This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks.
Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

3-Dimensional Object Recognition Using 1-Dimensional Capsule Neural Networks

Amlan Basu
Ph.D. Student
Department of EEE
University of Strathclyde, Glasgow
amlan.basu@strath.ac.uk

Keerati Kaewrak
Ph.D. Student
Department of EEE
University of Strathclyde, Glasgow
keerati.kaewrak@strath.ac.uk

Lykourgos Petropoulakis
Senior Knowledge Exchange Fellow
Department of EEE
University of Strathclyde, Glasgow
l.petropoulakis@strath.ac.uk

Gaetano Di Caterina
Lecturer
Department of EEE
University of Strathclyde, Glasgow
gaetano.di-caterina@strath.ac.uk

John J. Soraghan
Professor
Department of EEE
University of Strathclyde, Glasgow
j.soraghan@strath.ac.uk

Abstract – Currently, indoor home object recognition systems lack the degree of accuracy required for reliable automated operations. In this paper, a 3-Dimensional (3D) object recognition deep neural network system, capable of recognizing indoor objects from 3D images with a view to assisting indoor robotic devices in performing home tasks, is presented. Almost invariably such systems drastically increase the number of total parameters which must be trained leading to very time-consuming training processes. Furthermore, the lack of suitably large annotated datasets for indoor objects adds to the training difficulties. To address these challenges a 3D dataset arranged in a 1D array format along with new architectures of 1D CapsNet is proposed. This effectively reduces the total number of parameters, resulting in faster and more accurate training of a neural network. Using the proposed architecture also appears to improve the training accuracy even when the datasets are relatively small. For this work, ModelNet-10 and the ModelNet-40 datasets are used. They have indoor home object images in 3D, which are few when compared to other datasets.

Keywords – Object recognition, Capsule Neural Network (CapsNet), PointCapsNet, 1D CapsNet, 3D Objects, 1D Array

I. Introduction

Over the years, object recognition and detection tasks have been carried out using different neural networks. In most of these cases, neural networks are trained using 2D images contained in widely used 2D datasets such as the COCO and the SUN databases. Many of the methods used for object recognition tasks are based on 2D architectures, i.e., both the neural networks developed and the data for training them are 2D. This has led to the following limitations: a) there is no guarantee that a neural network, which has learned to recognize an object in some orientation, will be able to automatically generalize and recognize the same object presented in some different orientation; b) to improve the recognition accuracy of objects using 2-D images very large datasets are needed to train the networks as the same object must be presented in different orientations under the same conditions – thus attempting to overcome the previous limitation.

Having larger datasets for objects is not necessarily a problem if enough objects and different 2D views can be acquired, even though this raises a question over suitably efficient image representations. Often, however, such large databases are not readily available. Thus, a requirement exists to design neural networks capable of delivering equivalent or better results using smaller datasets. Such networks not only would utilize resources more efficiently but could also reduce complexity and increase the speed of training.

Another issue arising with datasets is that objects are labelled using annotations on an image. This leads to the inclusion of the background along with the object, which creates confusion for neural networks during training. Labelling multiple objects through an annotation on the same image can create background overlap, this overlap therefore will appear under several different labels thus creating confusion during the training or learning process.

Therefore, there is a requirement for neural networks which can be trained on 3D datasets and which can acquire the complete information of the features of the objects along with their proper orientation. Some of these issues have been addressed in our previous work [1]. To address these issues, the use of 3D images of indoor home objects is proposed. In the 3D images, each image consists of only one object and nothing else.

In this work, a CapsNet with a pooling layer is presented for improved accuracy – this approach was also shown to provide superior results in our study [2]. Nonetheless, any 3D neural network will, inevitably, have a large number of parameters requiring training. To mitigate this drawback, it is proposed that a 3D dataset is represented in a way that enables the training of the neural network on 3D data but which prevents the total number of trainable parameters to increase appreciably.

In this paper, therefore, we address the aforementioned issues by presenting the datasets as 1D arrays (i.e., a long-

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J.
(Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks.
Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI),
Hyderabad, India.

chain of inputs). To process these inputs and extract information from them the CapsNet architecture is also converted to 1D. In this study, we make use of the ModelNet-10 and ModelNet-40 datasets [3] as this will help in comparing our results with other works as all the related works have used these datasets. These datasets are present in point cloud form which usually needs to be converted to 3D voxel grids to train the neural networks. This also leads to an increase in the total number of parameters. Therefore, to use the dataset in a point cloud format and to keep the total number of trainable parameters as low as possible, new CapsNet architectures are proposed and developed in 1D. The two different architectures are 1D CapsNetA and 1D CapsNetB. The only difference between them is the position of the Max Pool layer. Both the architectures are the combination of 1D convolutional layer, 1D Max Pool layer and 1D capsule. The architecture also appears to help produce improved accuracy even when the network is trained with limited amounts of data. The total number of trainable parameters also decreases because there is no requirement of input dimension.

Work presented here also includes the PointCapsNet. This is based on the PointNet architecture (Cheraghian and Petersson, 2019), which was the original design used to recognise 3D objects from point cloud datasets by using 1D arrays. PointCapsNet, therefore, is formed by integrating PointNet with 1D capsules. This was the initial system we designed and tested and the predecessor to the aforementioned 1D CapsNet architectures. The PointCapsNet analysis helped to understand the basic requirements needed to develop leaner and more efficient designs. It is, therefore, presented here for completion and comparison.

This work, therefore, focuses on developing neural networks which can perform highly accurate indoor object recognition even if they are trained on smaller datasets. Given, also, that these networks have a smaller number of total trainable parameters, the result is a more efficient training process.

This paper is structured as follows: following this introduction is the background work, which discusses the related works that have taken place so far. This is followed by the capsule design description which presents all the technical aspects of the most important parts of a CapsNet architecture. The following section introduces the proposed Point Capsule Neural Network (PointCapsNet). This is followed by the section on the 1D CapsNets developed to accomplish the main objectives of this work. Subsequently, the datasets are presented which consist of a description of images used for this work. The following section discusses and compares the results produced. Finally, the last section presents the conclusions.

II. Background Works

The research that are closely associated with the research presented in this paper, have flaws like unable to train a neural network directly on point cloud dataset. This leads to conversion of data from point cloud to 3D voxel grids. Further, the high number of trainable parameters leading to increase in training time of the neural network, specifically in CapsNet. All these will be discussed further using different works available.

Object detection and recognition are two very important and pioneering tasks in the field of computer vision. In recent years, many neural networks have been developed for performing object detection and recognition such as Fast R-CNN (Region-based Convolutional Neural Network) [4], Faster R-CNN [5], Mask R-CNN [6]. These methods are based on 2-D architectures with all the issues and difficulties discussed in the introduction section.

Charles R. Qi et al. [7] developed a neural network called PointNet that is a 1D neural network that recognizes 3D objects. It was trained on the ModelNet-40 dataset. The network accepts a number of points as inputs. The points represent the complete image in a 3D mesh. Therefore, during training, only a specified number of 3D points will be taken as inputs at one time. Transformation of features and inputs is applied. Then the Max Pooling layer aggregates the features of a point. Grades are given to the k number of classes in the output. Later in the network concatenation of output scores per point and local and global features are sent to the last MLP (Multi-Layer Perceptron) structure as input. Batch Normalization with ReLU activation function is used in every layer. Dropout characteristics are used in the last MLP. The training data, present in the point cloud, is converted into 3D voxel grids because of the irregularity (no proper edges, lines, curves, etc. which are important during information extraction) present in the geometric structure of the point cloud. However, for the PointNet point cloud, data can be used directly for training purposes. The total number of trainable parameters in PointNet is less than other neural networks but the accuracy remains relatively low when compared to other available architectures. In addition, the difference in accuracies between datasets is too large which is quite unusual.

Jiaxin Li et al. [8] introduced the SO-Net (Self-Organizing Network) which was developed specifically for directly learning the 3D objects from the point cloud dataset. The encoder part of SO-Net includes the use of k-nearest SOM (Self-Organizing Map) nodes for the normalisation of input points. Then based on kNN (k-nearest neighbour) search on SOM, the Max Pooling of normalized point features is done into feature nodes. The field overlap is determined by k. A kNN association is followed during the concatenation of kN normalized points and M node features, in the segmentation part of the network. Finally, aggregation using average pooling is performed on kN features. Even though SO-Net produces high accuracy rates the developed architecture is too complex and therefore it makes the training task complicated as well because of the increased trainable parameters.

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks. Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

Achlioptas et al. [9] also developed a system to directly train a network on the point cloud dataset. The work implements Generative Adversarial Network (GAN) and Auto-Encoders (AE) to accomplish the work. The first GAN is termed r-GAN (raw-GAN) in which the point cloud dataset is used as it is for training. The discriminator in r-GAN is similar to AE but does not have any batch normalization and has a leaky ReLU activation function. In the end, it has 5 FC (fully connected) layers with a ReLU activation function whose output is fed to the sigmoid function to get the final output. The second Gan is l-GAN (latent space GAN). In this GAN, the AE used is pre-trained. The AE is trained on each object separately using EMD (Earth Mover's Distance) and CD (Chamfer Distance) loss function. Further, in l-GAN, a single hidden layer MLP acts as a generator joined with an MLP of two hidden layers which acts as the discriminator. Separately a group of GMMs was also used that can generate point clouds. This is done by fitted distribution sampling and subsequently using the decoder of AE (pre-trained). The l-GAN has worked very well with CD loss function and has produced the best accuracy. However, looking into the accuracies produced it can be inferred that the architecture produced better results on data with fewer classes than on data with more classes. Therefore, if the classes increase then the accuracy decreases. This is the biggest drawback and raises question marks on the efficiency of the developed architecture.

Ayesha Ahmad et al. [10] developed 5 different Capsule Neural Network (CapsNet first introduced by Sabour et al. (Sabour et al., 2017)) architectures for 3D object recognition. All the CapsNets are trained and tested on the ModelNet-10 and ModelNet-40 datasets.

A 3D Capsule introduced by Ali Cheraghian et al. [11] is based on some concepts of PointNet and Capsules in 3D. Input for the architecture is in point cloud form as well. These are the highest obtained percentages in terms of Capsule Neural Networks. However, in terms of efficiency, there are still some issues in the architecture. Since this is a 3D CapsNet the total number of trainable parameters is too high which makes the training very slow.

Another CapsNet for the same task was introduced by Ryan Lambert et al. [12] which is named as 3D-CapsNet. However, the input form of the dataset is a 3D image in this architecture. The architecture is similar to one introduced by Sabour et al. [13], but, in this case, it is converted into 3D form. Here the total number of parameters is even greater than the 3D CapsNet introduced by Ali Cheraghian et al. (Cheraghian and Peterson, 2019), as both the data and the neural network are in 3D.

Asako Kanezaki et al. [14] introduced RotationNet which, until now, has the highest accuracy on both ModelNet-40 (97.37%) and Modelnet-10 (98.46%) datasets. RotationNet is a CNN that takes inspiration from the drawbacks of MVCNN (Multi-View Convolution Neural Network) [15]. However, it does not take training input as point cloud but as an image just like MVCNN. In the case of both MVCNN and RotationNet, 2-D images of the objects from one or more than one angle are shown to the neural network which automatically increases the dataset.

Apart from the aforementioned works and the problems associated with them, some other neural networks are implemented for the same work like PointNet++ [16], PointConv [17], PointCNN [18], MLVCNN (multi-loop view CNN) [19], LDGCNN (Linked Dynamic Graph CNN) [20], etc. They are either trained and tested only on ModelNet-40 or on the ModelNet-10 dataset and are known for their accuracy but have the same problems discussed in previous paragraphs.

From the above examples, it is clear that to train a neural network directly on a point cloud dataset, the neural network needs to have properties similar to PointNet or the point cloud needs to be converted to 3D voxel grids. This, of course, leads to an increase in the total number of trainable parameters. To mitigate this issue the development of 1D CapsNetA and 1D CapsNetB is proposed in this paper. Further, the paper concentrates more on making CapsNets better because they can produce good accuracy on smaller datasets [1, 2]. Therefore, the aforementioned works (apart from CapsNets) discussed are to establish a proper comparison with the work presented in this paper.

III. Capsule Design

The most important component of CapsNets is a Capsule. The Capsule was introduced by Hinton et al. [21] to overcome the disadvantages faced by Convolutional Neural Networks (CNN). In CNN, the pooling layers looses a lot of vital information while extracting and resizing the content. In addition, a CNN fails to learn the relationship between the different extracted features because of the absence of any function that can help in acquiring the required information, which can be found in (Sabour et al. [13]) and (Hinton et al. [21]). Therefore, to overcome these problems, a capsule with a different structure was developed by G. Hinton et al. [21].

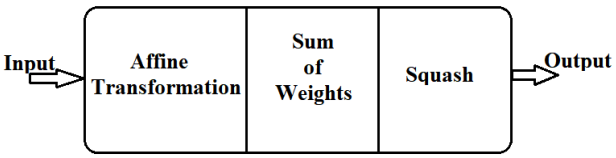

Fig. 1. Basic Capsule design

The first process in the capsule shown in Fig. 1 is the affine transformation that helps in acquiring the relationship between the features extracted by the convolutional layer. The input to the affine transformation is in vector form which is responsible for applying the transformation matrix. The transformation matrix helps in representing the missing spatial relationships. Missing spatial relationships are the coordinates that are not acquired as information by the layers before capsules. The coordinates help to locate the exact position of different

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks. Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

information acquired by layers before the capsule. Since capsules also have a collection of neurons, the sum of weights is the second function. The sum of weights is performed for vector inputs received after affine transformation. The last function is the squash function which does the resizing of the extracted information like the pooling layers do in CNNs. However, this function does not lose any information as it is a non-linear function that takes the input in a vector form and without changing the direction resizes the information in the unit vector.
The mathematical expressions or formulae for the functions in the capsule are,

1. Affine Transformation:

$$\hat{u}_{j|i} = W_{ij}\, u_i \quad (1)$$

Where, $\hat{u}_{j|i}$ – Prediction vectors, W_{ij} – Weight matrix and u_i – Output of a capsule.

2. Sum of Weights:

$$s_j = \sum_i c_{ij}\, \hat{u}_{j|i} \quad (2)$$

Where, c_{ij} – Coupling coefficient,

$$c_{ij} = exp(b_{ij}) / \sum_k exp(b_{ik}) \quad (3)$$

Equation 3 is nothing but routing softmax.

3. Squashing:

$$v_j = \frac{||s_j||^2}{1+||s_j||^2} \frac{s_j}{||s_j||} \quad (4)$$

Where, v_j - Output vector of capsule j, s_j – Total input of capsule j and $\frac{||s_j||^2}{1+||s_j||^2}$ – squashing and $\frac{s_j}{||s_j||}$ - unit scaling. The number of capsules in layer l and l+1 is denoted by i and j.

IV. Capsule Neural Networks (CapsNets)

CapsNet is the neural network that has convolutional layer and capsule. CapsNet introduced by Sabour et al. [13] implemented dynamic routing. This is a mechanism to connect lower-level capsules to higher level capsules to transfer information. Dynamic routing also helps in finding the coupling coefficient which makes the calculation of sum of weights of vectors possible.

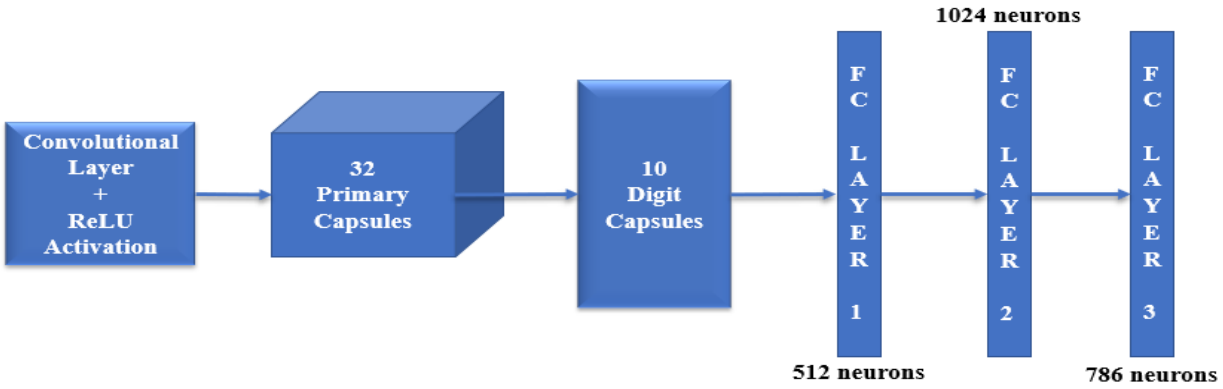

Fig. 2. Schematic diagram of Sabour et al. [13] CapsNet

The CapsNet was trained tested on the MNIST dataset. The CapsNet design had one convolutional layer with ReLU (Rectified Linear Unit) activation function, 32 primary capsules (the capsule responsible for performing the functions of basic capsule), 10 digit capsules (responsible for storing the information received from the primary capsule in the form of digit) and 3 fully connected (FC) layers with the first two having ReLU activation and the last one having a sigmoid activation function. The first second and third FC layers have 512, 1024 and 784 neurons respectively. The diagram of the same can be seen in Fig. 2. For this work, an analysis of how PointNet architecture works with a capsule was needed. Therefore, a proposed PointCapsNet was developed. The schematic diagram of the proposed architecture is shown in Fig. 3. The architecture is developed by integrating primary and digit capsules in the PointNet architecture. This architecture was specifically developed to train a neural network on a dataset with point cloud structure without conversion to 3D voxel grids.

All these are included in PointCapsNet as originally designed in PointNet. With reference to Fig 3 in PointCapsNet, the primary and digit capsule layers are connected between the global feature layer and the last MLP (Multi-Layer Perceptron) layer having a dropout.
Since the architecture shown in Fig. 3 accepts data in 1D array format, therefore, the primary and digit capsules integrated in the PointCapsNet are also 1D. However, the PointCapsNet did not perform as expected, especially in producing better accuracy. In addition, the architecture is somewhat complicated because of the many different functions. The need for a simpler architecture and improved performance led to the development of two different CapsNet architectures. These architectures not only incorporate the capsule but also pooling layers, making the architecture CapsNets with a pooling layer.

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks. Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

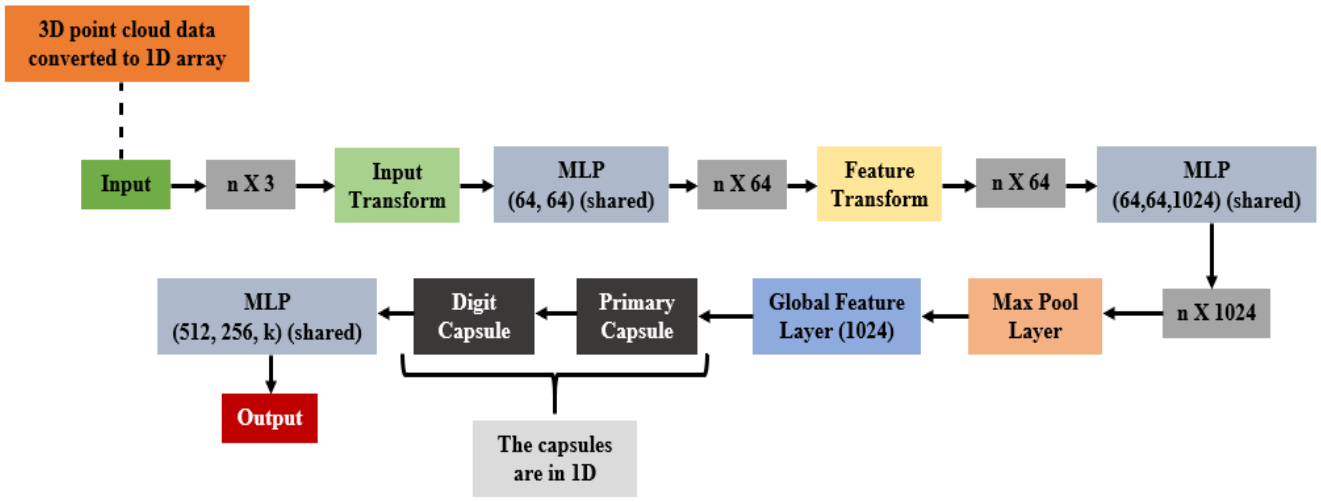

Fig. 3. PointCapsNet (Point Capsule Neural Network) Architecture

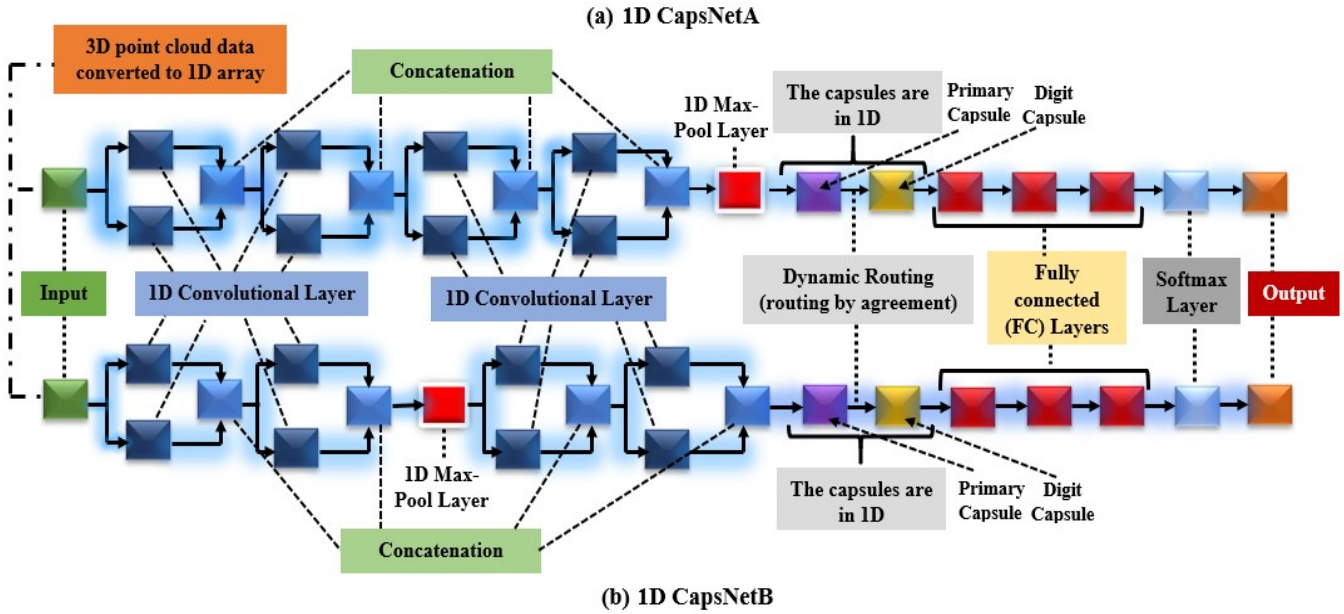

Fig. 4. Architectures of One-dimensional Capsule Neural Networks (1D CapsNets) with Max-Pool
4(a). 1D CapsNetA with Max-Pool, 4(b). 1D CapsNetB with Max-Pool

The two CapsNet architectures developed because of the limitations of PointCapsNet are the 1D CapsNetA and the 1D CapsNetB which are shown in Fig. 4. In the 1D CapsNetA with Max Pool layer is shown in Fig. 4a, the first four layers of the architecture comprise two convolutional layers whose outputs are concatenated. Each concatenated output becomes the input for the next two convolutional layers. The filters of the convolutional layers in the first, second, third and fourth layers are 64, 128, 256 and 512 respectively. All convolutional layers have a channel size of 3. The concatenated output from the fourth layer becomes an input for the Max Pool layer (channel size 2X2) that aggregates the features and transfers the data to the capsules. In the primary capsule layer, the orientation of all the features presented to it, are extracted and resized, so that the memory size of the CapsNet does not explode. The number of primary capsules here is 32 and the number of channels is 16. The 32 capsules are equally distributed over 16 channels i.e., each channel has 2 primary capsules. The acquired information by the primary capsule is sent to the digit capsule layer (DigitCaps) where all the information is stacked as per the classification category. For ModelNet-10 and ModelNet-40 datasets, the number of cells used in DigitCaps is 10 and 40 respectively (as there are 10 different categories in ModelNet-10 and 40 different categories in Modelnet-40). The number of channels used in DigitCaps is 16.

Then all the information from DigitCaps is sent to the FC layers. There are three FC layers in this architecture. The first and second FC layers have 4096 neurons and the third FC layer has 5 neurons. All the convolutional and FC layers have a ReLU activation function. Finally, the architecture incorporates a softmax layer which helps to give a proper output. The 1D CapsNetB with Max Pool is the second CapsNet architecture proposed and shown in Fig. 4b. The only difference between the 1D CapsNetA and 1D CapsNetB with Max Pool is the position of the Max Pool layer in the architectures. The Max Pool layer in Fig.

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks. Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

4b is placed after the second layer of concatenated convolutional layers. This helps to aggregate the extracted features from the first two sets of concatenated convolutional layers. All the convolutional layers, max pool layer and capsules are of 1D. The padding for every layer is kept unchanged (in the programming language, same).

In Table 1, conv1_1 and conv1_2, conv2_1 and conv2_2, conv3_1 and conv3_2 and conv4_1 and conv4_2 are concatenated convolutional layers in layers 1, 2, 3 and 4 respectively for the 1D CapsNetA design. By contrast, in 1D CapsNetB the same concatenated layers appear as layers 1, 2, 4 and 5 respectively. All the convolutional layers, Max Pool layer and capsules are of 1D. The technical specifications of both 1D CapsNetA and 1D CapsNetB remain the same. The stride for all convolutional layers and Max pool layer is 1.

Table 1. Complete specifications of 1D CapsNetA and 1D CapsNetB

Name	Specifications	1D CapsNetA Layers	1D CapsNetB Layers	Strides
Input	1D ModelNet-10 and ModelNet-40 dataset	Yes	Yes	-
Conv 1_1 Conv 1_2	Output Concatenated Filters: 64 Channel Size: 3	Yes, Layer-1	Yes, Layer-1	1
Conv 2_1 Conv 2_2	Output Concatenated Filters: 128 Channel Size: 3	Yes, Layer-2	Yes, Layer-2	1
Max-Pool	Channel Size: 2 Padding: Same	No	Yes, Layer-3	1
Conv 3_1 Conv 3_2	Output Concatenated Filters: 256 Channel Size: 3	Yes, Layer-3	Yes, Layer-4	1
Conv 4_1 Conv 4_2	Output Concatenated Filters: 512 Channel Size: 3	Yes, Layer-4	Yes, Layer-5	1
Max-Pool	Channel Size: 2 Padding: Same	Yes, Layer-5	No	1
Primary capsule	Number of capsules: 32 Number of channels: 16	Yes, Layer-6	Yes, Layer-6	1
Digit Capsule	Number of capsules: 10&40 Number of channels: 16	Yes, Layer-7	Yes, Layer-7	-
FC Layer -1	Number of Neurons: 4096	Yes, Layer-8	Yes, Layer-8	-
FC Layer-2	Number of Neurons: 4096	Yes, Layer-9	Yes, Layer-9	-
FC layer - 3	Number of Neurons: 5	Yes, Layer-10	Yes, Layer-10	-
Softmax Function		Yes, Layer-11	Yes, Layer-11	-

In 1D CapsNet, when the point cloud dataset is sent as an input to the neural network it is presented as a 1D array. The acquired information can be considered similar to the input. Therefore, it becomes possible to recognise the 3D objects even after the networks have been trained using a 1D array (this is point data from a sampled 3D surface of the point cloud dataset which is represented as a 1D array). Also, the number of total trainable parameters decreases because of the 1D array technique. In this technique, the point cloud dataset is shown to CapsNet in chunks of points instead of as a whole image, i.e., when the network is trained then the 3D data (which is in point cloud form) before entering the first layer of the neural network gets converted into a 1D array. Therefore, the number of points that is required to be trained at once is specified. This implies that no dimensional input is required for training which directly reduces the trainable parameters substantially.

V. Results

Table 2 shows the benchmark accuracies produced by the different neural networks on ModelNet-10 and ModelNet-40 datasets, including the accuracies produced by the architectures presented in this paper as well as the total number of parameters required to be trained by each network. The accuracy produced by 1D CapsNetA, 1D CapsNetB and PointCapsNet on ModelNet-10 (indoor objects only) are 92.03%, 91.48% and 89.43% respectively, whilst the accuracies on ModelNet-40 are 91.04%, 89.97% and 83.4% respectively. The presented architectures have surpassed many other state-of-the-art

This is an accepted author's manuscript of Basu, A., Kaewrak, K., Petropoulakis, L., Di Caterina, G., & Soraghan, J. (Accepted/In press). 3-Dimensional object recognition using 1-dimensional capsule neural networks. Paper presented at IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Hyderabad, India.

architectures in terms of accuracy, although they are not the top performers as some other architectures display slightly better accuracy. This additional accuracy, however, is achieved at the cost of requiring a lot more parameters to train (anything between twice to 250 times as many parameters) with inevitable lengthier training times.

By contrast, the 1D CapsNet architectures have considerably fewer trainable parameters – i.e., just over 1 million parameters for each of the 1D CapsNet architectures. Even the top performer, RotationNet still has 5 times increase in trainable parameters compared to the 1D CapsNet. It should also be noted that many of these neural networks are based on completely different concepts. The kind of inputs used to train these neural networks and the concepts involved in their development are very different from the ones used in this work. In particular, RotationNet is a CNN that uses 2-D datasets of objects taken from one or more than one angle. This increases the size of the dataset. Therefore, as already mentioned, the efficiency of training this network decreases. An additional cost both in time and effort is the development of these databases, assuming of course that three 2-D views for each object used are always available. The implementation of work presented in this paper is done using 4 Nvidia 1080Ti GPUs.

Table 2. Accuracies produced by different Neural Networks on ModelNet-40 and ModelNet-10

Neural Network	ModelNet-40 Accuracy (%)	ModelNet-10 Accuracy (%)	Total Number of Parameters
1D CapsNetB	**91.04**	**92.03**	**1 million (approx.)**
1D CapsNetA	**89.97**	**91.48**	**1 million (approx.)**
PointCapsNet	**83.4**	**89.43**	**1million (approx.)**
CapsNet Sabour et al. [13]	30.39	37.7	160 million (approx.)
3D CapsNet Architecture-1 [10]	88.67	91.48	200 million (approx.)
3D CapsNet Architecture-2 [10]	89.66	91.37	200 million (approx.)
3D Capsule [11]	92.7	94.7	150 million (approx.)
3D-CapsNets [12]	82.73	93.08	250 million (approx.)
RotationNet [14]	97.37	98.46	5 million (approx.)
Achlioptas et al. [9]	84.5	95.4	2 million (approx.)
PointNet	89.2 [16]	77.6 [22]	880K (approx.)
MVCNN [15]	90.1%	-	5 million (approx.)

Conclusion

A 3D object recognition using neural networks is a very important topic in computer vision because it enables high-performance recognition rates. However, in developing 3D neural networks, the problem which arises is the total number of trainable parameters which tend to grow appreciably. This results in very long training times and often reduced rates of performance to what it could have been achieved with better optimization. The problem is compounded when there is limited availability of resources and/or when datasets are relatively small. Therefore, developing accurate 3D object recognition systems with a fewer number of trainable parameters is both desirable and efficient.

The work and implementations presented in this paper show how the challenging task of recognising 3D objects with fewer trainable parameters, smaller datasets and good levels of accuracy can be achieved. The approach shown here was to develop two 1D CapsNets architectures differing only in the location of a Max Pool layer. The presented architectures are trained on a point cloud dataset. As indicated in this work a specific PointNet architecture is not necessary to extract data from point cloud datasets. In this case, this was achieved by a combination of convolutional layers, a Max Pool layer and capsules without the need for conversion to 3D voxel grids.

The two architectures, 1D CapsNetA and 1D CapsNetB were developed for recognising 3D objects using the point cloud dataset (i.e., the inputs are in point form). The 1D CapsNetB produced better accuracy on both ModelNet-10 and ModelNet-40 datasets. This architecture utilizes a Max Pool layer just after the first two concatenated convolutional layers and then the subsequent output of the layer is carried forward by the remaining convolutional layers. This arrangement worked better than the 1D CapsNetA architecture which has the Max Pool layer after the fourth and last concatenated convolutional layer. This indicates that the aggregation of point cloud dataset works better when it is done earlier in the process with the re-extraction of information of the aggregated output by the Max Pool layer subsequently being processed by the following two concatenated convolutional layers before the output is sent to the capsule layers.

Moreover, good accuracy was produced on a very small dataset i.e., only using 2589 images. This is because of the proper orientation extraction by the capsules. It should also be noted that the total number of trainable parameters for both the 1D CapsNet architectures are several orders smaller when compared to the architectures using capsules and also the 3D CapsNets. This results in a compact and more efficient network requiring fewer resources to train whilst making the training and learning faster.

Notably, the PointCapsNet architecture has produced comparable outputs. However, in the case of the ModelNet-40 dataset, it failed to produce reasonably high accuracy. The reason behind this could be attributed to the presence of the MLP in the initial layers for feature extraction instead of a convolutional layer.

It must be noted that some neural networks have produced better results than the neural network architectures presented in this paper. However, it must be highlighted that the inputs of these networks are different than for the architectures presented in this paper. These different inputs compromise the objectives that this paper tries to address. For example, RotationNet, which has the highest accuracy on both ModelNet-10 and ModelNet-40 datasets, is trained using 2-D images taken from different viewpoints of 3D objects. This assumes that multiple angle images for objects will always be available – not generally true. It also uses a much larger dataset size. This goes against the objective of obtaining better accuracy from smaller datasets. RotationNet also happens to have 5 times as many trainable parameters than the 1D CapsNet architectures thus affecting the training and learning speed of the neural network.

In conclusion, then, the 1D CapsNet architectures presented in this paper have fewer total number of trainable parameters, can acquire information from the extracted features using convolutional and Max Pool layers, can recognize the objects in 3D (specifically the indoor home objects) and have the ability to produce good accuracy on smaller datasets. Overall, then this indicates a compact, less complex and more efficient neural network structure that can perform at the highest level of 3D object recognition.

References

[1] A. Basu, L. Petropoulakis, G. Di Caterina, and J. Soraghan, "Indoor home scene recognition using capsule neural networks," *Procedia Computer Science,* vol. 167, 2020, pp. 440-448.

[2] A. Basu, K. Kaewrak, L. Petropoulakis, G. Di Caterina, and J. J. Soraghan, "Modified Capsule Neural Network (Mod-CapsNet) for indoor home scene recognition," *International Joint Conference on Neural Network,* 2020, pp. 1-6.

[3] A. X. Chang *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012,* 2015.

[4] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis & Machine Intelligence,* no. 6, 2017, pp. 1137-1149.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *IEEE international conference on computer vision,* 2017, pp. 2980-2988.

[7] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *IEEE conference on computer vision and pattern recognition*, 2017, pp. 652-660.

[8] J. Li, B. M. Chen, and G. Hee Lee, "So-net: Self-organizing network for point cloud analysis," *IEEE conference on computer vision and pattern recognition,* 2018, pp. 9397-9406.

[9] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," *International conference on machine learning,* 2018, pp. 40-49.

[10] A. Ahmad, "Object Recognition in 3D data using Capsules," *PhD Thesis, Syracuse University,* 2018.

[11] A. Cheraghian and L. Petersson, "3dcapsule: Extending the capsule architecture to classify 3d point clouds," *IEEE Winter Conference on Applications of Computer Vision,* 2019, pp. 1194-1202.

[12] R. Lambert, "Investigation: Capsule Nets for Content-Based 3D Model Retrieval," *Dimension,* vol. 30, p. 30x30x30.

[13] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Advances in neural information processing systems*, 2017, pp. 3856-3866.

[14] A. Kanezaki, Y. Matsushita, and Y. Nishida, "Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5010-5019.

[15] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," *IEEE international conference on computer vision*, 2015, pp. 945-953.

[16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems,* 2017.

[17] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," *IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 2019, pp. 9621-9630.

[18] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on χ-transformed points," *Advances in neural information processing systems,* 2018, pp. 828-838.

[19] J. Jiang, D. Bao, Z. Chen, X. Zhao, and Y. Gao, "MLVCNN: Multi-Loop-View Convolutional Neural Network for 3D Shape Retrieval," *AAAI Conference on Artificial Intelligence,* 2019, vol. 33, pp. 8513-8520.

[20] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, " Linked Dynamic Graph CNN: Learning through Point Cloud by Linking Hierarchical Features," *International Conference on Mechatronics and Machine Vision in Practice*," 2021, pp. 7-12.

[21] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," *International conference on artificial neural networks,* 2011, pp. 44-51.

[22] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla, and J. Azorin-Lopez, "Pointnet: A 3d convolutional neural network for real-time object class recognition," *International joint conference on neural networks,* 2016, pp. 1578-1584.