The Visual Computer (2024) 40:799–810
https://doi.org/10.1007/s00371-023-02817-x

ORIGINAL ARTICLE

Query semantic reconstruction for background in few-shot segmentation

Haoyan Guan[1] · Michael Spratling[1]

Accepted: 3 February 2023 / Published online: 16 March 2023

Abstract
Few-shot segmentation (FSS) aims to segment unseen classes using a few annotated samples. Typically, a prototype representing the foreground class is extracted from annotated support image(s) and is matched to features representing each pixel in the query image. However, models learnt in this way are insufficiently discriminatory, and often produce false positives: misclassifying background pixels as foreground. Some FSS methods try to address this issue by using the background in the support image(s) to help identify the background in the query image. However, the backgrounds of these images are often quite distinct, and hence, the support image background information is uninformative. This article proposes a method, QSR, that extracts the background from the query image itself, and as a result is better able to discriminate between foreground and background features in the query image. This is achieved by modifying the training process to associate prototypes with class labels including known classes from the training data and latent classes representing unknown background objects. This class information is then used to extract a background prototype from the query image. To successfully associate prototypes with class labels and extract a background prototype that is capable of predicting a mask for the background regions of the image, the machinery for extracting and using foreground prototypes is induced to become more discriminative between different classes. Experiments achieves state-of-the-art results for both 1-shot and 5-shot FSS on the PASCAL-5^i and COCO-20^i dataset. As QSR operates only during training, results are produced with no extra computational complexity during testing.

Keywords Few-shot learning · Semantic segmentation · Metric learning

1 Introduction

The ability to segment objects is a long-standing goal of computer vision, and recent methods have achieved extraordinary results [1–3]. These results depend on a large number of pixel-level annotations which are time-consuming and costly to produce. When facing the situation where few exemplars from a novel class are available, these methods overfit and perform poorly. To deal with this situation, few-shot segmentation (FSS) methods aim to predict a segmentation mask for a novel category using only a few images and their corresponding segmentation ground-truths.

Most current FSS algorithms [4–11] follow a similar sequence of steps. Features are extracted from support and query images by a shared convolutional neural network (CNN) which is pre-trained on ImageNet [4,12–14]. Then the support image ground-truth segmentation mask is used to identity the foreground information in the support features. Generally, the object class is represented by a single foreground prototype feature vector [9,11,13,15,16]. Finally, a decoder is used to calculate the similarity of the foreground prototype and every pixel in the query feature-set to predict the locations occupied by the foreground object in the query image. This standard approach ignores the importance of background features that can be mined for negative samples in order to reduce false positives, and hence, make the model more discriminative.

Some FSS methods [13,15,17] extract background information from support images by using the support masks to identify the support image background. RPMMs [13] uses the Expectation-Maximization (EM) algorithm to mine more background information in the support images. MLC [18]

✉ Haoyan Guan
haoyan.guan@kcl.ac.uk

Michael Spratling
michael.spratling@kcl.ac.uk

[1] Department of Informatics, King's College London, London WC2B 4BG, UK

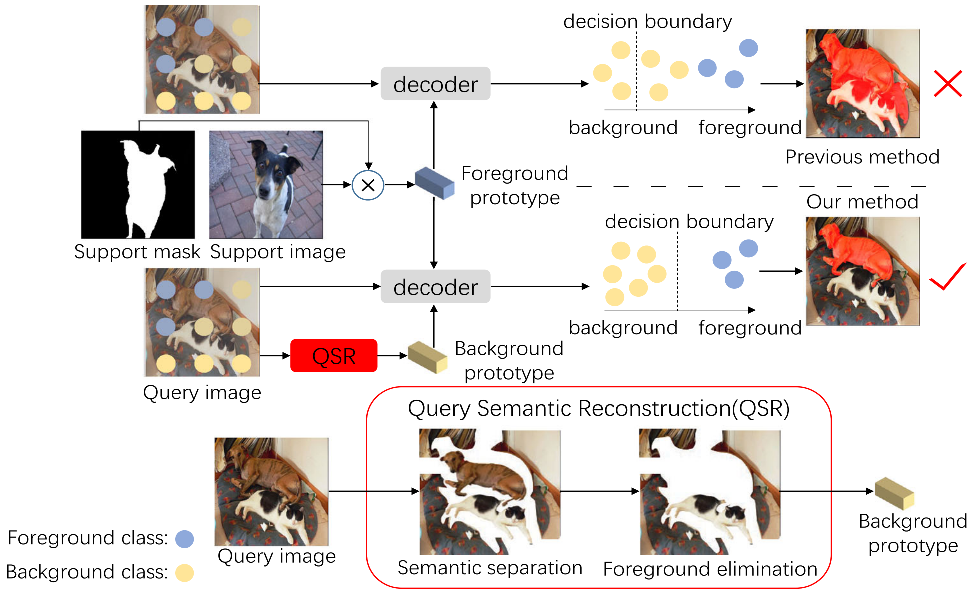

Fig. 1 Motivation for our method. Most previous FSS methods (as shown above the dashed line) use a decoder to classify features of the query image, by comparing them to a foreground prototype extracted from the support image and mask. This process often produces false positives: misclassifying the background (e.g. cat) as the foreground (e.g. dog). QSR (as shown below the dashed line) uses background information extracted from the query image at training time to learn a more descriminative decoder which is achieved by the semantic separation and foreground elimination

extracts a global background prototype by averaging together the backgrounds extracted from the whole training data in an offline process and then updates this global background prototype with the support background during training. However, the same category object may appear against different backgrounds in different images. The background information extracted from or aligned with the support image(s) is, therefore, unlikely to be useful for segmenting the query image. Existing FSS methods ignore the fact that the background information of an image is most relevant for segmenting that specific image.

In this paper, we are motivated by the issue illustrated in Fig. 1 and design a method that can extract background information from the query image itself to make existing FSS algorithms be more discriminative. Our method, Query Semantic Reconstruction (QSR), separates the feature extracted from a query image according to known classes and latent classes. Known classes are the categories that appear in the training data, like dog and cat in the example used in Fig. 1. Latent classes are unknown categories like mat and wall which are not explicitly labelled in the training data, but which can appear in the background in the training images. QSR learns to eliminate the foreground information according to the class labels. The remaining classes are used to define a prototype for the background of the query image that excludes contributions from the foreground class.

The extracted foreground and background prototypes are used as input to the prototype decoder module from the underlying, baseline, FSS method. The decoder produces predictions of foreground and background masks. The predictions are compared to a ground-truth mask and the loss is used to tune the parameters of the model. For these foreground and background prototypes to be effective at identifying the foreground and background regions of the query image, the whole model must be able to make the prototypes discriminative of features representing different semantics in the images. Hence, our method trains the underlying FSS method so that at test time it is able to more accurately segment images. Our method only predicts background masks during training to optimize the whole model. Hence, during testing the method is identical to that of the baseline.

The main contributions of our work are as follows:

1. To address the long-standing high false positive problem in FSS and to demonstrate that background information from the query image itself can be employed usefully for segmentation, we propose QSR that can be applied to many existing FSS algorithms to ensure they are better able to discriminate between foreground and background objects.
2. QSR improves existing FSS methods through optimized training. During testing our method is identical to the baseline, so no additional parameters or extra computation is needed at test-time.
3. We demonstrate the effectiveness of QSR using three different baselines methods: CaNet [4], ASGNet [9] and PFENet [16]. For the PASCAL-5^i dataset, QSR improves mIOU results of 1-shot and 5-shot FSS by 1.0% and 1.5% for CaNet, 1.8% and 2.1% for ASGNet, and by 1.9% and 4.8% for PFENet. For the COCO-20^i dataset, QSR improves ASGNet by 2.8% and 1.6%, PFENet by 4.5% and 3.8%.
4. Our method achieves new state-of-the-art performance on PASCAL-5^i, with mIOU of 62.7% in 1-shot, and 66.7% in 5-shot. On the COCO-20^i dataset, our method

achieves strong results of 36.9% in 1-shot, and 41.2% in 5-shot.

2 Related work

2.1 Semantic segmentation

Semantic segmentation requires the prediction of per-pixel class labels. The introduction of end-to-end trained fully convolutional networks [3] has provided the foundation for recent success on this task. Additional innovations to improve segmentation accuracy further have included a multi-scale cascade model named U-Net [19], dilated convolution [20] and pyramid pooling [21]. In contrast to these methods, we explore semantic segmentation in the few-shot scenario.

2.2 Few-shot learning

Few-shot learning (FSL) explores methods to enable models to quickly adapt to perform classification of new data. FSL methods can be categorized into generation, optimization or metric learning approaches. Generation methods [22–25] generate samples or features to augment the novel class data. Optimization approaches [26,27] learn commonalities among different tasks, and then a novel task can be fine-tuned on a few annotated samples based on the commonalities. Metric learning methods [28,29] learn to produce a feature space that allows samples to be classified by comparing the distance between their features. Most FSL methods focus on image classification and cannot be easily adapted to produce the per-pixel labels required for segmentation.

2.3 Few-shot segmentation learning

The first FSS method [30] employed a two-branch comparison framework that has become the basis for FSS methods. PaNet [15] used prototype feature-vectors to represent support object classes and then compared their similarity with query features to make predictions. Other methods have improved different aspects of this process, for example, by extracting multiple prototypes representing different semantic classes [9,13], by iteratively refining the predictions [4], or using a training-free prior mask generation method [16]. Some methods extract information not only from support images, mining latent classes from the training dataset to search for more prototypes [18], or supplementing prototypes with support predictions [11].

3 Problem setting

Formally, we define a base dataset $\mathcal{D}_{base}$ with known classes $\mathcal{C}_{known}$. The FSS task is to use $\mathcal{D}_{base}$ to train a model which is able to segment new classes $\mathcal{C}_{novel}$, for which only a few annotated examples are available. The key point of FSS is that $\mathcal{C}_{novel} \notin \mathcal{C}_{known}$. Specifically, $\mathcal{D}_{base}$ is a large set of image-mask pairs $(I^j, M^j)_{j=1}^{Num}$, where M^j is the semantic segmentation mask for the training image I^j, and Num is the number of image-mask pairs. During testing, the model has access to a support set $S = (I_s^i, M_s^i)_{i=1}^{k} \in \mathcal{C}_{novel}$, where M_s^i is the semantic segmentation mask for support image I_s^i, and k is the number of image-mask pairs, which is small (typically either 1 or 5 for 1-shot and 5-shot tasks respectively). A query (or test) set $Q = (I_q, M_q) \in \mathcal{C}_{novel}$ is used to evaluate the performance of the model, where M_q is the ground-truth mask for image I_q. The model uses the support set S to predict a segmentation mask, $\hat{M}_f$, for each image I_q in query set Q.

4 Method

4.1 Overview

Figure 2 illustrates our method for 1-shot segmentation. Both support and query images are input into a shared CNN. In common with our baselines, CaNet [4], ASGNet [9] and PFENet [16], we use a ResNet [1] pre-trained on ImageNet [12] for this encoder backbone and choose features generated by $block2$ and $block3$. All parameter values in

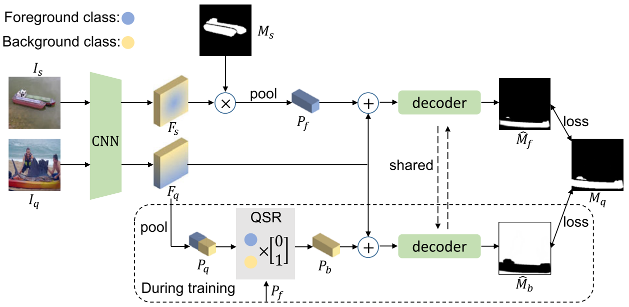

Fig. 2 An overview of our method for 1-shot segmentation. Like other FSS methods, our method extracts a foreground prototype from the support image and uses this to predict a foreground segmentation mask for the query image. QSR (dashed box) operates at training time to learn to represent different semantic categories in the query image, and uses this class information to define a background prototype. The background prototype is then used to predict a segmentation mask for the background regions of the query image via the same decoder as is used for the foreground prediction. To improve the accuracy of this additional prediction, the decoder is induced to become more discriminate. This ability to discriminate between foreground and background objects results in improved performance at test time, when the process illustrated in the dashed region is not used

*block*2, *block*3, and earlier layers are fixed. These features are concatenated and encoded using a convolution layer. The convolution layer parameters are optimized by the loss function (details in Sect. 4.3). For CaNet [4] and ASGNet [9], this layer has a 3 × 3 convolution kernel shared between support and query branches. For PFENet [16], two independent 1 × 1 convolution layers are defined for support and query features, respectively. After the convolution layer, the CNN produces support features F_s and query features F_q of size $d \times h \times w$, where d is the number of channels, and h, w are the height and width.

As for the baseline methods [4,9,16], masked average pooling (MAP) was used to extract the foreground prototype P_f:

$$P_f = \frac{\sum_{i=1}^{hw} F_s(i) \cdot \mathbb{1}[M_s(i) = 1]}{\sum_{i=1}^{hw} \mathbb{1}[M_s(i) = 1]} \tag{1}$$

where i indexes the spatial locations of features, and $\mathbb{1}[\cdot]$ is the indicator function, which equals 1 if the argument is True and 0 otherwise.

Global average pooling (GAP) was used to extract a query prototype P_q from the query features F_q:

$$P_q = \mathrm{GAP}(F_q) \tag{2}$$

Both the foreground and query prototypes were input to our QSR method (defined in Sect. 4.2). QSR maps different regions of the query image to semantic classes, and uses this class information to generate a background prototype P_b:

$$P_b = \mathrm{QSR}(P_q, P_f) \tag{3}$$

In Sect. 4.3, we describe how we utilise the prototype decoder module from the baseline FSS method. These modules are used to predict final semantic segmentation masks. The foreground prototype P_f is used to make a foreground prediction $\hat{M}_f$ and the background prototype P_b is used for a background prediction $\hat{M}_b$. The prototype decoder modules for foreground and background prediction are identical and share parameters. Our method only predicts a background mask during training. During testing, the method is identical to the baseline and only uses the foreground prototype to predict the foreground mask.

In this paper, we limited ourselves to being consistent with the baselines: using a frozen backbone CNN and masked average pooling to extract a single foreground prototype. In addition, we also extract only one background prototype making is possible to share parameters in the decoder module that is applied to both the foreground and background prototype. Future work might usefully explore improved methods of representing foreground objects, for example, by using multiple prototypes.

4.2 Query semantic reconstruction

Our method assumes that images contain objects from known classes and latent classes. Known classes are ones corresponding to the labels provided in the training data and we define them as $\mathbb{C}^k = \{C_0^k, C_1^k, \ldots, C_{N_k}^k\}$. The number of known classes, N_k, is defined by the training dataset, for example $N_k = 15$ in PASCAL-5^i [31]. During training, the foreground class C_f is contained in $\mathbb{C}^k$. Latent classes are given the generic label of ‘background’ in the training data. However, we define multiple latent classes to represent possible background objects and they are defined as $\mathbb{C}^l = \{C_0^l, C_1^l, \ldots, C_{N_l}^l\}$. The number of latent classes, N_l, is a hyper-parameter and the effects of different values were explored in experiments, the results of which are reported in Table 6. The background class must be a member of the set of latent classes or the set of known classes, excluding the class of the foreground object, which can be expressed as:

$$C_b \in \mathbb{C}^l \cup \mathbb{C}^k \backslash C_f \tag{4}$$

Mapping between prototype feature-vectors and classes is achieved using a layer of weights. A known class weight matrix W_k whose size is $N_k \times d$ maps from the $1 \times d$ prototype to the N_k known class labels. Hence, each row vector in W_k represents the corresponding category in $\mathbb{C}^k = \{C_0^k, C_1^k, \ldots, C_{N_k}^k\}$. In the same way, a latent classes weight matrix W_l, with size $N_l \times d$, maps from a prototype to the latent categories in $\mathbb{C}^l = \{C_0^l, C_1^l, \ldots, C_{N_l}^l\}$. W_k and W_l are both randomly initialized.

The known class weights can be learnt directly from the training data. In each episode, (P_f, C_f) is calculated from (F_s, M_s), where $C_f \in \mathbb{C}^k$. $P_f \times W_k$ is used as the prediction for the category of the foreground object. Cross-Entropy (CE) loss can then be used to update the known class weights to provide better representations of object class labels:

$$\mathcal{L}_{known} = \mathrm{CE}(C_f, P_f \times W_k) \tag{5}$$

The true latent class labels are unknown, so learning the latent classes weights assumes that all categories (both known and latent) should be independent of each other. A possible method to achieve this is the application of contrastive loss [32,33] to constrain each class representation to be independent by maximizing the orthogonality of their representations. A previous FSS method, ASR [8], has used contrastive loss to generate orthogonal semantic prototypes for foreground classes. In this paper, we apply the technique used in [32], a more efficient method, to constrain all class weights to be independent. Specifically, we define W_c as the concatenation of W_k and W_l, (i.e. W_c has size $(N_k + N_l) \times d$),

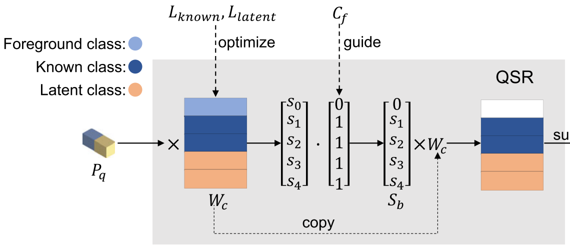

Fig. 3 Query semantic reconstruction (QSR). A query prototype P_q is multiplied with the semantic class weights W_c (which are optimized by $\mathcal{L}_{known}$ and $\mathcal{L}_{latent}$) to generates score values measuring the correlation between P_q and each class. The score for the current foreground class C_f is set to zero. The score S_b is multiplied with W_c to reconstruct a background prototype P_b eliminating any contribution from the foreground class. Note that the foreground class is one of the known classes, but is shown using a different colour for clarity

we first calculate the cross-correlation matrix, W, as:

$$W = W_c \times W_c^T \tag{6}$$

The loss function for learning the latent class weights is defined as:

$$\mathcal{L}_{latent} = \sum_{i}^{N_k+N_l} (1 - W_{ii})^2 + \sum_{i}^{N_k+N_l} \sum_{j \neq i}^{N_k+N_l} W_{ij}^2 \tag{7}$$

where i, j index the spatial location of the cross-correlation matrix. The latent loss tries to make the cross-correlation matrix close to the identity matrix. This causes each category to be statistically independent of all others.

As illustrated in Fig. 3, a background score, S_b, is calculated to measure the correlation between each non-foreground class and the query image prototype:

$$S_b = (P_q \times W_c) \cdot \mathbb{1}[\mathbb{C}^l \cup \mathbb{C}^k \backslash C_f] \tag{8}$$

where P_q is the query prototype from Eq. (2). Finally the background prototype is calculated, by back-projecting the scores (which represent the classes predicted to be present in the background) through the weights that represent the classes:

$$P_b = \sum_{i=1}^{N_k+N_l} (W_c(i,:) \times S_b(i)) \tag{9}$$

where the colon means the whole dimension. This generates a prototype that represents a mixture of feature-vectors representing the classes believed to be present in the background of the query image.

In order to be able to share the same decoder with the baseline, d is set to 256. However, such a large value may cause the background prototypes to be redundant. On PASCAL-5^i, the ratio between the class number $(N_k + N_l)$ and d in W_c is 30:256, compared to about 8:1 in [32]. Although these two ratios are used in unrelated tasks, and we also have the known loss to constrain the W_k part of W_c, in future work it would be worth-while setting d as a hyper-parameter that can be tuned for different datasets.

4.3 Prototypes decoder module

We use CaNet [4], ASGNet [9] and PFENet [16] as baselines on which to test our method. These methods have been widely used as the underlying model enhanced by various previous techniques [10,11,13]. Unlike most previous methods that modify the structure of the baseline decoder network, we try to improve it through better training. Each baseline incorporates a prototype decoder module (called the Iterative Optimization Module in CaNet, FPN in ASGNet and the Feature Enrichment Module in PFENet) that takes as input the foreground prototype and query features, and outputs a predicted segmentation mask $\hat{M}_f$. In addition to using this module in the standard way, we also use it with the foreground prototype replaced by the background prototype, so that it outputs a background prediction $\hat{M}_b$. When predicting the background mask in the ASGNet baseline, we use only one background prototype ignoring its ability to use multiple prototypes. PFENet also uses a prior mask (H) to supplement $\hat{M}_f$ and this input is replaced by (1 - H) to predict $\hat{M}_b$ when using PFENet as the baseline.

Based on the two predicted segmentation masks, we define two loss functions which are consistent with those used by the baselines:

$$\mathcal{L}_{base}(f) = \text{CE}(\hat{M}_f, M_q) \tag{10}$$

$$\mathcal{L}_{base}(b) = \text{CE}(\hat{M}_b, 1 - M_q) \tag{11}$$

The overall loss combines the losses defined in Eqs. 5, 7, 10 and 11, as follows:

$$\mathcal{L} = \mathcal{L}_{base}(f) + \alpha \mathcal{L}_{base}(b) + \beta(\mathcal{L}_{known} + \mathcal{L}_{latent}) \tag{12}$$

where α and β are parameters to balance the losses. Results for experiments investigating the effects of these hyper-parameters are reported in Table 7. When $\alpha = \beta = 0$, $\mathcal{L} = \mathcal{L}_{base}(f)$ and the whole method degenerates to the baseline.

For multi-shot tasks (i.e. when applied to k-shot FSS when $k > 1$), we use the same method as the corresponding baseline. Specifically, CaNet [4] designs an attention mechanism to fuse different features generated by each of the k support images. ASGNet [9] uses super-pixels to generate multiple prototypes of support images. PFENet [16] averages the foreground prototypes from k support images together. As QSR obtains the background prototype from the query image, QSR is unaffected by the number of support images which makes QSR easy to integrate with different baseline methods.

5 Experiments

5.1 Experimental setup

5.1.1 Datasets

We evaluate our method on two benchmark datasets, PASCAL-5^i [30] and COCO-20^i [35]. PASCAL-5^i includes the PASCAL VOC2012 [31] and the extended SDS datasets [36]. It contains 20 classes which are divided into 4 folds each containing 5 classes. COCO-20^i is the MS-COCO dataset [37] with the 80 classes divided into 4 folds each containing 20 classes. Following previous standard practice [4,16], we use fourfold cross-validation to measure performance on both datasets: testing each fold in turn using a model that had been trained on the other three folds. A random sample of 1,000 query-support pairs is used to test each fold in PASCAL-5^i and 20,000 in COCO-20^i.

5.1.2 Implementation details

As mentioned above, we use CaNet [4], ASGNet [9] and PFENet [16] as baselines. The whole model is trained end-to-end. As QSR is only used in the training phase, the model is identical to the baseline during testing. The details specific to QSR were as follows: the class weights W_c (Sect. 4.2) were initialized from the uniform distribution $(-\sqrt{1/d}, \sqrt{1/d})$. The loss weights α & β (Eq. (12)) were set to 1.0 & 0.5 in PASCAL-5^i and 1.0 & 0.1 in COCO-20^i. The motivation for reducing β for COCO-20^i was because this dataset has more categories. The number of latent classes N_l (Sect. 4.2) was set to 15 in PASCAL-5^i and 60 in COCO-20^i to make $N_l = N_k$ for each dataset. Consistent with the baselines, d was set to 256. In common with many previous FSS methods [5–11], and the baselines, feature extraction was performed using a ResNet [1] pre-trained on ImageNet [12]. During training, we used the methods and hyper-parameters used by the baselines. Specifically, for CaNet [4], weights were optimised using SGD with momentum of 0.9 and a weight decay of 0.0005. Training was performed for 200 epochs with a learning rate of 0.00025 and a batch size of 4. For ASGNet [9], the model was trained with the SGD optimizer and an initial learning rate to 0.0025 with batch size 4 on Pascal-5^i, and 0.005 with batch size 8 on COCO-20^i. For PFENet [16], SGD was also used as the optimizer. The momemtum was set to 0.9 and the weight decay to 0.0001. On PASCAL-5^i, 200 epochs were used with a learning rate of 0.0025 and a batch size of 4. On COCO-20^i, the PFENet baseline was trained for 50 epochs with a learning rate of 0.005 and a batch size 8. On both datasets, the learning rate was reduced following the “poly” policy [38].

5.1.3 Evaluation metrics

Following standard practice, we use mean intersection over union (mIoU) as the primary evaluation metric. It computes the IoU for each individual foreground class and then calculates an average of these values over all classes (5 in PASCAL-5^i and 20 in COCO-20^i). We also report the results of FB-IoU, which calculates the mean IoU for the foreground (i.e. for all objects ignoring class labels) and the background. We use false positive rate (FPR) which is defined as $\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP+TN}}$, where FP is the number of background pixels incorrectly labelled as foreground, and TN is the number of background pixels correctly labelled as background.

5.2 Comparison with the state-of-the-art

Tables 1 and 2 compare our method with other approaches on PASCAL-5^i. When QSR is applied to PFENet, the method outperforms the previous state-of-the-art in both the 1-shot and 5-shot settings. For each baseline, the QSR method improves performance on every fold, and overall, for both 1-shot and 5-shot segmentation tasks. This is achieved with only a small increase in the number of learnable parameters, as indicated in the last column of Table 2. These additional parameters are due to matrix W_c (see Sect. 4.2), and are only used during training: at test time the proposed method uses an identical number of parameters as the corresponding baseline. The ability to improve performance for three existing FSS methods suggests that QSR may have the potential to provide a general-purpose method of improving the accuracy of FSS approaches. Additional results using a different backbone architecture are shown in Table 3. These results show that increasing the size of the backbone does not, in this case, improve performance, but that QSR continues to improve performance in comparison with the baseline.

Table 4 compares our method with other approaches on COCO-20^i. QSR is able to increase performance when used in conjunction with both baselines, and for the ASGNet baseline increase performance a level that is state-of-the-art. This is achieved with only a small increase in the number of learnable parameters used during training. The number of additional parameters is 15.36k. The reason for the larger increase in parameters here compared to that for PASCAL-5^i is due to matrix W_c being larger due to an increase in the number of classes. More detailed results for the proposed, showing performance on individual folds and with different backbones, are shown in Table 5. These results show that QSR is consistent in improving performance across folds.

Table 1 mIoU (%) results for 1-shot and 5-shot FSS on PASCAL-5^i. 'Mean' is the mIoU averaged across folds. The best result for each column is in bold. Methods of first two rows use VGG16 [34] for feature extraction, while all others use ResNet50 [1]

	1-shot					5-shot				
Method	P-5^0	P-5^1	P-5^2	P-5^3	Mean	P-5^0	P-5^1	P-5^2	P-5^3	Mean
OSLSM [30]	33.6	55.3	40.9	33.5	40.8	35.9	58.1	42.7	39.1	43.9
PANet [15]	42.3	58.0	51.1	41.2	48.1	51.8	64.6	59.8	46.5	55.7
RPMMs [13]	55.2	66.9	52.6	50.7	56.3	56.3	67.3	54.5	51.0	57.3
CWT [7]	56.3	62.0	59.9	47.2	56.4	61.3	68.5	68.5	56.6	63.7
ASR [8]	53.8	69.6	51.6	52.8	56.9	56.2	70.6	53.9	53.4	58.5
RePRI [17]	59.8	68.3	62.1	48.5	59.7	64.6	71.4	71.1	59.3	66.6
MMNet [10]	58.0	70.0	58.0	55.0	60.2	60.0	70.6	56.3	60.3	61.8
SCL [11]	63.0	70.0	56.5	57.7	61.8	64.5	70.9	57.3	58.7	62.9
MLC [18]	59.2	**71.2**	**65.6**	52.5	62.1	63.5	71.6	**71.2**	58.1	66.1
CANet [4]	52.5	65.9	51.3	51.9	55.4	55.5	67.8	51.9	53.2	57.1
CANet+QSR (ours)	56.1	66.3	51.5	52.3	56.4	59.3	68.7	52.8	53.6	58.6
ASGNet [9]	58.8	67.9	56.8	53.7	59.3	63.7	70.6	64.2	57.4	63.9
ASGNet+QSR (ours)	62.0	68.4	57.8	56.1	61.1	66.5	71.2	65.1	61.0	66.0
PFENet [16]	61.7	69.5	55.4	56.3	60.8	63.1	70.7	55.8	57.9	61.9
PFENet+QSR (ours)	**63.1**	69.9	58.7	**58.9**	**62.7**	**68.3**	**71.7**	63.1	**63.6**	**66.7**

5.3 Ablation study

The following ablation studies were conducted with the PFENet baseline using the 1-shot setting on PASCAL-5^i.

5.3.1 Numbers of latent classes

Table 6 compares the performance achieved when using different numbers of latent classes, N_l. When $N_l = 0$ there are no latent classes, only known classes, and $W_c = W_k$ (see Sect. 4.2). It can be seen that the best results were produced when $N_l = 15$, which is equal to the number of categories in the training data (15 in PASCAL-5^i). As the number N_l increased, the results become poorer. However, for every value of N_l tested, the performance of the proposed method improves on the results produced by the the baseline model (60.8%, see Table 1).

5.3.2 Effects of loss weight

Table 7 shows the impact of different loss weights, α and β (see Eq. (12)) on the results. When $\alpha = \beta = 0$, the loss function becomes equivalent to the baseline loss $\mathcal{L}_{base}(f)$, the results produced are therefore identical to those of the baseline model. All combinations of non-zero values for α and β produced mIoU results that were better than those of the baseline. For the loss weights tested, the best results were produced with $\alpha = 1$, meaning that the background and foreground information was weighted equally, and $\beta = 0.5$.

Table 2 FB-IoU (%) results of 1-shot and 5-shot FSS on PASCAL-5^i. 'Params' is the number of learnable parameters (values preceded by a plus show the number QSR added during training). − denotes results that were not provided in the original paper. For methods listed in Table 1 but not here, no relevant data was provided in the published work

Method	1-shot	5-shot	Params
OSLSM [30]	61.3	61.5	276.7 M
PANet [15]	66.5	70.7	14.7 M
RPMMs [13]	−	−	19.6 M
SCL [11]	71.9	72.8	−
MLC [18]	−	−	**8.7 M**
CANet [4]	66.2	69.6	19.0 M
CANet+QSR	69.1	70.8	+7.68K
ASGNet [9]	69.2	74.2	10.4 M
ASGNet+QSR	71.3	75.0	+7.68K
PFENet [16]	73.3	73.9	10.8 M
PFENet+QSR	**75.4**	**77.2**	+7.68K

The best result for each column is in bold

5.3.3 Background prototype from support images

Table 8 explores the effects of extracting background information from different images. In the baseline, background information was not used, and the results are the same as the underlying FSS method. For the results labelled 'Support', the background information was extracted from the support image, rather than the query image. This was achieved by replacing the query features F_q in Eq. (3) with the support features F_s, but keeping other settings unchanged to allow for a fair comparison. It can be seen that this method produces

Table 3 mIoU (%) results for 1-shot and 5-shot FSS on PASCAL-5^i. These results were produced using a different feature extraction backbone than was used for the corresponding results in Table 1

	1-shot					5-shot					
Method	P-5^0	P-5^1	P-5^2	P-5^3	Mean	P-5^0	P-5^1	P-5^2	P-5^3	Mean	Backbone
PFENet	60.5	69.4	54.4	55.9	60.1	62.8	70.4	54.9	57.6	61.4	ResNet101
PFENet+QSR (ours)	**60.6**	**70.1**	**57.1**	**59.1**	**61.7**	**64.2**	**72.1**	**61.3**	**62.2**	**65.0**	ResNet101

The best result for each column is in bold

Table 4 Mean mIoU (%) results for 1-shot and 5-shot FSS on COCO-20^i. Results for SCL and PFENet (including our variant) were obtained using a ResNet101 [1] for feature extraction, and other results were obtained using a ResNet50

Method	1-shot	5-shot
RPMMs [13]	30.6	35.5
CWT [7]	32.9	41.3
ASR [8]	32.6	34.4
RePRI [17]	34.1	41.6
MMNet [10]	37.2	38.0
SCL [11]	37.0	39.9
MLC [18]	33.9	40.6
ASGNet [9]	34.6	42.5
ASGNet+QSR	**37.4**	**44.1**
PFENet [16]	32.4	37.4
PFENet+QSR	36.9	41.2

The best result for each column is in bold

little improvement over the baseline. For the results labelled 'Query', the background information was extracted from the query image. This is our proposed QSR method of extracting background prototypes, which produces a more significant improvement in the results. Hence, extracting background information from the query image is more effective than extracting it from the support image. We believe that this is due to there being a diverse range of backgrounds against which objects from the same category can appear in different images. Extracting foreground and background information from different training images enables the decoder to be trained to correctly distinguish foreground objects from a larger variety of backgrounds.

5.3.4 Importance of prototype reconstruction

Table 9 shows the effects of using different methods to extract the background prototypes. The results labelled 'Mask' used the query image segmentation masks (which are available during training) to obtain the background prototypes directly. Specifically, masked average pooling (Eq. (1)) was used to generate background prototypes replacing those generated by QSR in Eq. (3). The final loss function in Eq. (12) becomes $\mathcal{L} = \mathcal{L}_{base}(f) + \mathcal{L}_{base}(b)$. As Table 9 shows, this method improves the results compared to the baseline, which reinforces the idea that using background information can improve the training of the model. However, QSR provides a further improvement in the results, suggesting that the background prototypes created through the proposed method are more informative for training the model, presumably as the background prototypes are more representative of the entire training dataset, rather than just the current query image.

5.4 Model analysis

The following experiments to analyse QSR were performed with the PFENet baseline using the 1-shot setting in PASCAL-5^i.

Table 5 mIoU (%) results for 1-shot and 5-shot FSS in COCO-20^i. This table shows more detailed results, with performance on each fold, compared to Table 4. In addition, it also shows additional results for our proposed method when using a ResNet101 as the feature extraction backbone. This allows a more direct comparison with the published results for PFENet using a ResNet101

	1-shot					5-shot					
Method	C-5^0	C-5^1	C-5^2	C-5^3	Mean	C-5^0	C-5^1	C-5^2	C-5^3	Mean	Backbone
ASGNet	–	–	–	–	34.6	–	–	–	–	42.5	ResNet50
ASGNet+QSR (ours)	**34.8**	39.8	**40.7**	**34.3**	**37.4**	**40.2**	**47.5**	**48.0**	**40.7**	**44.1**	ResNet50
PFENet	34.3	33.0	32.3	30.1	32.4	38.5	38.6	38.2	34.3	37.4	ResNet101
PFENet+QSR (ours)	34.1	38.4	35.5	32.3	35.1	36.9	40.1	38.0	37.7	38.2	ResNet50
PFENet+QSR (ours)	33.6	**41.0**	39.2	33.8	36.9	36.3	44.9	44.3	39.4	41.2	ResNet101

The best result for each column is in bold

Table 6 Effects of different numbers of latent classes, N_l

N_l	P-5^0	P-5^1	P-5^2	P-5^3	Mean
0	62.2	69.3	57.7	58.4	61.9
15	63.1	**69.9**	**58.7**	**58.9**	**62.7**
30	**63.6**	69.5	57.7	58.6	62.4
45	61.8	69.5	57.2	57.5	61.5
60	61.6	69.1	58.0	57.7	61.6

The best result for each column is in bold

Table 7 Effects of different loss weights, α and β

α	β	P-5^0	P-5^1	P-5^2	P-5^3	Mean
0.0	0.0	61.7	69.5	55.4	56.3	60.8
0.5	0.5	62.1	69.6	55.0	58.8	61.4
0.5	1.0	62.3	69.8	55.0	58.3	61.4
1.0	0.5	**63.1**	**69.9**	**58.7**	**58.9**	**62.7**
1.0	1.0	61.8	66.3	58.1	58.5	61.2

The best result for each column is in bold

Table 8 Effects of different sources for background prototypes

Method	P-5^0	P-5^1	P-5^2	P-5^3	Mean
Baseline	61.7	69.5	55.4	56.3	60.8
Support	62.1	69.8	54.5	57.1	60.9
Query	**63.1**	**69.9**	**58.7**	**58.9**	**62.7**

The best result for each column is in bold

Table 9 Effects of methods to reconstruct background prototypes

Method	P-5^0	P-5^1	P-5^2	P-5^3	Mean
Baseline	61.7	69.5	55.4	56.3	60.8
Mask	**63.4**	69.5	54.9	57.4	61.3
QSR	63.1	**69.9**	**58.7**	**58.9**	**62.7**

The best result for each column is in bold

Table 10 False positive rate (%) results. The smaller the value, the lower the rate of mispredicting background regions as foreground. The results for baselines were produced using the original authors code, as no FPR results were reported in their papers

Method	P-5^0	P-5^1	P-5^2	P-5^3	Mean
CaNet	10.9	7.9	9.8	10.1	9.7
CaNet+QSR	**7.4**	**7.1**	**9.7**	**8.9**	**8.3**
ASGNet	8.7	8.8	9.6	9.2	9.1
ASGNet+QSR	**6.2**	**7.8**	**8.8**	**8.2**	**7.8**
PFENet	5.8	**7.7**	7.6	8.6	7.4
PFENet+QSR	**5.2**	7.9	**6.9**	**6.0**	**6.5**

The best result for each column is in bold

5.4.1 What latent classes represent

Latent classes (see Sect. 4.2) are used to represent classes that are undefined in the training dataset, but may correspond to unlabelled background features. To visualise these latent classes, we identified the three highest scores (see Eq. (8)) for latent classes, then generated a background prototype for each of these high-scoring latent classes in turn, and used those prototypes to segment the image. The results for two example images are shown in Fig. 4. Since QSR only predicts the background in the training phase, the figure shows the results for two training images.

It can be seen that each latent class represents a certain area of the background. This shows that the latent weights do represent the unknown categories of the background. However, these categories do not correspond to meaningful categories, that might be given distinct labels by a human. This is because QSR constrains the latent classes to be statistically independent from each other and the known classes. This constraint does not force latent classes to correspond to specific background classes, but allows them to learn combinations of background features. It can also be seen that when the background prototype is generated using all non-foreground classes, in the way we propose, that this prototype does an excellent job of identifying almost all background regions in the two example images. This is even the case (as shown for the chair example) when the situation is challenging due to the object occupying a very small proportion of the image and both the background and foreground in the query image having little similarity with the support image.

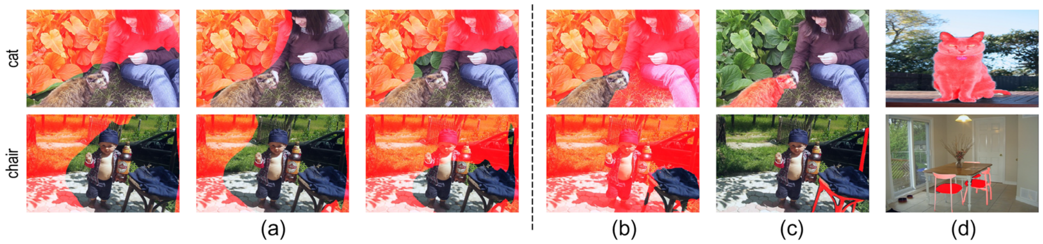

Fig. 4 Visualized results of latent classes and background predictions. For each class, **a** the prediction results for three latent classes, **b** the final background prediction, **c** the query with foreground masks, **d** the support image with foreground masks

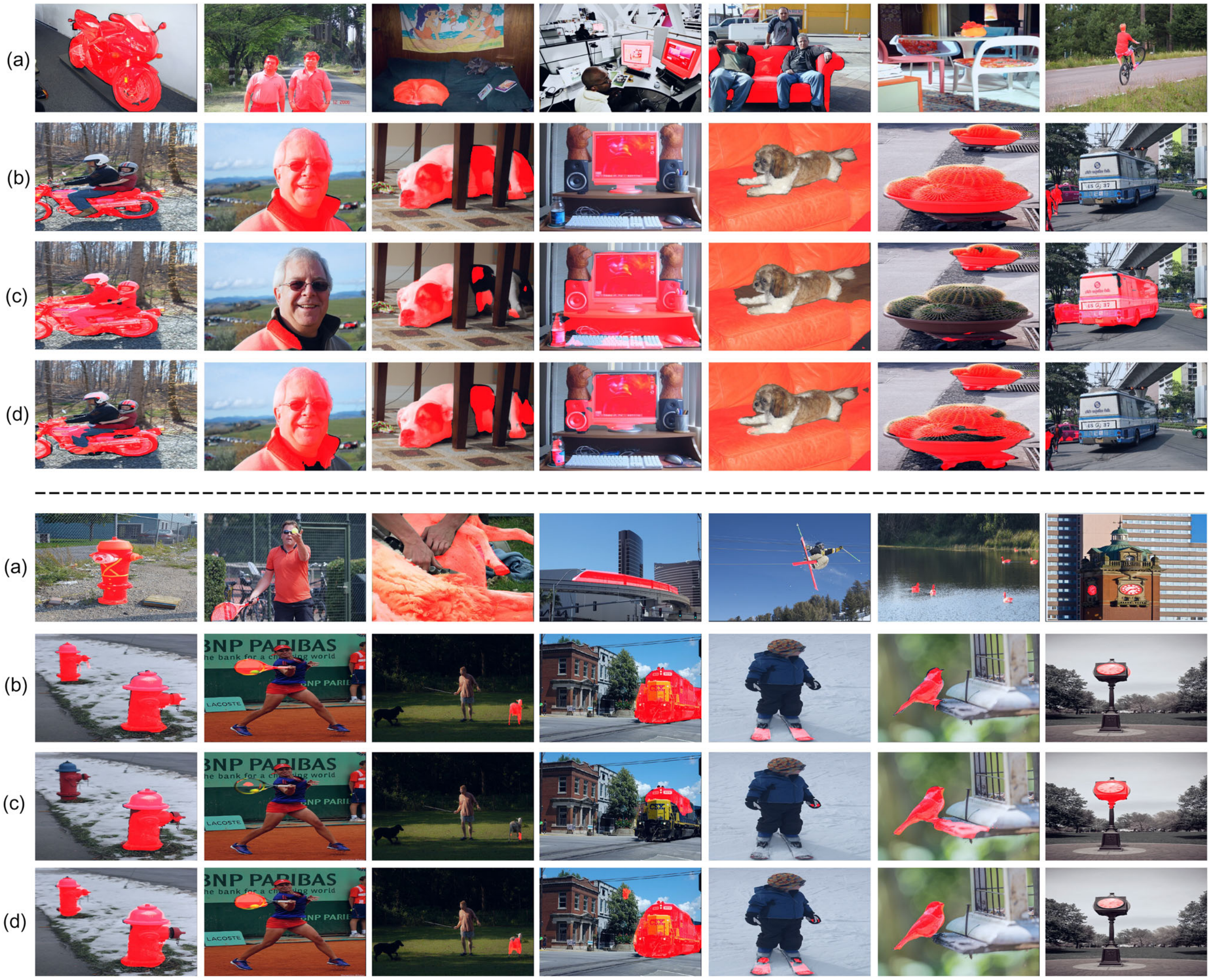

Fig. 5 Qualitative results for 1-shot FSS on PASCAL-5^i. **a** Support images and their ground-truths. **b** Query images and their ground-truths. **c** Predictions produced by PFENet. **d** Predictions produced by PFENet+QSR

5.4.2 False positive rate

QSR uses background information during training in order to make the model more descriminative and the foreground prototypes extracted during testing less likely to be matched with the background. The results shown in Table 10 demonstrate that QSR does indeed reduce the FPR compared to the corresponding baseline FSS algorithm.

5.4.3 Qualitative results

Fig. 5 shows some qualitative results. In the far right column above the line is an example of an unsuccessful segmentation, but a result where the false positive rate is reduced.

6 Conclusion

This paper proposes query semantic reconstruction (QSR) for few-shot segmentation. By associating the query image with semantics during training, QSR obtains background information from the query image to mine negative samples in order to make a more discriminative model that reduces false positives. QSR improves the performance of three different baselines, and for one of them the improvement is sufficient to produce state-of-the-art results for both the 1-shot and 5-shot settings on PASCAL-5^i. Future work might usefully explore improved methods of representing foreground objects or the use of background information at test time. In addition, due to limited computing resources, we did not tune the number of latent classes N_l (see Sect. 4.2) on COCO-20^i. Trying more N_l may produce better performance.

Acknowledgements The authors acknowledge the use of the research computing facility at King's College London, King's Computational Research, Engineering and Technology Environment (CREATE), and the Joint Academic Data science Endeavour (JADE) facility. This research was funded by the King's - China Scholarship Council (K-CSC).

Declarations

Conflict of interest The authors have no conflicts of interest/competing interests to declare that are relevant to the content of this article.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
2. He, J., Deng, Z., Zhou, L., Wang, Y., Qiao, Y.: Adaptive pyramid context network for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7519–7528 (2019)
3. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
4. Zhang, C., Lin, G., Liu, F., Yao, R., Shen, C.: Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5217–5226 (2019b)
5. Siam, M., Oreshkin, B.N., Jagersand, M.: Amp: Adaptive masked proxies for few-shot segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5249–5258 (2019)
6. Zhang, C., Lin, G., Liu, F., Guo, J., Wu, Q., Yao, R.: Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 9587–9595 (2019a)
7. Lu, Z., He, S., Zhu, X., Zhang, L., Song, Y.Z., Xiang, T.: Simpler is better: few-shot semantic segmentation with classifier weight transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8741–8750 (2021)
8. Liu, B., Ding, Y., Jiao, J., Ji, X., Ye, Q.: Anti-aliasing semantic reconstruction for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9747–9756 (2021)
9. Li, G., Jampani, V., Sevilla-Lara, L., Sun, D., Kim, J., Kim, J.: Adaptive prototype learning and allocation for few-shot segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2021)
10. Wu, Z., Shi, X., Lin, G., Cai, J.: Learning meta-class memory for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 517–526 (2021)
11. Zhang, B., Xiao, J., Qin, T.: Self-guided and cross-guided learning for few-shot segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8312–8321 (2021)
12. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**, 211–252 (2015)
13. Yang, B., Liu, C., Li, B., Jiao, J., Ye, Q.: Prototype mixture models for few-shot semantic segmentation. In: European Conference on Computer Vision, Springer, pp. 763–778 (2020)
14. Siam, M., Doraiswamy, N., Oreshkin, B.N., Yao, H., Jagersand, M.: Weakly supervised few-shot object segmentation using co-attention with visual and semantic inputs. In: IJCAI (2020)
15. Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: Panet: few-shot image semantic segmentation with prototype alignment. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 9197–9206 (2019)
16. Tian, Z., Zhao, H., Shu, M., Yang, Z., Li, R., Jia, J.: Prior guided feature enrichment network for few-shot segmentation. IEEE Trans. Pattern Anal. Mach. Intell. (2020)
17. Boudiaf, M.,Kervadec, H., Masud, Z.I., Piantanida, P., Ayed, I.B., Dolz, J.: Few-shot segmentation without meta-learning: a good transductive inference is all you need? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2021)
18. Yang, L., Zhuo, W., Qi, L., Shi, Y., Gao, Y.: Mining latent classes for few-shot segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 8721–8730 (2021)
19. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 234–241 (2015)
20. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European conference on computer vision (ECCV), pp. 801–818 (2018)
21. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2881–2890 (2017)
22. Hariharan, B., Girshick, R.: Low-shot visual recognition by shrinking and hallucinating features. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3018–3027 (2017)
23. Wang, Y.X., Girshick, R., Hebert, M., Hariharan, B.: Low-shot learning from imaginary data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7278–7286 (2018)
24. Chen, Z., Fu, Y., Zhang, Y., Jiang, Y.G., Xue, X., Sigal, L.: Multi-level semantic feature augmentation for one-shot learning. IEEE Trans. Image Process. **28**, 4594–4605 (2019)
25. Liu, J., Sun, Y., Han, C., Dou, Z., Li, W.: Deep representation learning on long-tailed data: A learnable embedding augmentation perspective. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2970–2979 (2020)
26. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning, PMLR. pp. 1126–1135 (2017)
27. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR (2017)
28. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA, pp. 4080-4090 (2017)

29. Grant, E., Finn, C., Levine, S., Darrell, T., Griffiths, T.: Recasting gradient-based meta-learning as hierarchical Bayes. In: 6th International Conference on Learning Representations, ICLR 2018 (2018)
30. Shaban, A., Bansal, S., Liu, Z., Essa, I., Boots, B.: One-shot learning for semantic segmentation. In: BMVC (2017)
31. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vis. **88**, 303–338 (2010)
32. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny, S.: Barlow twins: self-supervised learning via redundancy reduction. In: International Conference on Machine Learning (ICML) (2021)
33. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15750–15758 (2021)
34. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
35. Nguyen, K., Todorovic, S.: Feature weighting and boosting for few-shot segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 622–631 (2019)
36. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Simultaneous detection and segmentation. In: European Conference on Computer Vision, Springer, pp. 297–312 (2014)
37. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision, Springer. pp. 740–755 (2014)
38. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**, 834–848 (2017)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Haoyan Guan received a B. Eng. degree in Harbin Institute of Technology and MSc degree in Shanghai Jiaotong University. He is currently pursuing PhD degree in Computer Science at the Department of Informatics, King's College London. His research is concerned with the few shot learning and metric learning in computer vision.

Michael Spratling received a B. Eng. degree in Engineering Science from Loughborough University and MSc and PhD degrees in Artificial Intelligence and Neural Computation from the University of Edinburgh. He is currently Reader in Computational Neuroscience and Visual Cognition at the Department of Informatics, King's College London. His research is concerned with understanding the computational and neural mechanisms underlying visual perception.