3D scanner, and subsequently, the acquired scan data was employed to construct a 3D model within the Artec Studio software. Following this, selected portions of the model were intentionally removed to simulate missing parts, and a 3D point cloud was generated through the utilization of CloudCompare software. The resulting point clouds are presented in Fig. 2(b) and (d), each comprising 50,000 points.

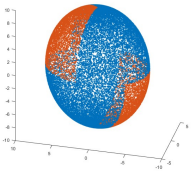
ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21



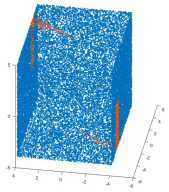
(a) Cone



(b) Cylinder



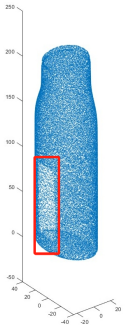
(c) Ellipsoid



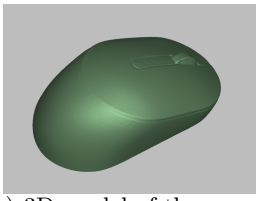
(d) Cuboid

Fig. 1. The point clouds representing the four distinct shapes have been partitioned into two subsets: training data and extrapolation data. The blue segment represents the data allocated for training and interpolation purposes, and the orange segment represents the data used for extrapolation and the subsequent validation of extrapolation results.

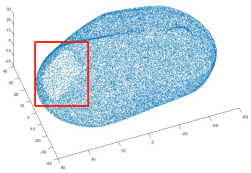
(a) 3D model of the bottle



(b) Incomplete 3D point cloud of the bottle



(c) 3D model of the mouse



(d) Incomplete 3D point cloud of the mouse

Fig. 2. 3D models and incomplete 3D point clouds of the objects, where the missing parts are marked by red boxes.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

2.3 Data processing

In practical applications, the acquired 3D point cloud may be situated at arbitrary positions within the field of view. To address this variability, we initially calculate the average of all point coordinates, designating this computed value as the central point of the point cloud. Subsequently, a novel Cartesian coordinate system is established with this central point serving as the origin. Conceptually, this procedure can be interpreted as a realignment of the point cloud to coincide with the origin of the pre-existing Cartesian system. Following this spatial adjustment, the point cloud data undergoes a transformation from the Cartesian system to the spherical coordinate system, which is characterized by the variables θ, ϕ, and r, and is computed by the following equations [18]:

$$\theta = \arccos \frac{z}{\sqrt{x^2+y^2+z^2}} = \arccos \frac{z}{r} = \operatorname{arcot} \frac{z}{x^2+y^2} \quad , \tag{3}$$

$$\phi = \operatorname{atan2}(y,x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0 \\ \frac{\pi}{2} \operatorname{sgn} y & \text{if } x = 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \wedge y \geq 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \wedge y < 0 \end{cases} \quad , \tag{4}$$

$$r = \sqrt{x^2+y^2+z^2} \quad . \tag{5}$$

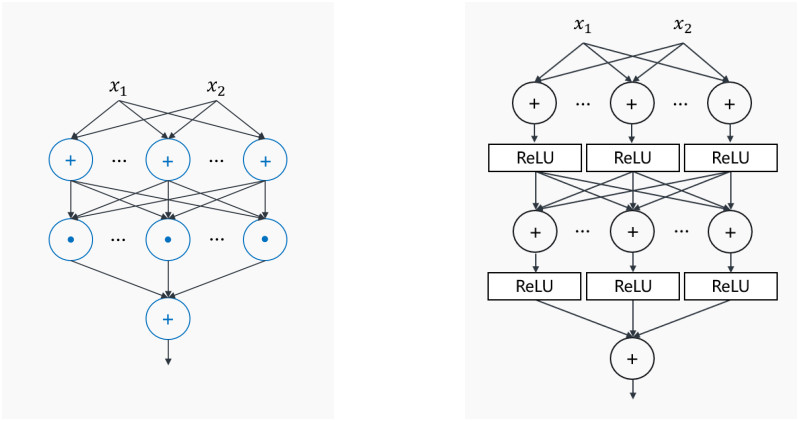
In this way, the association between r and (θ, ϕ) exhibits a one-to-one correspondence in the majority of cases, rendering it amenable to regression networks. Consequently, θ and ϕ serve as inputs for a regression network, with r being the output to be predicted. This underscores the necessity of transforming data from Cartesian coordinates to spherical coordinates. Retaining the data in Cartesian coordinates can lead to ambiguities, as the triplet (x, y, z) does not ensure a one-to-one mapping. For example, using x and y as neural-network inputs and z as the output can result in multiple points sharing identical x and y values but differing z values, thereby complicating the training of the network.

2.4 Network training

The employed complex-valued product-unit network is characterized by a streamlined three-layer architecture. The initial layer is composed of 10 complex-valued summation units, followed by a subsequent layer housing 120 complex-valued product units, the output layer features a singular complex-valued summation unit, notably absent of any activation function, as visually depicted in Fig. 3(a). In contrast, the reference network is a standard three-layer neural network operating with ReLU activation functions, as illustrated in Fig. 3(b). ReLU is a widely adopted activation function in neural networks, defined by the mathematical expression $f = \max(0, x)$, where x represents the input to the unit [6, 14]. As previously indicated, a complex-valued neural network exhibits twice the weights and bias parameter count compared to the standard neural network with the identical architecture, but the number of connections between neurons

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

is the same. This increases the total number of refresh parameters. So in order to make a fair comparison, we not only employed a standard neural network mirroring the exact architecture of the complex-valued product-unit network, both featuring an equivalent number of neurons per layer, but we also introduced an even lager standard neural network with double the network size in terms of number of neurons. Specifically, the network comprises 20 standard summation units in the first layer, 240 standard summation units in the second layer, and the output layer still contains only 1 standard summation unit. The larger standard neural network not only has a comparable number of weights and bias parameters to the number of complex-valued product units, but also has more connections between neurons. Their total number of refresh parameters is much greater than that of the complex-valued product-unit network.



(a) Complex-valued product-unit network (b) Standard neural network

Fig. 3. Neural networks. Neural units labelled in blue represent complex-valued neural units.

All neural networks are configured with mean square loss functions. However, the complex-valued product-unit network extends this loss function to accommodate the complex-valued domain, giving

$$\mathrm{L_{CMSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)(y_i - \hat{y}_i)^* \quad , \tag{6}$$

where y_i represents the ground-truth and $\hat{y}_i$ represents the predicted values. This formulation considers the complex conjugate $(\cdot)^*$. In contrast, the standard neural networks employ the mean square error (L_{MSE}) given by

$$\mathrm{L_{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad . \tag{7}$$

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

In addition, the three models underwent similar training procedures, employing the Adam optimizer for 2500 epochs. A linear learning rate scheduler was applied throughout these epochs, gradually reducing the learning rate from its initial value to 0.01 times that rate. However, to ensure optimal performance from the networks, we did not standardize the learning rates. Given that the product-unit network is more sensitive to weight changes compared to the standard neural network [12], it was assigned a smaller initial learning rate of 0.01. For the standard neural networks, when training with the cone and cylinder point clouds, a larger initial learning rate of 0.1 was used. However, during training with the ellipsoid and cuboid point clouds, we observed that the standard neural network performed more efficiently with an initial learning rate of 0.01 as opposed to 0.1. Consequently, we retained the 0.01 initial learning rate for these scenarios. In the real task, the initial learning rates were also all 0.01.

2.5 Identify the missing parts of the point cloud in the real-world task

In the real-world task, we cannot easily obtain input data for the missing sections of the point cloud. These data must be computationally derived. To solve this problem, we devised a density-based algorithm specifically tailored to identify these missing portions within 3D point clouds.

Our approach is grounded in the spherical coordinate system. Initially, we generate numerous small-scale grids covering plausible ranges of θ and ϕ values. Each grid represents a distinct section on the surface of the 3D model, effectively segmenting the point cloud into discrete regions. These regions are associated with specific intervals of θ and ϕ. The subsequent step involves quantifying the number of points residing within each demarcated region, enabling the calculation of point density for that specific area. Subsequently, we introduce a predefined threshold, expressed as a percentage. If the point density within a given region falls below this threshold, relative to the average density of its neighboring regions, the region is identified as missing.

Employing this algorithm enables the efficient identification of areas within the 3D model lacking sufficient data points. This process ensures that the reconstructed model is more accurate, enhancing this method overall robustness in practical applications.

3 Results

3.1 Synthetic objects

The neural networks were initially trained independently using the incomplete point cloud data. Subsequently, inputs from the complete point cloud were fed into the trained networks for prediction. Given that the outputs of the complex-valued product-unit network are also complex valued, it necessitates computing the absolute value of these complex numbers. The subsequent result is a 3D point

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

cloud as predicted by the trained model. It is important to emphasize that this 3D point cloud is entirely derived from the model's predictions. The predicted outcomes generated from input data that are part of the training set are termed "interpolation", while those from unseen data are referred to as "extrapolation".

Each point cloud underwent five test runs on each network. Loss values were computed for interpolation, extrapolation, and the overall total loss in each test, while training time was recorded. The average values from these five tests are comprehensively outlined in Table 1. Furthermore, for each shape, Fig. 4 illustrates the best outcomes, chosen from the five obtained through the complex-valued product-unit network and the ten acquired via the standard neural networks, respectively.

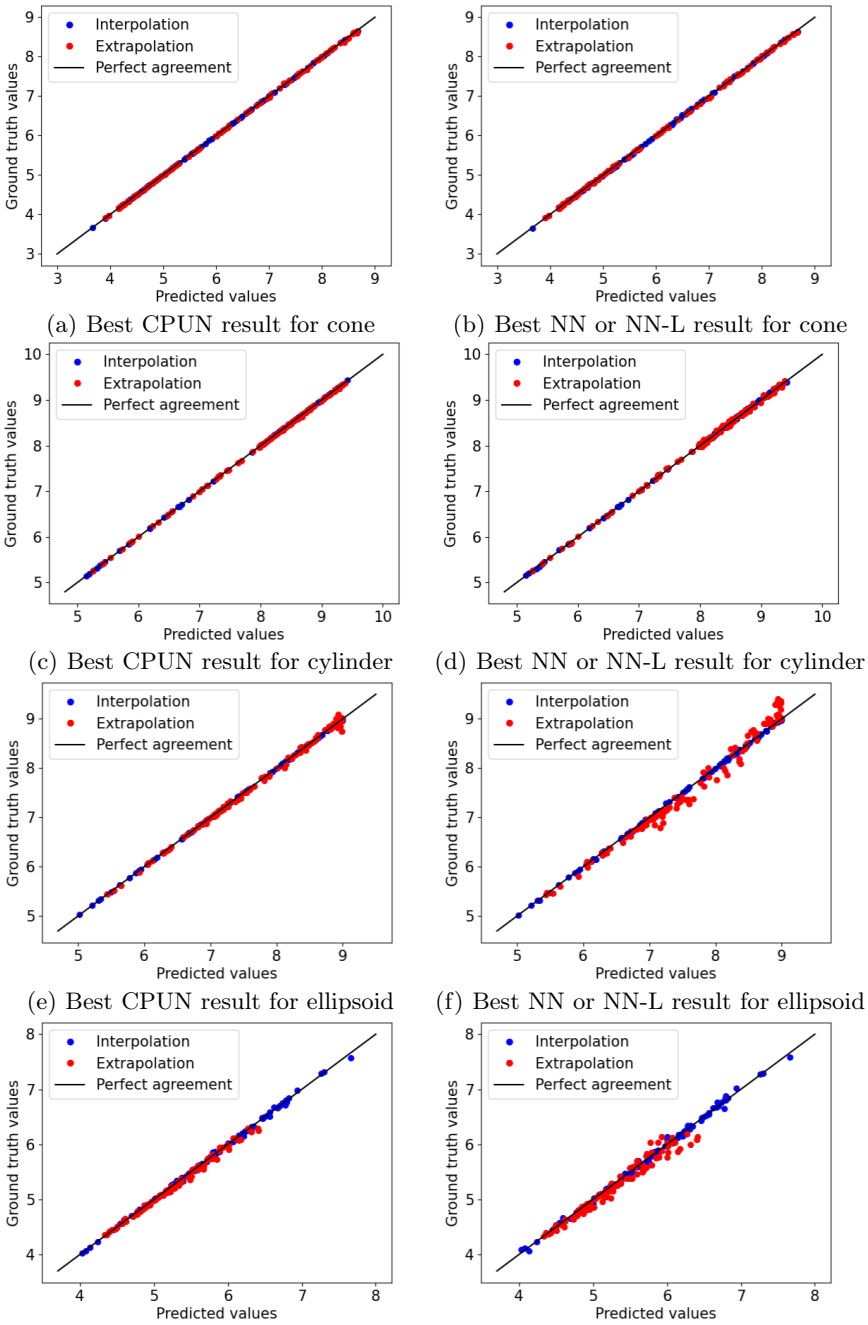
Table 1. Results of predictions for the synthetic objects. "avg." signifies average, "interp." and "extrap." denote interpolation and extrapolation, respectively. "CPUN" is an acronym for the complex-valued product-unit network, while "NN" refers to the standard neural network with an architecture identical to that of CPUN. Furthermore, "NN-L" denotes the larger standard neural network with twice as many neurons per layer except for the output layer as the NN. The best results are marked in bold.

Object	Cone			Cylinder		
Network	CPUN	NN	NN-L	CPUN	NN	NN-L
Avg. interp. loss [10^{-04}]	**0.065**	1.516	10.874	**0.099**	5.988	32.601
Avg. extrap. loss [10^{-04}]	**2.939**	5.565	17.909	**2.061**	16.190	42.456
Avg. total loss [10^{-04}]	**1.638**	3.732	14.724	**1.178**	11.601	38.024
Avg. training time [s]	285.2	**270.75**	272.81	277.48	**261.78**	281.69
Object	Ellipsoid			Cuboid		
Network	CPUN	NN	NN-L	CPUN	NN	NN-L
Avg. interp. loss [10^{-04}]	**0.297**	4.615	1.012	**17.538**	74.498	25.050
Avg. extrap. loss [10^{-04}]	**75.618**	947.080	639.980	**63.950**	272.860	178.540
Avg. total loss [10^{-04}]	**26.052**	326.844	219.500	**19.452**	82.660	31.370
Avg. training time [s]	417.14	**368.05**	401.38	594.53	**535.46**	583.17

As summarized in Table 1, for more straightforward geometries, such as the cone and cylinder, the complex-valued product-unit network has significantly lower average interpolation and extrapolation loss values in contrast to the two standard neural networks. When it comes to intricate geometries like the ellipsoid and cuboid, all networks have higher loss values. However, the complex-valued product-unit network remains superior, outperforming the standard neural networks across every performance metric, especially in terms of extrapolation capabilities. The observed variance in losses across different geometries aligns with expectations, considering the inherent challenges tied to representing complex structures such as the cuboid within a spherical coordinate system.

Additionally, a marginal difference in training duration was observed. Specifically, for networks with a comparable number of parameters, such as the complex-valued product-unit network and the standard neural network with a larger ar-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21



(a) Best CPUN result for cone (b) Best NN or NN-L result for cone

(c) Best CPUN result for cylinder (d) Best NN or NN-L result for cylinder

(e) Best CPUN result for ellipsoid (f) Best NN or NN-L result for ellipsoid

(g) Best CPUN result for cuboid (h) Best NN or NN-L result for cuboid

Fig. 4. Comparison of predicted results with ground-truth. Each plot incorporates 200 data points, which are evenly randomly sourced from interpolated and extrapolated results. If the predicted value is exactly the same as the ground-truth value, the point should be on the black line.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

chitecture, the complex-valued product-unit network necessitated slightly longer training times in the majority of scenarios.

3.2 Real-world task with real objects

The real-world task diverges from theoretical task for synthetic objects in their approach. For the real-world task, we retained the original incomplete point cloud data, focusing solely on predicting its missing segments. That is, we engaged only in extrapolation, excluding interpolation. Ultimately, the extrapolated results were merged with the original data to reconstruct a complete point cloud.

We trained the complex-valued product-unit network and the standard neural network with a larger architecture described above using two incomplete point clouds. The absent segments of these point clouds were pinpointed utilizing the density-based method outlined above. Following this, the trained networks were employed to predict the designated missing segments of the point clouds. The predictions were subsequently contrasted with the 3D point cloud derived from the complete model, as depicted in Fig. 5.

The Real-world task offers a more realistic representation of the uncertainties that the real world presents, as compared to the theoretical task for synthetic objects counterparts. In our observations, when dealing with irregular objects such as a bottle or computer mouse, the complex-valued product-unit network delivers more accurate results than the standard neural network. As depicted in Fig. 5, the point cloud segments predicted by the standard neural network displays considerable distortions. This is in agreement with the results in 1D and 2D investigated in [12]. In contrast, the predictions made by the complex-valued product-unit network align closely with the point cloud derived from the complete model, showing minimal deviations.

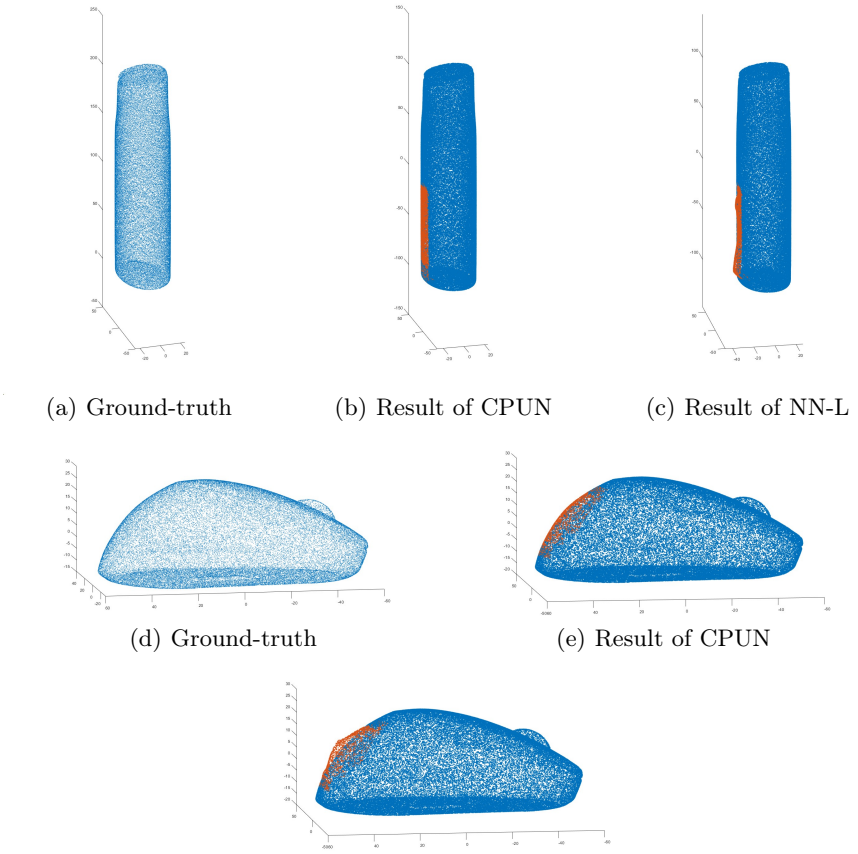
4 Conclusion and discussion

Based on our findings, it is evident that, having identical architectures, the complex-valued product-unit network exhibits superior predictive capabilities compared to the standard neural network. This disparity is particularly pronounced in terms of extrapolation ability. Even when the standard neural network is augmented with a doubling of neurons, its performance in both interpolation and extrapolation still falls short of that achieved by the complex-valued product-unit network.

The discernible discrepancy in performance becomes even more pronounced in the real-world task, particularly when dealing with real-world objects characterized by irregular shapes and structures. Standard neural networks consistently exhibit more severe distortions in the results they predict, in stark contrast to the highly accurate structures predicted by the complex-valued product-unit network in the face of such complex scenarios.

For this issue, we also analysed the reasons behind it. We believe that the effectiveness of neural network architectures is intricately tied to their inherent design and their capacity to meet specific challenges. While the standard

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21



(a) Ground-truth (b) Result of CPUN (c) Result of NN-L

(d) Ground-truth (e) Result of CPUN

(f) Result of NN-L

Fig. 5. The outcomes predicted by the two networks are compared to the ground-truth. The blue segment denotes the data acquired from scanning and subsequent processing (raw data), while the orange segment illustrates the extrapolated results derived from both networks.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

neural network stands as a versatile tool with broad applications, its performance, especially in domains like 3D point cloud analysis, may be surpassed by more specialized networks. The improved performance of the complex-valued product-unit network compared to the standard neural network observed in our experiments can presumably be attributed to its distinctive architecture. The initial layer, composed of standard summation units, allows linear transformation of the data to be learned. As the data progresses through subsequent layers containing complex-valued product units, the model can capture more intricate and contextually rich information, including nonlinear coupling between inputs, potentially crucial for understanding and completing complex 3D shapes.

However, it is worth noting the slightly prolonged training times for the complex-valued product-unit network. This delay indicates the computational demands of handling complex-valued operations. But, given the marked performance improvement, this trade-off will be justifiable for many applications, particularly those where accuracy is paramount.

In the course of our research, it became evident that the performance of the neural networks in 3D point-cloud completion tasks is susceptible to rotations of the object's point cloud. Specifically, when an object's point cloud is centered around the y-axis and then rotated to the x-axis by a certain angle, the predictive accuracy deteriorates. This decline in accuracy can be attributed to the fact that the mathematical representation of the point cloud undergoes a significant transformation after the rotation. In this regard, we believe that combining the method for identifying the 6D pose of an object with our current method is a good direction for our future research. Furthermore, we plan to explore the extension of our application to the challenge of complementing the incomplete contour of a 3D object in a camera view, particularly in situations where occlusion issues may arise.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International conference on machine learning. pp. 40–49. PMLR (2018)
2. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.J.: Learning representations and generative models for 3d point clouds. arXiv preprint arXiv:1707.02392 (2017)
3. Alenya, G., Dellen, B., Foix, S., Torras, C.: Robotized plant probing: Leaf segmentation utilizing time-of-flight data. IEEE Robotics & Automation Magazine **20**(3), 50–59 (2019)
4. Allen, B., Curless, B., Popović, Z.: The space of human body shapes: reconstruction and parameterization from range scans. ACM transactions on graphics (TOG) **22**(3), 587–594 (2003)
5. Artec 3D: Artec Space Spider (2024), https://www.artec3d.com/portable-3d-scanners/artec-spider, [Online; accessed 16-January-2024]
6. Behnke, S.: Hierarchical neural networks for image interpretation, vol. 2766. Springer (2003)
7. Bishop, C.M., et al.: Neural networks for pattern recognition. Oxford university press (1995)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

8. Clark, J.W., Gernoth, K.A., Dittmar, S., Ristig, M.: Higher-order probabilistic perceptrons as bayesian inference engines. Physical Review E **59**(5), 6161 (1999)
9. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**, 273–297 (1995)
10. Dai, A., Ruizhongtai Qi, C., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5868–5877 (2017)
11. Dellen, B., Jaekel, U., Freitas, P.S., Clark, J.W.: Predicting nuclear masses with product-unit networks. Physics Letters B **852**, 138608 (2024)
12. Dellen, B., Jaekel, U., Wolnitza, M.: Function and pattern extrapolation with product-unit networks. In: Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part II 19. pp. 174–188. Springer (2019)
13. Durbin, R., Rumelhart, D.E.: Product units: A computationally powerful and biologically plausible extension to backpropagation networks. Neural computation **1**(1), 133–142 (1989)
14. Hahnloser, R.H., Sarpeshkar, R., Mahowald, M.A., Douglas, R.J., Seung, H.S.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. nature **405**(6789), 947–951 (2000)
15. Husain, F., Colome, A., Dellen, B., Alenya, G., Torras, C.: Realtime tracking and grasping of a moving object from range video. IEEE International Conference on Robotics and Automation (ICRA) pp. 2617–2622 (2014)
16. Husain, F., Dellen, B., Torras, C.: Consistent depth video segmentation using adaptive surface models. IEEE transactions on Cybernetics **45**, 266 – 278 (2 2014)
17. Husain, F., Dellen, B., Torras, C.: Robust surface tracking in range image sequences. Digital Signal Processing pp. 37–44 (2014)
18. Itō, K.: Encyclopedic dictionary of mathematics, vol. 1. MIT press (1993)
19. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the fourth Eurographics symposium on Geometry processing. vol. 7, p. 0 (2006)
20. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM Transactions on Graphics (ToG) **32**(3), 1–13 (2013)
21. Leerink, L., Giles, C., Horne, B., Jabri, M.: Learning with product units. Advances in neural information processing systems **7** (1994)
22. Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Ojea, J.A., Goldberg, K.: Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. arXiv preprint arXiv:1703.09312 (2017)
23. Schmidt, P., Vahrenkamp, N., Wächter, M., Asfour, T.: Grasping of unknown objects using deep convolutional neural networks based on depth images. In: 2018 IEEE international conference on robotics and automation (ICRA). pp. 6831–6838. IEEE (2018)
24. Schölkopf, B., Smola, A.J., Bach, F., et al.: Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press (2002)
25. Shawe-Taylor, J., Cristianini, N., et al.: Kernel methods for pattern analysis. Cambridge university press (2004)
26. Smith, E., Meger, D., Pineda, L., Calandra, R., Malik, J., Romero Soriano, A., Drozdzal, M.: Active 3d shape reconstruction from vision and touch. Advances in Neural Information Processing Systems **34**, 16064–16078 (2021)
27. Stutz, D., Geiger, A.: Learning 3d shape completion from laser scan data with weak supervision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society (2018)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21

28. Wolnitza, M., Kaya, O., Kulvicius, T., Wörgötter, F., Dellen, B.: 6d pose estimation and 3d object reconstruction from 2d shape for robotic grasping of objects. In: 2022 Sixth IEEE International Conference on Robotic Computing (IRC). pp. 67–71 (2022). https://doi.org/10.1109/IRC55401.2022.00018
29. Zhang, Y., Liu, Z., Li, X., Zang, Y.: Data-driven point cloud objects completion. Sensors **19**(7), 1514 (2019)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_21