ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2.
Available from http://www.i6doc.com/en/.

From Three to Two Dimensions: 2D Quaternion Convolutions for 3D Images

Valentin Delchevalerie[1], Benoît Frénay[1] and Alexandre Mayer[2]

1- University of Namur - NaDI/naXys institutes - Faculty of Computer Science
Rue Grandgagnage 21, 5000 Namur - Belgium

2- University of Namur - naXys institute - Department of Physics
Rue de Bruxelles 61, 5000 Namur - Belgium

Abstract. In fields like biomedical imaging, it is common to manage 3D images instead of 2D ones (CT-scans, MRI, 3D-ultrasound, etc.). Although 3D-Convolutional Neural Networks (CNNs) are generally more powerful compared to their 2D counterparts for such applications, it also comes at the cost of an increase in computational resources (both in time and memory). In this work, we present a new way to build 2D representations of 3D images while minimizing the information loss by leveraging quaternions. Those quaternion CNNs are able to offer competitive performance while significantly reducing computational complexity.

1 Introduction

Convolutional Neural Networks (CNNs) are one of the most powerful tools we have today to manage images. Among the computer vision tasks, a few involve 3D images instead of the more standard 2D ones. For example, tools like CT-scans, MRI or 3D-ultrasound are commonly used in biomedical imaging. Even though CNNs first proved their efficiency with 2D images [1], their extension to 3D images is straightforward. However, while using 3D-CNNs generally leads to better performances when working with 3D images, it comes at the cost of a significant increase in computational resources (both in time and memory). Therefore, there is a real opportunity to save computational resources by developing methods to bring back 3D convolutions to 2D ones. Nonetheless, the key challenge to do this is to build a 2D representation of 3D images that minimizes the loss of information. While 2D representations can be obtained by wrapping slices along one arbitrary dimension with the channels, it leads to a drastic loss of information.

In this work, we present a new way to represent 3D images in a 2D fashion that allows trading the 3D convolutions for 2D ones. This is mainly inspired by the work of Zhou et al. [2] for hyper-spectral images, and builds on their use of quaternions. The 3D images are first transformed into quaternions, where the three complex parts are used to build multiple views of the image along its three respective axes. By also defining the filters as quaternions, it becomes more intuitive for the model to encode correlations between the dimensions, as opposed to a simple scalar representation that treats each dimension independently.

Section 2 presents the state of the art in using 2D convolutions with 3D images. After that, Section 3 provides some theoretical background on quaternions

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

along with the motivations of this work. Section 4 describes our method, called Quaternion-CNN (Q-CNN). Sections 5 and 6 follow up with the experimental setup and the results, respectively. Finally, Section 7 concludes this work.

2 Related Work

For now, two main strategies exist for applying 2D convolutions to 3D images. The first one is the most straightforward and consists in wrapping slices along one of the three dimensions with the channels. However, by doing so, the local connectivity prior (one of the main reasons for the efficiency of CNNs) is not satisfied anymore for one arbitrary dimension. In other words, slices along this dimension are considered as independent and the 3D structure is partially broken, resulting in a loss of information. Those models are generally referred to as 2.5D-CNNs [3]. The advantage is that it is straightforward to use and leads to both significant speed ups and improvements in memory usage. However, it is generally at the cost of lower performances due to the loss of information. Still, it is often the most used method as in addition to the significant speed up, it also makes it easy to take advantage of standard pre-trained 2D models.

The second approach is introduced by Yang et al. [4] who propose Axial-Coronal-Sagittal (ACS) convolutions. Channels are split into three groups for different views of the 3D structure (axial, coronal and sagittal), and 2D filters are extended along a third dimension. Although it makes it possible to use 2D pre-trained models, ACS CNNs are still performing 3D convolutions in their inner working, preventing any gain in the number of numerical operations. Indeed, in their work, Yang et al. specify that other early studies [3, 5] already investigated the use of three views CNNs with 2D convolutions, but those methods generally have trouble into capturing large 3D contexts.

Finally, many hybrid approaches like [6, 7] propose to build CNN with both 2D convolutions (for which the weights could be loaded from pre-trained models) and 3D ones (randomly initialized). The different representations are learned in parallel and merged at the end in a fusion network that performs the considered task. However, from a numerical performance perspective, the use of 3D convolutions still constitutes a significant bottleneck.

3 Background on Quaternions and Motivations

Quaternions are a generalization of complex numbers. A quaternion is defined by three complex parts and one real one. Conventionally, a quaternion is represented as $z = a + ib + jc + kd$, where $a, b, c, d \in \mathbb{R}$ and i, j, k are such that $i^2 = j^2 = k^2 = ijk = -1$ and $ij = -ji = k$; $ki = -ik = j$; $jk = -kj = i$.

For several years, the use of quaternions in deep learning has been considered through different approaches [2, 8, 9]. The key motivation is the expressiveness of quaternion algebra when it comes to extract complex features that are obtained through the correlation of several views of the data. Indeed, thanks to the inter-dependency of the complex parts, cross-correlations between the different views

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

are more natural to express than when using a scalar representation.

As an example, Zhu et al. [9] show that using quaternions to store the red, green and blue channels of colored images can be beneficial in CNNs. In this case, each image is considered as one whole object, and the channels are no longer independent. By also defining filters as quaternions, convolutions inherently capture complex interactions across channels, which might be missed when they are processed independently. Next to that, and with a similar motivation, Zhou et al. [2] leverage the use of quaternions to build a better representation of 2D multi-spectral images. Their idea is to use the different parts a, b, c and d to store several views of the data. Namely, a stores a local representation of the data by using 3D average pooling, b and c store multiple linear projections along the two spatial dimensions and d along the spectral one. Thanks to the use of quaternions, features from those different views can be cross-correlated to ease the extraction of meaningful features. However, their work remains limited to hyper-spectral images that only involve two spatial dimensions.

4 Proposed Method

Our method is motivated by the points highlighted in the previous sections. The idea builds on the works of Yang et al. [4] (for the decomposition of 3D images in ACS views) and Zhou et al. [2] (in the way quaternions are exploited) in order to extend the last one to the case of three spatial dimensions. This section presents how 3D images are converted into 2D quaternion images, as well as how the CNN components are adapted to handle this representation.

4.1 Preprocessing Layer

A preprocessing layer is defined to convert 3D images into a quaternion ACS representation. For each of the three spatial dimensions, trainable linear projections are performed to generate multiple 2D images. For a grayscale 3D image of shape $W \times H \times D$, this preprocessing will output three 2D views of shape $C_D \times W \times H$, $C_H \times W \times D$ and $C_W \times H \times D$, respectively, where C_D, C_H and C_W are numbers of channels. By considering $W = H = D$, each one of those three views can be stored as a complex part of a quaternion matrix (leaving the real part initialized to zeroes). In this situation, we fix the number of linear projections to the initial size of the input images ($C_W = C_H = C_D = W$), for convenience. Although the number of input values is three times bigger, it now involves 2D views that can be treated with 2D convolutions. Thus, it leads to a significant gain deeper in the network, as each feature map becomes a quaternion matrix of shape $C \times W \times W$, while it becomes of shape $C \times W \times W \times W$ in 3D-CNNs.

4.2 CNN Components

In order to handle the quaternion representation of 3D images, the different components of CNNs have to be adapted. Regarding convolutions, filters are

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

defined as quaternions and a separable quaternion convolution (as proposed in [2]) between an input $\mathcal{F} = F^{(r)} + iF^{(i)} + jF^{(j)} + kF^{(k)}$ and a filter $\mathcal{K} = K^{(r)} + iK^{(i)} + jK^{(j)} + kK^{(k)}$ can be defined as

$$\begin{aligned}\mathcal{I} = \mathcal{K} * \mathcal{F} = &\left(I^{(r)} + I^{(i)} + I^{(j)} + I^{(k)}\right) + i\left(I^{(i)} - I^{(r)} - I^{(k)} + I^{(j)}\right) \\ &+ j\left(I^{(j)} + I^{(k)} - I^{(r)} - I^{(i)}\right) + k\left(I^{(k)} - I^{(j)} + I^{(i)} - I^{(r)}\right),\end{aligned} \tag{1}$$

where $I^{(\cdot)} = K^{(\cdot)} * F^{(\cdot)}$ is a vanilla convolution. Therefore, separable quaternion convolutions are the result of the combination of four vanilla convolutions. Note that the idea to extend the dense layers is similar, by also performing the same combinations between the different parts of the quaternions.

Regarding the non-linear activation, batch-normalization and pooling layers, the extension to quaternion values is straightforward by considering the different parts independently. Finally, after the final output layer, only the real part of the quaternions is kept (the predicted probabilities have to be real numbers) as it empirically lead to better results than using squared modulus. Note that even though the real part is zero for the input, real parts are progressively populated deeper in the network thanks to the presence of mixed terms in Equation (1).

5 Experimental Setup

To assess the potential benefit of using our method, 2.5D, 3D, ACS and Q-CNNs are trained and evaluated on three different toy datasets of 3D biomedical images from MedMNIST3D [10]. MedMNIST3D is a collection of datasets that provides baselines for the classification of various medical 3D images. In this work, we considered OrganMNIST3D, NoduleMNIST3D and SynapseMNIST3D that all contain grayscale images of $64 \times 64 \times 64$ pixels. OrganMNIST3D is made of $1,742$ Abdominal CT images from 11 classes, NoduleMNIST3D of $1,633$ Chest CT images from 2 classes and SynapseMNIST3D of $1,759$ electron microscope images from 2 classes. Training and testing sets are used as provided by MedMNIST3D.

For each model, the architecture presented in Table 1 is used. To allow a fair comparison, a parameter λ is introduced to control the number of filters in each layer so that the numbers of trainable parameters are similar in the different methods. All the convolutions are performed with 3×3 filters and followed by a ReLU activation function. Data augmentation to randomly flip the images along the three axes as well as dropout are used during training. Every training is performed through 10 independent runs on one NVIDIA A100.

6 Results

The results are presented in Table 2. On the first two datasets, Q-CNNs are at least able to achieve state-of-the-art performances and can even be better than 3D and ACS-CNNs while being significantly faster (only marginally slower than 2.5D-CNNs). Note that ACS-CNNs are much slower because among all the

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

Layer	**# C**	2.5D CNN	3D/ACS CNN	Q-CNN (our)
Preprocessing	64	-	-	Q-converter
$2 \times \begin{cases} \textit{Conv.} \\ \textit{Norm.} \end{cases} = \star$	16λ	Conv2d BNorm2d	Conv3d BNorm3d	Conv2d Q-BNorm2d
	32λ	Conv2d BNorm2d	Conv3d BNorm3d	Conv2d Q-BNorm2d
Pool.	-	Avg2d	Avg3d	Q-Avg2d
$\star$	$\times 4$			
G. Pool.	-	G. Avg2d	G. Avg3d	G. Q-Avg2d
Dense	64λ	Linear	Linear	Q-Linear
Dense	n	Linear	Linear	Q-Linear
λ		1.66	1.70 / 1.00	0.80

Table 1: Architecture used for the different methods. C is the number of output channels, n the number of classes and W the size of the input image. λ is the parameter that tweaks the number of filters (# C). The $\star$ symbolizes a convolutional block that is repeated at two different positions in the network.

	OrganMNIST3D		NoduleMNIST3D		SynapseMNIST3D	
Models	*Acc.*	*Time*	*Acc.*	*Time*	*Acc.*	*Time*
2.5D-CNN	58.14 ±3.57	234 ±8	83.51 ±1.46	236 ±7	70.51 ±6.22	244 ±8
3D-CNN	**72.44** ±3.73	383 ±4	85.46 ±1.96	386 ±3	**83.55** ±4.46	410 ±3
ACS-CNN	71.52 ±1.52	994 ±4	83.82 ±2.82	1030 ±5	79.15 ±3.78	1098 ±4
Q-CNN (our)	72.14 ±2.38	236 ±7	**85.64** ±0.80	239 ±7	70.06 ±3.19	252 ±7

Table 2: Mean accuracy (%) and training time (sec) along with the standard deviations for 10 independent runs. Highest performance are highlighted in bold.

filter parameters, many of them are non-trainable (they are the duplication of 2D filters along a new dimension). Therefore, as we compare models with similar number of trainable parameters, ACS-CNNs have to perform a larger number of numerical operations. Regarding the last dataset, observations are different as both 2.5D-CNNs and Q-CNNs seem to struggle and achieve similar performances significantly below those obtained by 3D-CNNs. However, one may notice that ACS-CNNs also achieve lower performances, which may indicate that it is more difficult to break free from the 3D representation for this task.

7 Conclusion and Limitations

In this work, we present a new approach to be able to perform 2D convolutions when working with 3D images with the objective to save computational

ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

resources. This is achieved by drawing inspiration from the works of Yang et al. [4] and Zhou et al. [2] by building a 2D representation of 3D images through the consideration of different views and the use of quaternions.

Even though this work is still a first step toward this direction and future work should be performed to further assess the performances of Q-CNNs (for example, by tackling segmentation tasks), we show promising preliminary results. Q-CNNs are able to achieve better or state-of-the-art performances on two among the three datasets considered, while being significantly more efficient. Indeed, quaternion convolutions inherit from $\mathcal{O}\left(W^2\right)$ time and space numerical complexities of 2D convolutions, while ACS and 3D convolutions exhibit $\mathcal{O}\left(W^3\right)$ ones (W being the spatial resolution of the input). Thanks to the use of linear projections for the conversion of 3D images into 2D quaternion images, the difficulty of 2.5D-CNNs to capture large 3D contexts can be mitigated. Furthermore, the use of quaternions make it easier for the network to capture correlations among the different channels.

Acknowledgments

A.M. is funded by the Fund for Scientific Research (F.R.S.-FNRS) of Belgium. V.D. benefits from the support of the Walloon region with a PhD grant from FRIA (F.R.S.-FNRS). This research used computational resources from Lucia, funded by the Walloon Region under grant agreement n°1910247.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NeurIPS*, volume 25, 2012.

[2] Heng Zhou, Xin Zhang, Chunlei Zhang, et al. Quaternion convolutional neural networks for hyperspectral image classification. *Eng. Appl. Artif. Intell.*, 123:106234, 2023.

[3] Holger R. Roth, Le Lu, Ari Seff, et al. A new 2.5D representation for lymph node detection using random sets of deep convolutional neural network observations. *Med. Image Comput. Comput. Assist. Interv.*, 17(Pt 1):520–527, 2014.

[4] Jiancheng Yang, Xiaoyang Huang, Yi He, et al. Reinventing 2d convolutions for 3d images. *IEEE J BIOMED HEALTH*, 25(8):3009–3018, 2021.

[5] Adhish Prasoon, Kersten Petersen, Christian Igel, et al. Deep Feature Learning for Knee Cartilage Segmentation Using a Triplanar Convolutional Neural Network. In *MICCAI*, pages 246–253, 2013.

[6] Yingda Xia, Lingxi Xie, Fengze Liu, et al. Bridging the Gap Between 2D and 3D Organ Segmentation with Volumetric Fusion Net. In *MICCAI*, pages 445–453, 2018.

[7] Hao Zheng, Yizhe Zhang, Lin Yang, et al. A new ensemble learning framework for 3D biomedical image segmentation. In *Proceedings of AAAI'19*, pages 5909–5916, 2019.

[8] Chase Gaudet and Anthony Maida. Deep Quaternion Networks, 2018. arXiv:1712.04604.

[9] Xuanyu Zhu, Yi Xu, Hongteng Xu, et al. Quaternion convolutional neural networks. In *ECCV*, September 2018.

[10] Jiancheng Yang, Rui Shi, Donglai Wei, et al. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.