In our previous structure, the network is trained on one-shot support set. Once Conv-LSTM is imported into our framework, it will enable us to train with k-shot support set. For every batch(if batch size=1), one query image and k support image masks will be fed into our neural network.

We unroll this procedure in Fig. 3 for better understanding the k-shot fusion process. k-shot support image masks enter Conv-LSTM in turn. Conv-LSTM plays a critical role in summarizing the total features of the k-shot support image masks.

For better mixing the feature from the support set in k-shot learning, a function loss is designed as follows:

$$L = -\frac{1}{ks^2}\sum_{i=0}^{k}\sum_{m=0,n=0}^{s} Y_{m,n} log X_{m,n} \tag{12}$$

where Y is binary label, X means the neural network output probability, k is shot number, s represents image size.

This loss enforces our A-MCG module to function well on **every** support set image rather than only supervises the segmentation of single support image.

Experimental Result

Training details

We implement our code based on the tensorflow framework (Abadi et al. 2016). Specially, a scaffold framework named tensorpack (Wu and others 2016) is used for quickly setting up our experiment. All our models are trained by Stochastic Gradient Descent(SGD) (Bottou 2010) solver with learning rate=1e-4, momentum=0.99 on one Nvidia Titan XP GPU. To fully fill GPU memory, we set the batch size 12. The weights of the support branch and the query branch are initialized with ImageNet (Deng et al. 2009) pre-trained weights. For the weight initialization of A-MCG module, Xavier initialization (Glorot and Bengio 2010) is adopted. All the images in the support and query branch are resized to 320×320. No further augmentation is employed except the image resizing. For Batch Normalization(BN) (Ioffe and Szegedy 2015), we employ current batch statistics at training and use the moving average statistics of BN during validation time.

We use the cross-entropy loss as the object function for training the network. The loss is summed up over all the pixels in a mini-batch.

When we experiment with Conv-LSTM for k-shot learning, we set k=5 by default. The max batch size can only be 6 because every time k support image masks will be fed into the support branch. Layer Normalization (Ba, Kiros, and Hinton 2016) is utilized in our Conv-LSTM for speeding up convergence.

Dataset and Metric

Dataset: We utilize dataset PASCAL-5^i (Shaban et al. 2017) to conduct our experiment. This dataset is originated from PASCAL VOC12 (Everingham et al.) and extended annotations from SDS (Hariharan et al.). The set of 20 classes in PASCAL VOC12 is divided into four sub-datasets as indicated in Table 2. Three sub-datasets are used as the training label-set L_{train}, the left one sub-dataset is utilized for test label-set L_{test}.

Table 2: PASCAL-$\boldsymbol{i^5}$ group information. The top table displays 4 groups of label and their semantic classes. The bottom table shows 4 sub-datasets and their training, validation components.

label set index	Semantic Classes
0	aeroplane, bicycle, bird, boat, bottle
1	bus, car, cat, chair, cow
2	dinningtable, dog, horse, motorbike, person
3	potted plant, sheep, sofa, train, tv/monitor

sub-dataset	train label set	val label set
0	1,2,3	0
1	0,2,3	1
2	0,1,3	2
3	0,1,2	3

The training set D_{train} is composed of all image-mask pairs from PASCAL VOC12 and SDS training sets that include at least one pixel in the segmentation mask from the label-set L_{train}. The masks in D_{train} are modified into binary masks by setting pixels whose semantic class are not in L_{train} as background class l_{ϕ}. The test set D_{test} is from PASCAL VOC12 and SDS validation sets, and the processing procedure for test set D_{test} is similar with training set D_{train}. Our evaluation mIoU is the average of 5 sub-dataset mIoUs. For a fair comparison with (Shaban et al. 2017), we take the same random seed and sample N=1000 examples for testing each of our models.

Metric: To compare the quantitative performance of the different models, mean intersection over union(mIoU) over two classes is used for our benchmark evaluation. For binary segmentation in our work, we first calculate the 2×2 confusion matrix, then compute the according IoU_l as $\frac{tp_l}{tp_l+fp_l+fn_l}$. tp_l is the number of true positives for class l, fp_l is the number of false positives for class l and fn_l is the number of false negatives for class l. The final mIoU is its average over the set of classes.

Ablation Study

Baseline. Our method is mostly compared with OSLSM (Shaban et al. 2017) and co-FCN (Rakelly et al. 2018). Both of them utilize the VGG (Simonyan and Zisserman 2014) as basic model. Different from them, we adopt ResNet101 (He et al. 2016) as our basic model, for ResNet101 owns much less parameter than VGG16, thus it is less prone to over-fitting. Besides, ResNet also enables larger batch size training in our architecture.

After removing the fully connected layers in the end, our ResNet101 baseline becomes a fully-convolutional structure. Support branch and query branch are fused by element-wise Add between the Res-5 output of them, followed by a naive convolution.