Quaternion-valued Correlation Learning for Few-Shot Semantic Segmentation

Zewen Zheng, Guoheng Huang*, Xiaochen Yuan*, Chi-Man Pun*, Hongrui Liu, and Wing-Kuen Ling,

Abstract—Few-shot segmentation (FSS) aims to segment unseen classes given only a few annotated samples. Encouraging progress has been made for FSS by leveraging semantic features learned from base classes with sufficient training samples to represent novel classes. The correlation-based methods lack the ability to consider interaction of the two subspace matching scores due to the inherent nature of the real-valued 2D convolutions. In this paper, we introduce a quaternion perspective on correlation learning and propose a novel Quaternion-valued Correlation Learning Network (QCLNet), with the aim to alleviate the computational burden of high-dimensional correlation tensor and explore internal latent interaction between query and support images by leveraging operations defined by the established quaternion algebra. Specifically, our QCLNet is formulated as a hyper-complex valued network and represents correlation tensors in the quaternion domain, which uses quaternion-valued convolution to explore the external relations of query subspace when considering the hidden relationship of the support sub-dimension in the quaternion space. Extensive experiments on the PASCAL-5^i and COCO-20^i datasets demonstrate that our method outperforms the existing state-of-the-art methods effectively. Our code is available at https://github.com/zwzheng98/QCLNet and our article "Quaternion-valued Correlation Learning for Few-Shot Semantic Segmentation" was published in IEEE Transactions on Circuits and Systems for Video Technology, vol. 33,no.5,pp.2102-2115,May 2023,doi: 10.1109/TCSVT.2022.3223150.

Index Terms—Few-shot learning, semantic segmentation, correlation learning, quaternion-valued convolution.

I. INTRODUCTION

Fig. 1. Illustration of the difference between Center-pivot convolution [11], [12] and Quaternion-valued convolution in correlation learning. For ease of visualization, we show the 2D correlation for matching pixels across a query and support image. $\mathbb{R}$ to $\mathbb{H}$ means **R**eal-value to **H**yper-complex valued. Each black wire that connects two different pixel locations represents a single weight of the convolutional kernel. To efficiently filter the 2D correlation, the center-pivot convolution decomposees the 4D correlation learning into two independent subspace learning. Specifically, it factors the 5×5 filter into support **(a)** and query **(b)** sub-dimension convolution kernels, which perform two different convolutions on separate 2D subspaces and thus cannot consider the interaction of the two subspace matching scores. In contrast, with the aggregation and encapsulation of support subspace of correlation maps into quaternion space (i.e., hypercomplex numbers), our method enhanced interaction between two subspaces by performing a double learning: **(1)** the convolution operator learns external/global relations among the elements of the query spatial dimension, **(2)** while the Hamilton product accomplishes the learning of the support subspace **(c)**.

NEURAL network architectures such as Convolutional Neural Networks (CNNs) have made unprecedented progress in semantic segmentation. However, strong semantic segmentation models [1], [2] rely heavily on large-scale datasets with dense annotation, and models trained on such datasets often fail to handle novel object categories. As a promising direction, few-shot segmentation (FSS) is proposed to tackle the above challenge. It aims to train a model on a dataset with sufficient data and quickly adapt to the segmentation prediction of novel classes by using only a few annotations. Specifically, models are episodically trained on base classes with sufficient data samples and then located the target objects on novel classes based on the semantic information provided by the support set.

Fueled by the success of few-shot classification [3], [4], current FSS models [5]–[10] often use a metric-learning based framework, which utilizes the prototypes calculated from the support features to guide the query branch for semantic segmentation. However, performing metric learning on the base dataset with abundant annotated samples inevitably introduces a bias towards the seen classes rather than being ideally class-agnostic. More specifically, this learning mechanism easily forces the model to 'remember' objects outside the base class as negative samples and the embeddings of latent novel classes are over-smoothed.

Such a problem can be easily solved by correlation learning methods in semantic correspondence task, which aims to construct the pixel-wise correlation between semantically similar images and exploring their internal latent relationships. Therefore, reformulating the FSS task as a semantic correlation learning problem can help the FSS model to capture more generic patterns, thereby improving the generalization. Recent works in correlation learning [11], [13] utilize high-dimensional convolutions to aggregate correlation tensors and show significant efficiency in learning accurate relations. However, there are several challenges in applying this real-valued high-dimensional convolution in FSS. First, the direct use of high-dimensional 4D convolution [13], [14] requires a large number of parameters, which increases the computational burden and is contrary to the original intention of lightweight design in FSS to ensure the generalization ability of the model. Second, there have been some attempts in utilizing

center-pivot 4D convolution [11], [12] to reduce the amount of high-dimensional convolution parameters and achieve good performance. Despite their success, we notice that the design of this method still presents problems. Specifically, they factor the 4D correlation learning into two independent 2D subspace learning and simply employ two different 2D real-valued convolutions. Figure 1 visualizes this factorization, which cannot consider the interaction of the two subspace matching scores.

In light of this, we argue that an ideal model of correlation learning should be structured as a union dual-space learning process and be able to maintain the internal dependencies within the two subspaces. Therefore, real-valued convolution is not suitable for correlation learning due to its inherent nature of only dealing with a single feature space (i.e., spatial and channel relations). Hyper-complex valued convolutional neural networks solved this problem by introducing multidimensional algebra to CNN. As in Quaternion Convolutional Neural Networks (QCNN), by encapsulating the extra space information to the quaternion space, the Hamilton product allows QCNN to encode both internal relations that exist inside quaternion space and global relations of outside feature space at the same time.

In this paper, we propose a Quaternion-valued Correlation Learning Network (QCLNet). We move beyond real-valued space to explore the properties of quaternion algebra, e.g., *Hamilton product*. As illustrated in Figure 1, by combining with the convolution operator, it allows the processing of support subspace of the correlation as a unique quaternion to perform a **double learning**: **(1)** the convolution operator learns external/global relations among the elements of the query spatial dimension, **(2)** while the Hamilton product accomplishes the learning of the support subspace. Specifically, to overcome the limitations of the computational burden and encapsulate the support information to the quaternion space, we propose a Correlation Aggregation Module (CAM) and utilize it to aggregate sparse information in high-dimensional correlation tensor. After that, we encapsulate the support spatial subspace of correlation maps in quaternion space (i.e., hypercomplex numbers) and propose a Quaternion Correlation Learning Module (QCLM) that consists of a series of quaternion-valued convolution and quaternion normalization (QN) to explore the external relations among the elements of the query spatial sub-dimension when considering the hidden relationship of the support subspace in the quaternion space. With correlation learning in the quaternion domain, the internal (i.e., the relations that exist inside support set) and external relations (i.e., edges or shapes features in query set) are learned simultaneously and thus the interactions of matching scores between query and support set are fully explored. As the interactions between the support and query set have been extracted, we further propose the Episodic Readout Module (ERM), which transforms quaternion features into real-valued features and utilizes low-level query features to refine the segmentation results. The contributions of this work are summarized as follows:

- We propose the Quaternion-valued Correlation Learning Network (QCLNet), which explicitly explores interac-

tions of matching scores between query and support images by leveraging operations defined by the established quaternion algebra.
- We introduce a quaternion perspective on correlation learning and propose a novel quaternion correlation learning module, QCLM, which encapsulates the support sub-dimension in quaternion space and performs quaternion-valued convolution with our proposed theoretically correct quaternion normalization (QN) to explore the interaction of two subspaces of the correlation.
- We propose a correlation aggregation module (CAM) and episodic readout module (ERM) to aggregate sparse information of the correlation tensors and adaptively refine the segmentation results with the low-level query features.
- Our method achieves better performance and effectiveness than other state-of-the-art methods on two FSS benchmark datasets: PASCAL-5^i and COCO-20^i.

II. RELATED WORK

A. Semantic Segmentation

Semantic segmentation is a fundamental and challenging task that has gained interest in the computer vision community for decades due to its ability to provide pixel-wise dense semantic prediction [1], [15]–[17]. Since the great success of fully convolutional neural networks in the field of semantic segmentation, various networks , such as Deeplab [18], PSP-Net [2], UNet [1] and SegNet [15] have been proposed in this field. Contextual information provides surrounding hints to help identify individual elements, thus later works contribute many benchmark blocks, such as the pyramid pooling module [2], deformable convolution [19], non-local module [20] to help enlarge the receptive field of the model and achieved good performance. However, powerful segmentation models cannot be extended to unseen class segmentation scenarios without updating the parameters of the model.

B. Few-Shot Segmentation

Few-Shot Segmentation (FSS) requires the model to quickly segment the target region in the input image with only a few annotated samples. Almost all existing models use two-branch architecture design to implement meta-learning. OSLSM [21] is the pioneering work for FSS, which includes two branch structures: conditional branch and segmentation branch. The conditional branch is used to generate classifier weights for the query image of each task. Afterward, global or multiple prototype-based methods were designed under these two-branch paradigm, representative models include PANet [9], PMMs [8], CANet [6], PPNet [22] and PFENet [10]. PANet [9] uses prototype alignment regularization to provide high-quality prototypes that are representative of each semantic class. The work of [23] generalizes FSS to a multi-class task and mainly studies the application of incremental learning in few-shot tasks. However, as methods based on prototypes have apparent limitations, e.g., performing metric learning on the base dataset inevitably introduces a bias towards the seen classes rather than being ideally class-agnostic. Recent works

[12] attempted to utilize efficient 4D convolution consisting of two decoupled 2D convolutions to fully exploit the multi-level correlations. However, this correlation-based methods still face the challenge of lacking the ability to simultaneously consider the internal interaction of the two subspaces due to the inherent nature of the real-valued 2D convolutions.

C. Semantic Correspondences

In recent years, finding dense semantic correspondences has been studied extensively in low-level vision. The objective of semantic correspondence is to find reliable correspondences between a pair of images with challenges of large intra-class variations [11], [13], [14], [24]. This setting is very similar to few-shot semantic segmentation, which aims to use the semantic features of the support set to guide the query branch for semantic segmentation. Rocco *et al.* [13] introduce the neighborhood consensus network that uses 4D convolution to learn local geometric constraints between neighboring correspondences, and thus requires a large number of parameters. Following the work, recent methods [14], [25] also adopt 4D convolution in a similar manner. The work of [11] resolves the former problem (quadratic complexity) by separating a 4D convolution into two center-pivot 2D kernels and downsampling the 4D cost-volume to maintain small memory footprints. Despite the center-pivot 4D convolution reducing the computational burden caused by the use of high-dimensional convolution, the hidden internal relations of matching scores are unfortunately unexplored. In our work, we use quaternion neural networks to fully explore the interactions of matching scores while yielding substantial improvements in parameter size.

D. Complex and Quaternion Networks

In various deep learning application areas [26]–[28], such as images, 3D audio, multi-sensor signals or human-pose estimation, some efforts have been made to extend real-valued neural networks to other number fields. Complex-valued neural networks [29] or quaternion neural networks [26], [30]–[32] (QNN) have been proposed to encapsulate multidimensional input features. In [30], a deep quaternion network is proposed, which simply replaces the real multiplications with quaternion ones. The work of [31], [33] further explores the application of QNN and QCNN to image processing, where they use Hamilton product to embed the three components (R,G,B) of a given pixel in a quaternion and maintain its internal dependencies in the subsequent convolution process. Similarly, we believe that few-shot correlation learning should be constructed as a dual-space learning process since it needs to consider both support and query subspace information. Therefore, instead of introducing multi-dimensional algebra to maintain the structural dependence of the three components (R,G,B), our approach mainly aims to achieve correlation learning by introducing QCNN to explore the external relations of query subspace when considering the hidden relationship of the support sub-dimension in the quaternion space.

TABLE I
THE DEFINITION OF NOTATIONS.

Notations	Description
$\mathcal{D}_{\text{base}}$, $\mathcal{D}_{\text{novel}}$	Base data, novel data
$\mathbf{F}^s$, $\mathbf{F}^q$	Support and query feature given in (7)
$\mathbf{C}_p$	Correlation maps given in (9)
$\mathbf{k}$	Separable 4D convolution kernel
$\mathbf{u}, \mathbf{x}$	Query and support 2D spatial coordinates
$\Psi(\cdot)$	A set of neighborhood coordinates centered on u and x
$\mathcal{F}_{\text{p}}^{\text{ca}}(\cdot)$	Correlation aggregation module defined in (11)
$\boldsymbol{Q}_p$	Quaternion feature given in (12)
$\mathbf{W}_q$	Quaternion convolution weights
$(\cdot)^{\text{H}}$	Conjugate transpose operator
$\tilde{\mathbf{C}}_{\mathbf{qq}}$	Augmented covariance matrix given in (16)
$\text{QN}(\cdot)$	Quaternion normalization defined in (18)
$\boldsymbol{\mu}_q$, $\boldsymbol{\sigma}^2$	Quaternion mean given in (19) and variance given in (20)
$\mathcal{F}_{\text{p}}^{\text{qcl}}(\cdot)$	Quaternion convolutional block defined in (21)
$\text{up}_{[\times 2]}(\cdot)$	Upsampling 2x operator
$\mathbf{F}_r$	Real-valued feature given in (23)
$\mathcal{G}(\cdot)$	Global average pooling
$\mathcal{P}_i(\cdot)$	Linear projection
$\tilde{\mathbf{M}}_q$	Predicted mask

III. QUATERNION ALGEBRA

This section introduces the necessary background of quaternion for this paper. Quaternion is a kind of hypercomplex number of rank 4, being a direct non-commutative extension of complex-valued numbers. Along with Hamilton products, quaternion algebra forms the crux of our proposed approaches.
Quaternion A quaternion Q in the quaternion domain $\mathbb{H}$, i.e., $Q \in \mathbb{H}$, can be represented as:

$$Q = r + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{1}$$

where r, x, y, and z are real numbers, and $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ are the quaternion unit basis. In a quaternion, r is the real part, while $x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ with $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$ is the imaginary part. A pure quaternion is a quaternion whose real part is 0, resulting in the vector $Q = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$. Operations on Quaternions are defined in the following.
Addition The addition of two Quaternions is defined as:

$$\begin{aligned}Q + P = Q_r + P_r + (Q_x + P_x)\,\mathbf{i} \\ + (Q_y + P_y)\,\mathbf{j} + (Q_z + P_z)\,\mathbf{k},\end{aligned} \tag{2}$$

where Q and P with subscripts denote the real value and imaginary components of Quaternion Q and P.
Scalar Multiplication The Multiplication with scalar α is defined as:

$$\alpha Q = \alpha r + \alpha x\mathbf{i} + \alpha y\mathbf{j} + \alpha z\mathbf{k}, \tag{3}$$

Conjugate The conjugate Q^* of Q is defined as:

$$Q^* = r - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}, \tag{4}$$

Norm The unit Quaternion $Q^{\triangleleft}$ is defined as:

$$Q^{\triangleleft} = \frac{Q}{\sqrt{r^2 + x^2 + y^2 + z^2}}, \tag{5}$$

Fig. 2. Overview of our Quaternion-valued Correlation Learning Network (QCLNet), which consists of correlation aggregation module, quaternion correlation learning module and episodic readout module. Form the query and support feature maps $\mathbf{F}^q$, $\mathbf{F}^s$, their matching scores are computed and stored in the 4D correlation tensor $\mathbf{C}_p$. Then the support spatial dimensions of correlation tensor are gradually reduced by correlation aggregation layers. In order to use the operations defined by the established quaternion algebra for correlation learning, we transform the encoded correlation $\widetilde{\mathbf{C}}_p$ into quaternion features $\boldsymbol{Q}_p$ and further use the quaternion correlation learning layers to produce the output quaternion feature $\widehat{\boldsymbol{Q}}_p$.

Hamilton Product The Hamilton product is used to replace the standard real-valued dot product, which represents the multiplication of two quaternions Q and P. It is defined as:

$$
\begin{aligned}
Q \otimes P = &(Q_r P_r - Q_x P_x - Q_y P_y - Q_z P_z) \\
&+ (Q_x P_r + Q_r P_x - Q_z P_y + Q_y P_z)\,\mathbf{i} \\
&+ (Q_y P_r + Q_z P_x + Q_r P_y - Q_x P_z)\,\mathbf{j} \\
&+ (Q_z P_r - Q_y P_x + Q_x P_y + Q_r P_z)\,\mathbf{k},
\end{aligned}
\tag{6}
$$

which intuitively encourages inter-latent interaction between quaternion Q and P. Therefore, hamilton product plays a crucial role in quaternion neural networks. As illustrated in Figure 4, the quaternion-weight components are shared through multiple quaternion-input parts during the Hamilton product, exploring hidden relations within elements. In this work, we use Hamilton product extensively for correlation learning, which is at the heart of the better interaction ability of FSS.

IV. METHOD

We adopt the meta-learning setting to conduct FSS. Typically, in FSS, given two disjoint image sets $\mathcal{D}_{\text{base}}$ and $\mathcal{D}_{\text{novel}}$ ($\mathcal{D}_{\text{base}} \cap \mathcal{D}_{\text{novel}} = \varnothing$), models are required to learn the correlation interaction on $\mathcal{D}_{\text{base}}$ with sufficient data and test on $\mathcal{D}_{\text{novel}}$. Both $\mathcal{D}_{\text{base}}$ and $\mathcal{D}_{\text{novel}}$ contain several episodes, and each of them is formed by a support set $S = (\mathbf{I}_i^s, \mathbf{M}_i^s)_{i=1}^K$ and a

query set $Q = (\mathbf{I}^q, \mathbf{M}^q)$ of the same class, where $K, \mathbf{I}_i^s, \mathbf{M}_i^s, \mathbf{I}^q$ and $\mathbf{M}^q$ represent the number of shot, the support image, the support binary mask, the query image and the query binary mask respectively. During the training of FSS, the model is optimized to segment the objects in the query image $\mathbf{I}^q$ by taking S and $\mathbf{I}^q$ in each episode (S, Q) as inputs. Segmentation performance is evaluated on $\mathcal{D}_{\text{novel}}$ across all the test episodes.

As most FSS models [6], [8]–[10], [22], the 1-way scenario is our focus in this paper, i.e., each pixel is classified as foreground or background. And we consider the 1-shot setting (i.e., $K = 1$ in S) to clearly illustrate our proposed approach.

A. Overview of QCLNet

We propose a novel FSS framework, Quaternion-valued Correlation Learning Network (QCLNet), as shown in Figure 2, to explore internal latent interaction between query and support images by leveraging operations defined by the established quaternion algebra. In this section, we first briefly describe the multi-channel correlation computation in Section IV-B. In Section IV-C, to transform the high-dimensional correlation tensor to the quaternion space for correlation learning, we propose the correlation aggregation module (CAM) to effectively aggregate the local information of correlation to a global context. Then, in Section IV-D, the quaternion correlation learning module (QCLM) is used to simultaneously

exploit the information of two sub-dimensions (i.e., the support and query) and consider their interaction. Finally, in Section IV-E, a readout module is used to fuse corresponding low-level query feature and refine the segmentation results. As such, QCLNet can be trained in an end-to-end manner and can transfer correlational knowledge from the seen to unseen domain (meta-testing). For convenience, a summary of notations is given in Table I.

B. Multi-channel Correlation Computation

Following the finding by [6], [10], [34], we fix the backbone weights, use a rich of features from the intermediate layers for multi-channel correlation computation. Specifically, for ResNet with layers divided into four groups (*block1-4*), the spatial size of feature maps in each block is the same. Then we use the feature maps after *block2* to produce a sequence of $\mathcal{N}$ pairs of intermediate feature maps $\{(\mathbf{F}_i^q, \mathbf{F}_i^s)\}_{i=1}^{\mathcal{N}}$. Denoting the above process as $\mathcal{B}$, given the support/query images $\mathbf{I}^s/\mathbf{I}^q$, we utilize support mask $\mathbf{M}^s$ to filter out the background area and obtain the intermediate feature maps:

$$\mathbf{F}_i^s = \mathcal{B}(\mathbf{I}^s) \odot \mathcal{R}_i(\mathbf{M}^s), \quad \mathbf{F}_i^q = \mathcal{B}(\mathbf{I}^q), \quad (7)$$

where $\odot$ is element-wise multiplication, and $\mathcal{R}_i(\cdot)$ denotes a function that resizes $\mathbf{M}^s$ along the channel dimension. To obtain the multi-channel correlation $\mathbf{c}_i(\mathbf{x}^q, \mathbf{x}^s) \in \mathbb{R}^{H_q^i \times W_q^i \times H_s^i \times W_s^i}$, we compute the cosine similarity between query and masked support features such that:

$$\mathbf{c}_i(\mathbf{x}^q, \mathbf{x}^s) = \mathrm{ReLU}\left(\frac{\mathbf{F}_i^q(\mathbf{x}^q) \cdot \mathbf{F}_i^s(\mathbf{x}^s)}{\|\mathbf{F}_i^q(\mathbf{x}^q)\| \|\mathbf{F}_i^s(\mathbf{x}^s)\|}\right), \quad (8)$$

where $\mathbf{x}^s, \mathbf{x}^q \in \mathbb{R}^2$ is the pixel coordinate of feature maps $\mathbf{F}_l^s$ and $\mathbf{F}_l^q$ respectively. As done in [34], we record correlation maps computed from the intermediate features in the same blocks to form a multi-channel correlation:

$$\mathbf{C}_p = \mathcal{F}_{\mathrm{concat}}(\{\mathbf{c}_i\}_{i \in \mathcal{N}_p}) \in \mathbb{R}^{H_q^p \times W_q^p \times H_s^p \times W_s^p \times |\mathcal{N}_p|}, \quad (9)$$

where $\mathcal{N}_p$ denotes the CNN layer indices belonging to the same block, $\mathcal{F}_{\mathrm{concat}}(\cdot)$ concatenates the input intermediate features and considers $|\mathcal{N}_p|$ as the feature channel, H_q^p, W_q^p, H_s^p, W_s^p represents the spatial resolution of the multi-channel correlation tensor.

C. Correlation Aggregation Module

1) Motivation: Due to the high-dimensional properties of the correlation C_p, learning the internal interactions between support and query feature requires extremely large computation (quadratic complexity) . As an alternative, the work of [12], [14] stores only the most promising matching scores (i.e., top K or center values) in C_p to effectively reduce the spatially sparse information. However, such an approach would ignore the neighborhood information in C_p, which is proven to be extremely critical for correlation learning [13].

To alleviate the above limitations, we gradually reduce and aggregate the spatial information in high-dimensional correlation tensors C_p by controlling the different strides of the 2D convolution in the separable 4D convolution. After correlation

aggregation, correlation tensor encapsulated into quaternion space is utilized for subsequent correlation learning. Details of CAM are as follows.

2) Module Structure: As shown in Figure 2, the CAM achieve correlation aggregation by applying separable 4D convolution, group normalization (GN) [35], ReLU activation, sequentially. In separable 4D convolution, we use two 2D convolution kernels to perform aggregation of two spatial dimensions (i.e., support and query) with different strides, where the support spatial dimension is reduced to $(2, 2)$ and the query spatial dimension remains the same as (H_q^p, W_q^p). Meanwhile, the separable 4D convolution also projects C_p at separate 2D subspaces to embed the $|\mathcal{N}_p|$ to a fixed dimension D. We now show the factorization of the separable 4D convolution kernel $\mathbf{k}$ into two 2D spatial convolution kernel $\mathbf{k}_s$ and $\mathbf{k}_q$:

$$\begin{aligned}(\mathbf{k} * \mathbf{c})(\mathbf{u}, \mathbf{x}) &= \sum_{\mathbf{x}' \in \Psi(\mathbf{x})} \mathbf{k}_s(\mathbf{x}' - \mathbf{x}) \left[\sum_{\mathbf{u}' \in \Psi(\mathbf{u})} \mathbf{k}_q(\mathbf{u}' - \mathbf{u})\mathbf{c}(\mathbf{u}', \mathbf{x}')\right] \\ &= \mathbf{k}_s(\mathbf{x}) * [\mathbf{k}_q(\mathbf{u}) * \mathbf{c}(\mathbf{u}, \mathbf{x})], \end{aligned}$$
$$(10)$$

where u and x are the query and support 2D spatial coordinates in correlation maps, and $\Psi(\cdot)$ denotes a set of neighborhood centered on 2D spatial coordinate u, x. Overall, the CAM is defined as:

$$\widehat{\mathbf{C}}_p = \mathcal{F}_{\mathrm{p}}^{\mathrm{ca}}(\mathbf{C}_p) \in \mathbb{R}^{H_q^p \times W_q^p \times 2 \times 2 \times D}. \quad (11)$$

D. Quaternion-valued Correlation Learning Module

1) Motivation: Existing correlation-based methods [11], [12] use center-pivot 4D convolutions to squeeze the matching scores of the hypercorrelation while reducing the large computational burden caused by high-dimensional correlation. However, since this method factors the 4D correlation learning into two independent 2D subspace learning, the interaction of the two subspace matching scores cannot be fully considered. Therefore, decomposing the union correlation learning into two independent real-valued convolutions is not ideal.

Inspired by recent hyper-complex convolutional approaches [26], [27], [31], [33], we consider introducing multidimensional algebra (i.e., quaternions algebra specifically) to few-shot correlation learning to alleviate the above problem. After the correlation aggregation, support spatial dimension in the correlation tensor is efficiently aggregated and becomes tractable. Then, by encapsulating the aggregated support subspace of the correlation maps into the quaternion space, Hamiltonian product in quaternion algebra allows quaternion convolution encodes both internal relations that exist inside quaternion space and global relations of outside feature space at the same time.

2) Quaternion-valued Correlation Learning: After the correlation aggregation, each vector of support spatial dimension in the correlation tensor is efficiently aggregated and has larger receptive fields, i.e., $\mathbb{R}^{H_q^p \times W_q^p \times H_s^p \times W_s^p \times |\mathcal{N}_p|} \rightarrow \mathbb{R}^{H_q^p \times W_q^p \times 2 \times 2 \times D}$. To apply quaternion algebra to correlation learning, we propose a quaternion representation that preserves the support spatial dimension of the correlation maps

Fig. 3. Illustration of Quaternion-valued convolution.

$\widehat{\mathbf{C}}_p$ and encapsulates it as a quaternion-valued feature maps $\boldsymbol{Q}_p \in \mathbb{H}^{H_q^p \times W_q^p \times D}$:

$$
\begin{aligned}
\boldsymbol{Q}_p = \ & \widehat{\mathbf{C}}_p(\mathbf{u}, \mathbf{x}_{(0,0)}) + \widehat{\mathbf{C}}_p(\mathbf{u}, \mathbf{x}_{(0,1)})\mathbf{i} \\
& + \widehat{\mathbf{C}}_p(\mathbf{u}, \mathbf{x}_{(1,0)})\mathbf{j} + \widehat{\mathbf{C}}_p(\mathbf{u}, \mathbf{x}_{(1,1)})\mathbf{k},
\end{aligned}
\tag{12}
$$

where $\mathbf{u}$, $\mathbf{x}$ are the 2D spatial coordinate of query and support in the correlation maps $\widehat{\mathbf{C}}_p$. As illustrated in Figure 2, encapsulating support subspace in a quaternion allows treating each vector of query spatial dimension as a single entity and thus to preserving support intra-subspace relations. Therefore, we further utilize quaternion-valued 2D convolution to explore the external relations of query subspace when considering the hidden relationship of the support sub-dimension in the quaternion space. In order to define the convolutional operation in the quaternion domain, we first define the Standard 2D convolution process as:

$$
\widehat{\mathbf{x}} = \phi\left(\mathbf{W}_r \otimes \mathbf{x} + \mathbf{b}_r\right), \tag{13}
$$

where $\mathbf{W}_r \otimes \mathbf{x}$ performs the convolution between the weight matrix $\mathbf{W}_r$ and the input $\mathbf{x}$, $\mathbf{b}_r$ is the bias and $\phi(\cdot)$ is any activation function. Then, we represent the quaternion weight matrix as $\mathbf{W}_q = \mathbf{W}_r + \mathbf{W}_x\mathbf{i} + \mathbf{W}_y\mathbf{j} + \mathbf{W}_z\mathbf{k}$, the quaternion input as $\mathbf{x} = \mathbf{q}_r + \mathbf{q}_x\mathbf{i} + \mathbf{q}_y\mathbf{j} + \mathbf{q}_z\mathbf{k}$ and the quaternion bias as $\mathbf{b}_q = \mathbf{b}_r + \mathbf{b}_x\mathbf{i} + \mathbf{b}_y\mathbf{j} + \mathbf{b}_z\mathbf{k}$. Therefore, $\mathbf{W} \otimes \mathbf{x}$ in Equation 13, is performed by a vector multiplication between two quaternions, i.e., by the Hamilton product:

$$
\begin{aligned}
\mathbf{W}_q \otimes \mathbf{x} = \ & (\mathbf{W}_r * \mathbf{q}_r - \mathbf{W}_x * \mathbf{q}_x - \mathbf{W}_y * \mathbf{q}_y - \mathbf{W}_z * \mathbf{q}_z) \\
& + (\mathbf{W}_x * \mathbf{q}_r + \mathbf{W}_r * \mathbf{q}_x - \mathbf{W}_z * \mathbf{q}_y + \mathbf{W}_y * \mathbf{q}_z)\,\mathbf{i} \\
& + (\mathbf{W}_y * \mathbf{q}_r + \mathbf{W}_z * \mathbf{q}_x + \mathbf{W}_r * \mathbf{q}_y - \mathbf{W}_x * \mathbf{q}_z)\,\mathbf{j} \\
& + (\mathbf{W}_z * \mathbf{q}_r - \mathbf{W}_y * \mathbf{q}_x + \mathbf{W}_x * \mathbf{q}_y + \mathbf{W}_r * \mathbf{q}_z)\,\mathbf{k},
\end{aligned}
\tag{14}
$$

and can be expressed in a matrix form:

$$
\mathbf{W}_q \otimes \mathbf{x} =
\begin{bmatrix}
\mathbf{W}_r & -\mathbf{W}_x & -\mathbf{W}_y & -\mathbf{W}_z \\
\mathbf{W}_x & \mathbf{W}_r & -\mathbf{W}_z & \mathbf{W}_y \\
\mathbf{W}_y & \mathbf{W}_z & \mathbf{W}_r & -\mathbf{W}_x \\
\mathbf{W}_z & -\mathbf{W}_y & \mathbf{W}_x & \mathbf{W}_r
\end{bmatrix}
\begin{bmatrix}
\mathbf{q}_r \\
\mathbf{q}_x \\
\mathbf{q}_y \\
\mathbf{q}_z
\end{bmatrix}.
\tag{15}
$$

We now show that the principle analysis of quaternion-valued correlation learning. A visual explanation of the quaternion-valued 2D convolution is shown in Figure 3, quaternion convolution allows the sharing of filters in channel dimensions, thus forcing each axis of the kernel to exploit the hidden internal relations in the quaternion space. Specifically, a quaternion kernel convolved against $\boldsymbol{Q}_p$ will perform a **double learning**: 1) the convolution operator learns outside (query) global relations among the elements of the query spatial dimension, 2) while the *Hamilton product* accomplishes the inside (support) internal learning of the support subspace (quaternion space). This double learning model, therefore, is particularly suitable for correlation learning in FSS because it can exploit the information of two sub-dimensions (i.e., the support and query) and consider their interaction at the same time.

It is worth noticing the important difference in terms of the number of learning parameters between real and quaternion valued convolution. Denote k as the number of kernels, l as kernel size and c as the number of input channels. In the case of a real-valued convolution layer with k $l \times l \times c$ kernels will have kcl^2 parameters, while to maintain equal k and c the quaternion equivalent has $\frac{k}{4}$ quaternion-valued kernels and $\frac{c}{4}$ quaternion-input channels. Therefore, the quaternion layers with $\frac{k}{4}$ $l \times l \times \frac{c}{4}$ has $\frac{kcl^2}{16} \times 4 = \frac{kcl^2}{4}$ parameters: each kernel has 4 parameter variable elements, namely $\mathbf{W}^r$, $\mathbf{W}^x$, $\mathbf{W}^y$, $\mathbf{W}^z$. In other words, the degrees of freedom in Quaternion-valued convolution is only a quarter of those in its real-space counterpart.

3) Quaternion normalization: The normalization [35], [36] is used to stabilize and speed up the training process of deep neural networks, which has been established as a very effective component in deep learning. The main idea behind normalization is to normalize inputs to have zero mean and unit variance along single or multiple dimensions. We notice that these formulations of normalization only work for real-values. Applying the above normalization to complex or hyper-complex numbers would be difficult since they can not simply translate and scale them such that their mean is 0 and their variance is 1. Therefore, we consider normalizing the quaternions using group normalization [35], which divides the channels into groups and computes within each group the mean and variance for normalization.

However, normalizing within each group introduces problems—*GN would not give equal variance in the multiple components of a quaternion, caused by independent variance calculations of each group.* To overcome this for complex numbers, we use the augmented covariance matrix in [37] to recover the complete second-order statistics in the quaternion domain, which is defined as:

$$
\tilde{\mathbf{C}}_{\mathbf{qq}} = \mathrm{E}\left\{\tilde{\mathbf{q}}\tilde{\mathbf{q}}^{\mathrm{H}}\right\} =
\begin{bmatrix}
\mathbf{C}_{\mathbf{qq}} & \mathbf{C}_{\mathbf{qq}^i} & \mathbf{C}_{\mathbf{qq}^j} & \mathbf{C}_{\mathbf{qq}^k} \\
\mathbf{C}_{\mathbf{qq}^i}^{\mathrm{H}} & \mathbf{C}_{\mathbf{q}^i\mathbf{q}^i} & \mathbf{C}_{\mathbf{q}^i\mathbf{q}^j} & \mathbf{C}_{\mathbf{q}^i\mathbf{q}^k} \\
\mathbf{C}_{\mathbf{qq}^j}^{\mathrm{H}} & \mathbf{C}_{\mathbf{q}^j\mathbf{q}^i} & \mathbf{C}_{\mathbf{q}^j\mathbf{q}^j} & \mathbf{C}_{\mathbf{q}^j\mathbf{q}^k} \\
\mathbf{C}_{\mathbf{qq}^k}^{\mathrm{H}} & \mathbf{C}_{\mathbf{q}^k\mathbf{q}^i} & \mathbf{C}_{\mathbf{q}^k\mathbf{q}^j} & \mathbf{C}_{\mathbf{q}^k\mathbf{q}^k}
\end{bmatrix},
\tag{16}
$$

where $(\cdot)^{\mathrm{H}}$ is the conjugate transpose operator, each $\mathbf{C}$ is the covariance between its two subscripts which represent

Fig. 4. Visual illustration of the quaternion2real module. GAP means global average pooling.

the real, i, j, and k components of $\tilde{q}$ respectively. To make Equation 16 more feasible for practical applications, we further utilize the $\mathbb{Q}$-properness [38] to simplify its computation. The $\mathbb{Q}$-properness implies that the quaternion vector $\mathbf{q}$ is not correlated with its vector involutions $\mathbf{q}^i$, $\mathbf{q}^j$, $\mathbf{q}^k$, i.e., $\mathbf{C_{qq^i}} = \mathbf{C_{qq^j}} = \mathbf{C_{qq^k}} = 0$. Thus, considering a $\mathbb{Q}$-proper quaternion, the covariance in Equation 16 becomes:

$$
\tilde{\mathbf{C}}_{\mathbf{qq}} = \begin{bmatrix} \mathbf{C_{qq}} & 0 & 0 & 0 \\ 0 & \mathbf{C_{q^i q^i}} & 0 & 0 \\ 0 & 0 & \mathbf{C_{q^j q^j}} & 0 \\ 0 & 0 & 0 & \mathbf{C_{q^k q^k}} \end{bmatrix}. \quad (17)
$$
$$
= \sum_{\delta \in \{r,x,y,z\}} \mathrm{E}\left\{\mathbf{q}_\delta^2\right\} \mathbf{I}
$$

Notwithstanding the above approach relies on the assumption that the input signal is $\mathbb{Q}$-proper, but it shows that the variance of a quaternion is obtained by the variance of its four components. Therefore, as an approximate of complete variance, we consider the average of the variance of each component as the quaternion variance and build the normalization as follows:

$$
\mathrm{QN}(\mathbf{x}) = \left(\frac{\mathbf{x} - \boldsymbol{\mu}_q}{\sqrt{\mathrm{var}\{\mathbf{x}\} + \epsilon}}\right)\gamma + \beta = \left(\frac{\mathbf{x} - \boldsymbol{\mu}_q}{\sqrt{\boldsymbol{\sigma}^2 + \epsilon}}\right)\gamma + \beta, \quad (18)
$$

where β is a shifting quaternion parameter, γ is a scalar parameter, and both of them are learnable parameters. $\boldsymbol{\mu}_q$ is the quaternion input mean, which is a quaternion itself, and $\boldsymbol{\sigma}^2$ is real-valued variance. The $\boldsymbol{\mu}_q$ and $\boldsymbol{\sigma}^2$ are defined as:

$$
\boldsymbol{\mu}_q = \frac{1}{C}\sum_{c=1}^{C}\mathbf{q}_{r,c} + \mathbf{q}_{x,c}\mathbf{i} + \mathbf{q}_{y,c}\mathbf{j} + \mathbf{q}_{z,c}\mathbf{k}, \quad (19)
$$
$$
= \bar{\mathbf{q}}_r + \bar{\mathbf{q}}_x\mathbf{i} + \bar{\mathbf{q}}_y\mathbf{j} + \bar{\mathbf{q}}_z\mathbf{k}
$$

$$
\boldsymbol{\sigma}^2 = \frac{1}{4C}\sum_{\delta \in \{r,x,y,z\}}\sum_{c=1}^{C}(\mathbf{q}_{\delta,c} - \bar{\mathbf{q}}_\delta) \otimes (\mathbf{q}_{\delta,c} - \bar{\mathbf{q}}_\delta)^*. \quad (20)
$$

To summarize, the quaternion convolutional block is defined as:

$$
\widehat{\boldsymbol{Q}}_p = \mathcal{F}_{\mathrm{p}}^{\mathrm{qcl}}\left(\boldsymbol{Q}_p\right) = \mathrm{ReLU}(\mathrm{QN}\left(\mathbf{W} \otimes \boldsymbol{Q}_p + \mathbf{b}\right)). \quad (21)
$$

Fig. 5. Visual illustration of the real-valued convolutional decoder. $\mathrm{up}_{[\times 2]}$ is bilinear interpolation by a factor of 2 and $\mathbf{F}_i^q$ is the low-level query feature extracted from *block i*.

4) Quaternion Aggregation: In the quaternion aggregation module (QAM), the output of each $\widehat{\boldsymbol{Q}}_p$ is upsampled and element-wise summed with the next level $\widehat{\boldsymbol{Q}}_{p+1}$ with one degree of finer resolution. A quaternion convolution layer then processes this merged quaternion feature to propagate semantic information to finer branches in a coarse-to-fine fashion. By applying QAM to quaternion features at different spatial scales, finer quaternion feature maps can be guided using the rich semantic information of deeper-level features, which dramatically boosts the performance. The QAM is defined as:

$$
\boldsymbol{Q}_2' = \mathcal{F}^{\mathrm{qcl}}\left(\widehat{\boldsymbol{Q}}_2 + \mathrm{up}_{[\times 2]}(\widehat{\boldsymbol{Q}}_3)\right)
$$
$$
\boldsymbol{Q}_1' = \mathcal{F}^{\mathrm{qcl}}\left(\widehat{\boldsymbol{Q}}_1 + \mathrm{up}_{[\times 2]}(\boldsymbol{Q}_2')\right) \quad (22)
$$

E. Episodic Readout Module

1) Transform Quaternion to Real: Note that most tasks, such as semantic segmentation, require outputs composed of real numbers. However, the output of the quaternion correlation learning module $\boldsymbol{Q}_1'$ consists of quaternions. Therefore, for FSS, we propose the quaternion2real module (Q2RM) to transform quaternion feature $\boldsymbol{Q}_1'$ Into ordinary features (i.e., $\mathbb{H}^{H_1^P \times W_1^P \times D} \rightarrow \mathbb{R}^{H_1^P \times W_1^P \times D}$), in which each element is a real number. Specifically, as illustrated in Figure 4, we split $\boldsymbol{Q}_1'$ into four components $\boldsymbol{Q}_{1,r}', \boldsymbol{Q}_{1,x}', \boldsymbol{Q}_{1,y}', \boldsymbol{Q}_{1,z}'$ and embed the global information by simply using global average pooling (GAP) to generate channel-wise statistics. Then, the real-valued feature map $\mathbf{F}_r$ is obtained through the soft-attention weight $\mathbf{W}_\delta^s$ on four components $\delta \in \{r, x, y, z\}$:

$$
\mathbf{F}_r = \sum_{\delta \in \{r,x,y,z\}} (\mathbf{W}_\delta^s \cdot \boldsymbol{Q}_{1,\delta}') \in \mathbb{R}^{H_q^1 \times W_q^1 \times D}. \quad (23)
$$

$\mathbf{W}_\delta^s$ is defined as:

$$
\mathbf{W}_\delta^s = \frac{e^{\mathcal{G}(\boldsymbol{Q}_{1,\delta}'))}}{\sum_{\delta \in \{r,x,y,z\}} e^{\mathcal{G}(\boldsymbol{Q}_{1,\delta}'))}}, \quad (24)
$$

where $\mathcal{G}(\cdot)$ is the global average pooling.

2) Real-valued Convolutional Decoder: In the work of [6], [7], [10], the features are bilinearly upsampled to the original image size, which may not successfully recover object segmentation details. Therefore, we propose a real-valued convolution decoder, as illustrated in Figure 5. The real-valued feature $\mathbf{F}_r$ is first concatenated with the corresponding low-level query features [39] from the backbone (e.g., *block1* and *block2* in ResNet-50 [40]) and then bilinear interpolation by

a factor of 2. We apply a 1×1 convolution on the low-level query features to reduce the number of channels since the low-level features usually contain a large number of channels (e.g., 256 or 512) which may outweigh the importance of $\mathbf{F}_r$. After the concatenation, the merged features are refined by utilizing 3×3 convolution and undergoes the above operations until the classification head which outputs the predicted mask $\mathbf{M}'_q \in [0,1]^{H \times W \times 2}$. The process is defined as follows:

$$\mathbf{M}'_q = \text{Decoder}\left(\mathbf{F}_r, \mathcal{P}_1(\mathbf{F}_1^q), \mathcal{P}_2(\mathbf{F}_2^q)\right), \quad (25)$$

where $\mathcal{P}_i(\cdot)$ linear projection. During testing, we take the maximum channel value at each pixel to obtain the predicted mask $\tilde{\mathbf{M}}_q \in \{0,1\}^{H \times W}$ for evaluation.

F. Attention Mechanism for K-shot Segmentation

In order to efficiently merge semantic information in the K-shot setting, we use a prior attention mechanism to dynamically fuse the predictions generated by different support images. Specifically, we compute the cosine similarity of the last layer of query and K support features to obtain K correlation tensors $\{\mathbf{c}_i(\mathbf{x}^q, \mathbf{x}^s)\}_{i=1}^K$, and then we take the maximum similarity among all support sub-dimension to generate the prior weight matrixes $\{\boldsymbol{w}_i\}_{i=1}^K \in \mathbb{R}^{H_q^p \times W_q^p}$:

$$\boldsymbol{w}_i = \max_{\mathbf{x}^s} \mathbf{c}_i(\mathbf{x}^q, \mathbf{x}^s). \quad (26)$$

Since the prior weight matrix is obtained by calculating the highest correspondence from support spatial sub-dimension, they provide pixel-level prior information about which support sample is more important. Therefore, we multiply the predictions of each shot branch with the weight matrix normalized by the softmax function, i.e., $\tilde{\mathbf{M}}^q = \sum_{i=1}^K \text{Softmax}(\boldsymbol{w}_i) \cdot \mathbf{M}'_{q,i}$. If the final prediction is above threshold τ, we assign foreground pixels to it, otherwise assignb ackground:

$$\hat{\mathbf{M}}^q_{(x,y)} = \begin{cases} 1 & \tilde{\mathbf{M}}^q_{(x,y)} > \tau \\ 0 & \text{otherwise} \end{cases}. \quad (27)$$

where (x,y) denotes the spatial location.

V. Experiments

A. Implementation Details

1) Datasets: The experiments are conducted on two standard benchmark datasets of FSS: PASCAL-5^i [41] and COCO-20^i [42]. The PASCAL-5^i dataset is obtained by combining PASCAL VOC 2012 with SBD [43], consisting of 20 object classes that are divided into 4 folds. The COCO-20^i [42] is a more challenging dataset, and it consists of 80 classes divided into 4 folds. Following the training/validation strategy of previous work [6], [7], [10], [22], [44] , we use three folds to build the training set, while the remaining fold is used to validate the model for cross-validation. During the evaluation, 1000 episodes (support-query pairs) from the test set are randomly selected to calculate the mean Intersection over Union (mIoU) and binary Intersection over Union (FBIoU) of all categories.

2) Experimental Setting: We use ResNet-50 [40] to conduct our main body experiments for a fair comparison with other methods. As in [6], [10], all backbone networks are initialized with ImageNet [49] pre-trained weights and are fixed during QCLNet training. The reason for doing so is to avoid them learning class-specific representations of the training data. Other layers are initialized by the default setting of PyTorch and the quaternion parameters are initialized following the proposal of [31]. It is worth mentioning that the number of quaternion feature maps is four times larger than real-valued, meaning 1 quaternion-valued feature map corresponds to 4 real-valued ones. To resolve the computational burden and preserve the representational ability of the model, the channel dimension D is fixed as 64 in all of the experiments. For ResNet, the features after *block2* are extracted to construct 3 pyramidal layers with different spatial scales. As such, by taking images with size 473×473 as input for ResNet-50, we can get the feature map with spatial size $[60 \times 60, 30 \times 30, 15 \times 15]$. In addition, we implement QCLNet using Pytorch [50] on Nvidia 3060 GPUs. QCLNet is trained in an episode-based meta-learning fashion using the Adam optimization algorithm [51] with a learning rate of $1e^{-3}$. The threshold τ in Eq. 27 for PASCAL-5^i and COCO-20^i is 0.5 and 0.6, respectively.

3) Evaluation Metric: We use the mean Intersection over Union (mIoU) and binary Intersection over Union (FB-IoU) as our evaluation metrics. For category c, IoU Is defined as $IoU_k = TP_k/(TP_k + FP_k + FN_k)$ where the TP_k, FP_k, FN_k are the number of true positives, false positives, and false negatives in segmentation masks. The mIoU metric average the IoUs of all test categories in a fold. The formulation follows $mIoU = \frac{1}{C} \sum_{k=1}^C IoU_i$ where C is the number of classes in each fold. And the FB-IoU calculates the mean of foreground and background IoUs regardless of the categories. As stated in [6], we mainly focus on mIoU since it considers the differences of all classes so that the performance bias of scarce classes can be alleviated.

B. Performance and Comparison

1) COCO-20^i: The COCO-20^i dataset is a very challenging dataset that contains many objects in a realistic scene image. We illustrate the mean-IoU in Table II, from which it can be seen that QCLNet achieves state-of-the-art results with a competitive parameter size (2.6M) under both 1-shot and 5-shot settings. For instance, under the 1-shot setting, with a VGG16 backbone, we outperform the HSNet and CAPL methods by 3.1% and 2.5%. Under the 1-shot setting, with a ResNet50 backbone, QCLNet outperform the HSNet methods by 2.4% and achieves the new state-of-the-art. In addition, we gain an impressive improvement of 3.1% and 2.4% in the 5-shot setting, which are significant margins for the challenging task. As such, the quaternion correlation learning in QCLNet indeed captures some inner benefits for boosting FSS performance, and we hope our model can shed light on future research in FSS.

Since most foreground classes only occupy a small spatial region of the whole image, the FB-IoU is biased toward the background class and causes it not convincing when evaluating

TABLE II

PERFORMANCE OF 1-SHOT AND 5-SHOT SEMANTIC SEGMENTATION ON THE COCO-20^i. THE BEST MEAN-IoUS ARE MARKED IN BOLD.

Method	Backbone	mean-IoU (1-shot)					mean-IoU (5-shot)					# learnable params
		Fold-0	Fold-1	Fold-2	Fold-3	Mean	Fold-0	Fold-1	Fold-2	Fold-3	Mean	
PPNet [22]	ResNet-50	28.1	30.8	29.5	27.7	29.0	39.0	40.8	37.1	37.3	38.5	31.5M
PMMs [8]	ResNet-50	29.3	34.8	27.1	27.3	29.6	33.0	40.6	30.1	33.3	34.3	-
RPMMs [8]	ResNet-50	29.5	36.8	28.9	27.0	30.6	33.8	42.0	33.0	33.3	35.5	-
PFENet [10]	ResNet-50	36.5	38.6	34.5	33.8	35.8	36.5	43.3	37.8	38.4	39.0	10.8M
ASGNet [7]	ResNet-50	-	-	-	-	34.6	-	-	-	-	42.5	10.4M
HSNet [12]	ResNet-50	36.3	43.1	38.7	38.7	39.2	43.3	51.3	48.2	45.0	46.9	**2.6M**
CAPL [23]	ResNet-50	-	-	-	-	39.8	-	-	-	-	48.3	-
Ours	ResNet-50	**39.8**	**45.7**	**42.5**	**41.2**	**42.3**	**46.4**	**53.0**	**52.1**	**48.6**	**50.0**	**2.6M**
FWB [45]	ResNet-101	17.0	18.0	21.0	28.9	21.2	19.1	21.5	23.9	30.1	23.7	43.0M
PFENet [10]	ResNet-101	34.3	33.0	32.3	30.1	32.4	38.5	38.6	38.2	34.3	37.4	10.8M
HFA [46]	ResNet-101	28.6	36.0	30.1	33.2	32.0	32.6	42.1	30.3	36.1	35.3	36.5M
SAGNN [47]	ResNet-101	36.1	41.0	38.2	33.5	37.2	40.9	48.3	42.6	38.9	42.7	-
HSNet [12]	ResNet-101	37.2	44.1	42.4	41.3	41.2	45.9	53.0	51.8	47.1	49.5	**2.6M**
CAPL [23]	ResNet-101	-	-	-	-	42.8	-	-	-	-	50.4	-
Ours	ResNet-101	**40.0**	**45.5**	**45.1**	**43.6**	**43.6**	**46.9**	**55.8**	**53.6**	**51.1**	**51.9**	**2.6M**

TABLE III

PERFORMANCE OF 1-SHOT AND 5-SHOT SEMANTIC SEGMENTATION ON THE PASCAL-5^i. THE BEST MEAN-IoUS ARE MARKED IN BOLD.

Method	Backbone	mean-IoU (1-shot)					mean-IoU (5-shot)					# learnable params
		Fold-0	Fold-1	Fold-2	Fold-3	Mean	Fold-0	Fold-1	Fold-2	Fold-3	Mean	
AMP [48]	VGG-16	41.9	50.2	46.7	34.7	43.4	41.8	55.5	50.3	39.9	46.9	15.8M
PANet [9]	VGG-16	42.3	58.0	51.1	41.2	48.1	51.8	64.6	59.8	46.5	55.7	14.7M
HSNet [12]	VGG-16	59.6	65.7	**59.6**	54.0	59.7	64.9	**69.0**	**64.1**	58.6	64.1	**2.6M**
Ours	VGG-16	**61.3**	**66.8**	58.4	**55.8**	**60.6**	**66.1**	68.5	63.2	**58.8**	**64.2**	**2.6M**
CANet [6]	ResNet-50	52.5	65.9	51.3	51.9	55.4	55.5	67.8	51.9	53.2	57.1	19.0M
PPNet [22]	ResNet-50	47.8	58.8	53.8	45.6	51.5	58.4	67.8	64.9	56.7	62.0	31.5M
PFENet [10]	ResNet-50	61.7	69.5	55.4	56.3	60.8	63.1	70.7	55.8	57.9	61.9	10.8M
ASGNet [7]	ResNet-50	58.8	67.9	56.8	53.7	59.3	63.7	70.6	64.2	57.4	63.9	10.4M
SAGNN [47]	ResNet-50	64.7	69.6	57.0	57.2	62.1	64.9	70.0	57.0	59.3	62.8	-
HSNet [12]	ResNet-50	64.3	**70.7**	60.3	60.5	64.0	70.3	73.2	**67.4**	**67.1**	**69.5**	**2.6M**
CAPL [23]	ResNet-50	-	-	-	-	62.2	-	-	-	-	67.1	-
Ours	ResNet-50	**65.2**	70.3	**60.8**	**61.0**	**64.3**	**70.6**	**73.5**	66.7	**67.1**	**69.5**	**2.6M**
FWB [45]	ResNet-101	51.3	64.5	56.7	52.2	56.2	54.8	67.4	62.2	55.3	59.9	43.0M
PFENet [10]	ResNet-101	60.5	69.4	54.4	55.9	60.1	62.8	70.4	54.9	57.6	61.4	10.8M
ASGNet [7]	ResNet-101	59.8	67.4	55.6	54.4	59.3	64.6	71.3	64.2	57.3	64.4	10.4M
HSNet [12]	ResNet-101	67.3	72.3	62.0	63.1	66.2	71.8	74.4	67.0	68.3	70.4	**2.6M**
CAPL [23]	ResNet-101	-	-	-	-	63.6	-	-	-	-	68.9	-
Ours	ResNet-101	**67.9**	**72.5**	**64.3**	**63.4**	**67.0**	**72.5**	**74.8**	**68.5**	**68.9**	**71.2**	**2.6M**

TABLE IV

COMPARISON OF FB-IoU PERFORMANCE OF 1-SHOT AND 5-SHOT SEGMENTATION ON THE COCO-20^i. △ MEANS INCREMENT OVER 1-SHOT SEGMENTATION RESULT.

Method	1-shot	5-shot	△
PANet [9]	59.2	63.5	4.3
PFENet [10]	58.6	61.9	3.3
DAN [44]	62.3	63.9	1.6
ASGNet [7]	60.4	67.0	**6.6**
SAGNN [47]	60.9	63.4	2.5
HSNet [12]	68.2	70.7	2.5
Ours	**69.9**	**73.5**	4.6

performance. However, we also make comparisons of our model with other advanced approaches to COCO-20^i and the numbers are competitive (see Table IV).

2) PASCAL-5^i: In Table III, we compare QCLNet with the state-of-the-art methods on PASCAL-5^i. QCLNet outperforms state-of-the-art methods under both 1-shot and 5-shot settings. Specifically, in the 1-shot settings, our method outperforms the state-of-the-art by 0.3% and 0.8% with ResNet-50 and ResNet-101 respectively. And QCLNet performs significantly better than other methods by 0.8% with the resnet101 backbone in

the 5-shot setting. It is worth mentioning that PFENet and SAGNN used the additional training of the model for k-shot setting while QCLNet uses a simple k-shot fusion strategy to fuse the 5-shot results.

3) Segmentation Examples: To better understand our proposed method, we show segmentation results during the meta-testing phase, as shown in Figure 7. It can be found in our method (4th row), the accurate and complete segmentation results of novel classes are apparently generated compared with the baseline method (3rd row), which verifies the effectiveness of quaternion-valued correlation learning. We further visualize the four components of quaternion features, Figure 6. Based on the interaction of quaternion-weight components and soft-attention Q2RM, QCLNet capture different key relations within the components of a quaternion.

C. Ablation Studies

In this section, we show an ablation analysis to inspect the effectiveness of our major contributions, justify the architectural choices we made and investigate whether QCL can effectively learn relations between support and query set while

Fig. 6. Visualization of the four components of the quaternion feature. (Best viewed in color)

TABLE V
ABLATION STUDY OF THE PROPOSED APPROACH ON PASCAL-5^i.
"CAM" DENOTES CORRELATION AGGREGATION MODULE WHILE
"QCLM" DENOTES QUATERNION CORRELATION LEARNING MODULE
WITH QUATERNION NORMALIZATION.

Components		mean-IoU	
		1-shot	5-shot
(I)	Baseline	57.4	63.1
(II)	+ CAM	61.7	67.8
(III)	+ QCLM	63.5	68.5
(IV)	+ Q2RM	64.0	68.8
(V)	+ Low-level feat	**64.3**	**69.5**

TABLE VI
PERFORMANCE UNDER DIFFERENT CORRELATION AGGREGATE
STRATEGIES. "TOP K" DENOTES AGGREGATES CORRELATION BY STORING
TOP K PROMISING MATCHING SCORES, "SEP 4D CONV" DENOTE
SEPARABLE 4D CONVOLUTION.

Different aggregators	mean-IoU	
	1-shot	5-shot
TopK	62.6	66.9
Sep 4D conv	**64.3**	**69.5**

suppressing the degrees of freedom of model's parameter. Throughout this section, all the experiments are conducted with ResNet-50 in the 1-shot and 5-shot settings on PASCAL-5^i. Each ablation experiment is conducted under the same experimental setting for a fair comparison. We define the baseline as a model that replaces CAM with simple max pooling followed by standard 2D convolution, skips QCLM, and disregards low-level query features. Then we evaluate the effectiveness of our methods by adding components progressively.

1) Correlation aggregation: CAM aims at aggregating sparse information in a high-dimensional correlation tensor before the correlation learning process. In Table V, by simply introducing CAM to the baseline method, we improved the performance from 57.4% and 63.1% to 61.7% and 67.8%, demonstrating the necessity of aggregating support sub-dimension information. And CAM not only eases the computational burden caused by the high-dimensional properties of the correlation tensors, but also helps construct quaternion representations to facilitate subsequent correlation learning.

2) Quaternion Correlation Learning: The CAM simply employs two real-valued 2D convolutions in two sub-dimensions, which lacks the ability to consider interactions of matching scores. Therefore, as shown in Table V, with CAM followed by QCLM, we further improve the segmentation performance by 1.8% and 0.7%. This result shows that by encapsulating support spatial dimension as a quaternion, our method preserves intra-subspace relations and then enhances the hidden interactions between support and query.

3) Effectiveness of Q2RM: In Table V, when using the Q2RM to transform quaternion features instead of simply averaging the quaternion components, the performance improvement is significant (0.5% and 0.3%), validating the importance of quaternion components statistics because they correspond to different support subspaces.

4) Effectiveness of Low-level Feature: We also demonstrate the effectiveness of the low-level feature. The results of Table V show that adding low-level features in the decoder can help to find accurate correspondences, which in turn yields better segmentation performance (0.3% and 0.7%). We consider this improvement is that the low-level feature contains fine object segmentation detail, which can resolve the ambiguities in the correlation map and expedite the learning process.

5) Ablation Study on CAM: As mentioned in Section IV-C, we propose to aggregate the sparse information by applying separable 4D convolution. Compared with simply storing Top K promising matching scores, the proposed module achieves a sizable gain (see Table VI) under 1-shot and 5-shot settings. We attribute this phenomenon to the different utilization of neighborhood information by the two methods. Specifically, one tends to use convolution to gradually aggregate sparse information in the the correlation tensor, while the other tends to keep only the top K matches into a sparse correlation tensor, which is challenging to get enough correlation statistics.

6) Ablation Study on Different 2D Kernels: We provide an ablation study on different 2D kernels to justify the use of our proposed quaternion-valued kernel for correlation learning. In specific, we replace the proposed quaternion-valued kernel with the group convolution kernel and the standard 2D convolution kernel, and leaving all the other components for a fair comparison. Table VII summarizes the results. The learnable params are the parameters of the entire model after replacing the kernel.

As shown in Table VII, while both the standard 2D kernel and our quaternion-valued kernel interact with different components in the channel dimension, the high dimensional space of hypercorrelation results in standard 2D convolution with four larger parameters (9.2M vs. 2.6M) and worse performance than ours. The performance gap indicates that our approach is more balanced mining the semantic relations of two subspaces and significantly improves model parameters. In addition, to demonstrate the effectiveness of the weight sharing strategy in

Fig. 7. Segmentation results of the proposed QCLNet and the baseline. From top to bottom: (a) support image, (b) support mask, (c) ground truth of query image, (d) predictions of baseline, (e) predictions of QCLNet.

TABLE VII
PERFORMANCE UNDER THREE DIFFERENT 2D CONVONLUTION KERNELS. "GCONV" DENOTES THE GROUP CONVOLUTION, "SCONV" DENOTES THE
STANDARD CONVOLUTION AND "QCONV" DENOTES THE QUATERNION-VALUED CONVOLUTION.

| Kernel type | mean-IoU (1-shot) | | | | | mean-IoU (5-shot) | | | | | # learnable |
	Fold-0	Fold-1	Fold-2	Fold-3	Mean	Fold-0	Fold-1	Fold-2	Fold-3	Mean	params
Gconv kernel	63.9	69.0	**60.9**	55.9	62.4	69.4	72.3	65.4	60.5	66.9	**2.6M**
Sconv kernel	64.4	**70.6**	60.3	**61.1**	64.1	69.9	73.1	65.6	**67.1**	68.9	9.2M
Qconv kernel	**65.2**	70.3	60.8	61.0	**64.3**	**70.6**	**73.5**	**66.7**	**67.1**	**69.5**	**2.6M**

TABLE VIII
PERFORMANCE UNDER DIFFERENT NORMALIZATION STRATEGIES IN QCL
MODULE. "BN" DENOTES BATCH NORMALIZATION, "GN" DENOTES
GROUP NORMALIZATION AND "QN" DENOTES THE QUATERNION
NORMALIZATION.

| Different normalization | mean-IoU | |
	1-shot	5-shot
BN	63.7	67.6
GN	64.2	69.0
QN	**64.3**	**69.5**

TABLE IX
COMPARISON OF DIFFERENT 5-SHOT SOLUTIONS. OUR ATTENTION
MECHANISM PERFORMS THE BEST AND BRINGS THE MOST INCREMENT
OVER 1-SHOT BASELINE.

Method	mean-IoU	Increment
1-shot baseline	64.3	0
Mask-avg	69.3	5.0
Attention (ours)	**69.5**	**5.2**

Hamilton product, we also added a grouped convolution kernel in Table 8 for comparison, which gives a separate weight for each components in the quaternion feature and has the same parameter size (2.6M vs. 2.6M) as the quaternion convolution. The group convolution is defined as:

$$\mathbf{W}_g \otimes \mathbf{x} = (\mathbf{W}_r * \mathbf{q}_r) + (\mathbf{W}_x * \mathbf{q}_x)\,\mathbf{i}$$
$$+ (\mathbf{W}_y * \mathbf{q}_y)\,\mathbf{j} + (\mathbf{W}_z * \mathbf{q}_z)\,\mathbf{k}. \quad (28)$$

In Table VII, compared to group convolution which sim-

ply performs independent convolution performing independent convolution of the four components of a quaternion, the QCNN using Hamilton product improves the segmentation performance by 1.9% and 2.6%. This indicates that the weight sharing strategy of Hamilton product significantly maintains the important structural information of the quaternion space and fully exploit the internal latent interrelationship among the four components of quaternion feature.

7) Ablation Study on QN: The normalization of quaternion features is an important component, which affects the stability

Fig. 8. Per-class performance gains on PASCAL-5^i dataset. Our proposed QCLNet achieves significant improements against the baseline.

TABLE X
MEAN AND STD. OF FIVE TEST RESULTS (CLASS MIOU) ON PASCAL-5^i.
EACH ROW SHOWS FIVE TEST RESULTS WITH THE VALUES OF MEAN AND
STANDARD DEVIATION (STD.).

Fold	Exp-1	Exp-2	Exp-3	Exp-4	Exp-5	Mean	Std.
0	65.2	65.4	64.2	65.5	64.5	65.0	0.579
1	70.3	71.2	69.6	69.9	70.8	70.4	0.652
2	60.8	60.5	60.9	61.6	60.3	60.8	0.497
3	61.0	61.0	61.8	60.6	60.1	60.9	0.624

of the complex-valued neural network training process. As shown in Table VIII, with the replacement of QN with BN and GN in the QCLM, we improve the segmentation performance by 1.9 % and 0.5% in the 5-shot setting. This shows that in the quaternion space , the variance of each component should be calculated jointly and kept consistent. It is for this reason that the QN can effectively stabilize the training process of quaternion neural networks.

8) Ablation Study on K-shot Fusion Schemes: In the k-shot setting, we compare our attention mechanism to existing k-shot fusion scheme. We report the results of QCLNet with different fusion solutions in Table IX. Our attention mechanism performs the best and brings the most increment over a 1-shot baseline. This indicates that the training-free prior information can be more effective in fusing information from different support examples, which can effectively indicate which support sample is more important in the FSS setting with large intra-class variance.

D. Extensions

1) Category-Wise Performance: In Figure 8, we compare the category-wise segmentation performance on PASCAL-5^i. Categories such as cow, tv/monitor, train, and sofa achieved the largest performance gains. These categories can be largely affected by object views and pose (i.e., the query and support images may be quite different). This result shows that our proposed QCLNet has the potential to exploit the latent relationship between query and support images.

2) Result Stability: To demonstrate the stability and robustness of our model, we conduct multiple experiments on our PASCAL-trained QCLNet with different support samples. As seen in Table X, the values of standard deviation are lower than 0.7, which shows that QCLNet is insensitive to support samples and exhibits strong stability.

VI. CONCLUSION

We propose a novel Quaternion-valued Correlation Learning Network (QCLNet) for FSS to alleviate the class bias problem and explore precise support-query dense relations by considering the support sub-dimension in the quaternion space. Our QCLNet is formulated as a hyper-complex valued network, which exploits the properties of quaternion algebra and captures internal latent relations between query and support set. It also shares the quaternion-weight components during the Hamilton product, which greatly suppresses the degrees of freedom of the model's parameters. Experiments have been conducted On the PASCAL-5^i and COCO-20^i datasets, and the results demonstrate that our method improves the performance of FSS, and is striking in contrast with the prior approaches. As a systematic yet theoretically based method for dense correlation learning, our QCLNet provides a novel perspective on the few-shot learning problem.

REFERENCES

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2015, pp. 234–241.

[2] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.

[3] S. Shao, L. Xing, R. Xu, W. Liu, Y.-J. Wang, and B.-D. Liu, "Mdfm: Multi-decision fusing model for few-shot learning," *IEEE Trans. Circuits Syst. Video Technol.*, 2021.

[4] W. Jiang, K. Huang, J. Geng, and X. Deng, "Multi-scale metric learning for few-shot learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 3, pp. 1091–1102, 2020.

[5] N. Dong and E. P. Xing, "Few-shot semantic segmentation with prototype learning," in *Proc. Brit. Mach. Vis. Conf.*, vol. 3, 2018.

[6] C. Zhang, G. Lin, F. Liu, R. Yao, and C. Shen, "Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5217–5226.

[7] G. Li, V. Jampani, L. Sevilla-Lara, D. Sun, J. Kim, and J. Kim, "Adaptive prototype learning and allocation for few-shot segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8334–8343.

[8] B. Yang, C. Liu, B. Li, J. Jiao, and Q. Ye, "Prototype mixture models for few-shot semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 763–778.

[9] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng, "Panet: Few-shot image semantic segmentation with prototype alignment," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9197–9206.

[10] Z. Tian, H. Zhao, M. Shu, Z. Yang, R. Li, and J. Jia, "Prior guided feature enrichment network for few-shot segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 01, pp. 1–1, 2020.

[11] J. Min and M. Cho, "Convolutional hough matching networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2940–2950.

[12] J. Min, D. Kang, and M. Cho, "Hypercorrelation squeeze for few-shot segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 6941–6952.

[13] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic, "Neighbourhood consensus networks," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[14] I. Rocco, R. Arandjelović, and J. Sivic, "Efficient neighbourhood consensus networks via submanifold sparse convolutions," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 605–621.

[15] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, 2017.

[16] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5693–5703.

[17] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1520–1528.

[18] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2017.

[19] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 764–773.

[20] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 593–602.

[21] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots, "One-shot learning for semantic segmentation," in *Proc. Brit. Mach. Vis. Conf.*, 2017.

[22] Y. Liu, X. Zhang, S. Zhang, and X. He, "Part-aware prototype network for few-shot semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 142–158.

[23] Z. Tian, X. Lai, L. Jiang, S. Liu, M. Shu, H. Zhao, and J. Jia, "Generalized few-shot semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11 563–11 572.

[24] J. Min, J. Lee, J. Ponce, and M. Cho, "Learning to compose hypercolumns for visual correspondence," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 346–363.

[25] S. Huang, Q. Wang, S. Zhang, S. Yan, and X. He, "Dynamic context correspondence network for semantic alignment," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2010–2019.

[26] S. Gai and X. Huang, "Reduced biquaternion convolutional neural network for color image processing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 3, pp. 1061–1075, 2021.

[27] S.-C. Pei and C.-M. Cheng, "Color image processing by using binary quaternion-moment-preserving thresholding technique," *IEEE Trans. Image Process.*, vol. 8, no. 5, pp. 614–628, 1999.

[28] C. Papaioannidis and I. Pitas, "3d object pose estimation using multi-objective quaternion learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2683–2693, 2019.

[29] A. Hirose and S. Yoshida, "Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 23, no. 4, pp. 541–551, 2012.

[30] C. J. Gaudet and A. S. Maida, "Deep quaternion networks," in *Int. Jt. Conf. Neural Networks*. IEEE, 2018, pp. 1–8.

[31] T. Parcollet, M. Morchid, and G. Linarès, "Quaternion convolutional neural networks for heterogeneous image processing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* IEEE, 2019, pp. 8514–8518.

[32] Y. Tay, A. Zhang, A. T. Luu, J. Rao, S. Zhang, S. Wang, J. Fu, and S. C. Hui, "Lightweight and efficient neural natural language processing with quaternion networks," in *Proc. Annu. Meet. Assoc. Comput Linguist.*, 2019, pp. 1494–1503.

[33] X. Zhu, Y. Xu, H. Xu, and C. Chen, "Quaternion convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 631–647.

[34] G. Yang and D. Ramanan, "Volumetric correspondence networks for optical flow," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 794–805, 2019.

[35] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2015, pp. 448–456.

[37] E. Grassucci, E. Cicero, and D. Comminiello, *Quaternion generative adversarial networks*. Springer, 2022, pp. 57–86.

[38] C. C. Took and D. P. Mandic, "Augmented second-order statistics of quaternion random signals," *Signal Process.*, vol. 91, no. 2, pp. 214–224, 2011.

[39] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 447–456.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[41] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[42] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 740–755.

[43] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 297–312.

[44] H. Wang, X. Zhang, Y. Hu, Y. Yang, X. Cao, and X. Zhen, "Few-shot semantic segmentation with democratic attention networks," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 730–746.

[45] K. Nguyen and S. Todorovic, "Feature weighting and boosting for few-shot segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 622–631.

[46] B. Liu, J. Jiao, and Q. Ye, "Harmonic feature activation for few-shot semantic segmentation," *IEEE Trans. Image Process.*, vol. 30, pp. 3142–3153, 2021.

[47] G.-S. Xie, J. Liu, H. Xiong, and L. Shao, "Scale-aware graph neural network for few-shot semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5475–5484.

[48] M. Siam, B. N. Oreshkin, and M. Jagersand, "Amp: Adaptive masked proxies for few-shot segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 5249–5258.

[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* IEEE, 2009, pp. 248–255.

[50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.