MULTI-PLATEAU ENSEMBLE FOR ENDOSCOPIC ARTEFACT SEGMENTATION AND DETECTION

Suyog Jadhav, Udbhav Bamba, Arnav Chavan, Rishabh Tiwari, Aryan Raj

Indian Institute of Technology (ISM), Dhanbad

ABSTRACT

Endoscopic artefact detection challenge consists of 1) Artefact detection, 2) Semantic segmentation, and 3) Out-of-sample generalisation. For Semantic segmentation task, we propose a multi-plateau ensemble of FPN[1] (Feature Pyramid Network) with EfficientNet[2] as feature extractor/encoder. For Object detection task, we used a three model ensemble of RetinaNet[3] with Resnet50[4] Backbone and FasterRCNN[5] (FPN + DC5[6]) with Resnext101 Backbone[7, 8]. A PyTorch implementation to our approach to the problem is available at github.com/ubamba98/EAD2020.

Index Terms- Endoscopy, FPN, EfficientNet, RetinaNet, Faster RCNN, Artefact detection.

1. DATASETS

The given dataset of EndoCV-2020 [9, 10, 11] has a total of 643 images for the segmentation task which we divided into three parts - train (474 images), validation (99 images) and holdout (70 images) in the sequence they were released. We made sure that the distribution of train and holdout were similar and that of validation was different. Validation set ensured that the model was not overfitting to the training data and at the same time generalizing well on holdout. For Detection, a similar strategy was adopted - train (2200 images), validation (232 images) and holdout (99 images)

2. METHODS

2.1. Data Pre-processing and Augmentations

Due to the variable aspect ratios and sizes in the training data, we adopted a stage-dependent rescaling policy. During the training stage, we cropped the images to a fixed size of 512x512 without resizing. This made sure that the spatial information was not lost and at the same time, the input to models was within trainable limits. During validation and testing time, we padded the images such that both dimensions are a multiple of 128 which is required for the EfficientNet backbone (to handle max-pooling in deeper models). As the number of samples in the dataset were relatively low and unbalanced, various augmentation techniques were adopted to prevent overfitting and achieve generalization. Horizontal and vertical flip, cutout (random holes)[12], random contrast, gamma, brightness, rotation were tested out. To strongly regularize the data we propose the use of CutMix for segmentation (Algorithm: 1) which was toggled on and off depending upon the variance of model outputs.

Algorithm 1 CutMix for Segmentation

```
for each iteration do
    input, target = get minibatch(dataset)
    if mode == training then
        input s, target s = shuffle minibatch(input, target)
        lambda = Unif(0,1)
        r_x = Unif(0,W)
        r_y = Unif(0,H)
        r_w = Sqrt(1 - lambda)
        r_h = Sqrt(1 - lambda)
        x1 = Round(Clip(r_x - r_w / 2, min=0))
        x2 = Round(Clip(r_x + r_w / 2, max=W))
        y1 = Round(Clip(r_y - r_h / 2, min=0))
        y2 = Round(Clip(r_y + r_h / 2, min=H))
        input[:, :, x1:x2, y1:y2] = input_s[:, :, x1:x2, y1:y2]
        target[:, :, x1:x2, y1:y2] = target_s[:, :, x1:x2, y1:y2]
    end if
    output = model_forward(input)
    loss = compute_loss(output, target)
    model_update()
end for
```

For Object detection spatial transformations - flip and random scaling and rotating were used.

2.2. Multi-Plateau Approach

Due to high variability and early overfitting nature in the dataset, the main focus was on making a strong ensemble by training models on different optimisation plateaus. A total of 8 different plateaus with permutations of two different optimisers and four different loss functions were optimised with EfficientNet backbone increasing the depth, width and resolution three times, going from B3 to B5 (Table 1). For

Table 1. Multi-Plateau Results

Encoder	Optimizer	Loss function	Validation (DICE)	FineTuning including Holdout
Efficientnet B3	Ranger	DICE	**0.4917**	0.4900
		BCE+DICE	0.4382	–
		BCE	0.4415	–
		BCE+DICE+JACCARD	**0.4771**	0.4630
	Over9000	DICE	0.4509	–
		BCE+DICE	0.4500	–
		BCE	0.4170	–
		BCE+DICE+JACCARD	0.4525	–
Efficientnet B4	Ranger	DICE	0.4568	–
		BCE+DICE	**0.4759**	0.4720
		BCE	0.4165	–
		BCE+DICE+JACCARD	0.**4718**	0.4666
	Over9000	DICE	0.3890	–
		BCE+DICE	0.4597	–
		BCE	0.4151	–
		BCE+DICE+JACCARD	0.4614	–
Efficientnet B5	Ranger	DICE	**0.4761**	**0.4987**
		BCE+DICE	0.4693	0.4643
		BCE	0.4374	–
		BCE+DICE+JACCARD	**0.4781**	0.4900
	Over9000	DICE	0.4352	–
		BCE+DICE	**0.4730**	0.4823
		BCE	0.4151	–
		BCE+DICE+JACCARD	**0.4726**	0.4798

optimisers, Ranger and Over9000 were used. Ranger is a synergistic optimiser combining RAdam (rectified Adam)[13] and LookAhead[14], and Over9000 is a combination of Ralamb[15] and LookAhead. A total of 2*4*3 = 24 models were trained, but in the final ensemble, only the models with a dice greater than 0.47 were considered. Average pixel-wise ensembling was adopted.

2.3. Multi Stage Training

Complete segmentation training pipeline was divided into four stages -
Stage 1 - CutMix was disabled to reduce regularization effect, encoder was loaded with ImageNet weights and freezed for the decoder to learn spatial features without being stuck into saddle point, crops were taken with at least one pixel having a positive mask.
Stage 2 - CutMix was enabled for strong regularization, and encoder was unfreezed to learn spatial features of endoscopic images.
Stage 3 - Random crops were trained instead of non-empty crops for the model to learn negative samples.
Stage 4 - Very few epochs with CutMix disabled and encoder freezed for generalization on original data.
For every consecutive stage best checkpoint of the previous stage was loaded.

2.4. Triple Threshold

After analysing predictions on holdout, it was found that the number of false positives was quite high. To counter this, we implemented a novel post-processing algorithm which specifically reduced the number of false positives in the predictions (Algorithm 2). Three sets of thresholds - max_prob_thresh, min_prob_thresh, min_area_thresh were tuned for this given task.

max_prob_thresh and min_prob_thresh were tuned using grid search on holdout dataset, whereas min_area_thresh was calculated by sorting sum of positive pixels of every class and taking the 2.5^{th} percent respectively. min_area_thresh used after calculation were 2000, 128, 256, 256 and 1024 respectively for every class. The results for triple threshold on single best model are compiled in Table 3 and comparison of our best performing models in Table 4.

Table 2. Object Detection Ensembling

Parameter	Values Tested	Description
iou_thresh	0.4, 0.5, 0.6	If two overlapping boxes have IoU value > iou_thresh, one of the boxes is rejected.
score_thresh	0.4, 0.5, 0.6	If a predicted box has a confidence value < score_thresh associated with it, the box is rejected.
weights	[1, 1, 1], [1, 1, 2], [1, 2, 1], [2, 1, 1], [1, 2, 2], [2, 1, 2], [2, 2, 1]	The weights are given to the predictions by each of the models. A model with higher weight has more influence on the final output than a model with a lower weight.

Algorithm 2 Triple Threshold

```
for each_sample do
    output_masks = model(each_sample)
    final_masks = []
    i = 0
    for each output_mask in output_masks do
        max_mask = output_mask > max_prob_thresh
        if max_mask.sum() < min_area_thresh[i] then
            output_mask = zeros(output_mask.shape)
        else
            output_mask = output_mask > min_prob_thresh
        end if
        i = i + 1
        final_masks.append(output_mask)
    end for
end for
```

Table 3. Triple Threshold Results

Min Thresh	Max Thresh	Val Precision
0.5	-	0.597
0.5	0.6	**0.601**
0.5	0.7	**0.608**
0.5	0.8	0.598
0.4	-	0.588
0.4	0.6	0.593
0.4	0.7	**0.600**
0.4	0.8	0.591

" - " indicates no triple threshold

2.5. Object Detection

For Object Detection, individual models were trained with SGD as optimizer and confidence threshold of 0.5. To counter the variance and improve the performance of our model predictions, general ensembling was performed. Retinanet with backbones of FPN and Resnet50 and Faster RCNN with backbones of FPN and Resnext 101 32xd were trained (Table 5). Our ensemble strategy involves finding overlapping boxes of the same class and average their positions while adding their confidences. For finding the best parameters for ensembling the three models predictions, we ran a grid search with all possible combinations of the given range of values (Table 2).

Table 4. Precision Values on Best Performing Models

Model	No Triple	Triple
B3-Ranger-DICE	0.492	0.494
B5-Ranger-DICE	0.597	**0.608**
B5-Ranger-BCE+DICE+JACCARD	0.549	**0.561**
B5-Over9000-BCE+DICE	0.520	0.530
B5-Over9000-BCE+DICE+JACCARD	0.505	0.515

Table 5. Object Detection Results

Model	Val. mAP	Hold. mAP
RetinaNet (FPN backend)	26.07	24.66
Faster RCNN (FPN backend)	20.11	21.47
Faster RCNN (DC5 backend)	27.64	26.15
Ensembled	**32.33**	**30.12**

3. RESULTS

We achieved a Segmentation score which was a weighted linear combination of dice, IOU and F2 of 0.5675 on the final leader board and for object detection task, an mAP of 0.2061 was obtained.

4. DISCUSSION & CONCLUSION

Gastric cancer accounts for around 1 million deaths each year which can be prevented by early diagnosis. In this paper we explored multi-plateau ensemble to generalize pixel level segmentation and localization of artefacts in endoscopic images. We developed novel augmentation and post-processing algorithms for better and robust model convergence.

5. REFERENCES

[1] Tsung-Yi Lin, Piotr Dollr, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2016.

[2] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019.

[3] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollr. Focal loss for dense object detection, 2017.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[5] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015.

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks, 2017.

[7] Saining Xie, Ross Girshick, Piotr Dollr, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2016.

[8] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2017.

[9] Sharib Ali, Felix Zhou, Christian Daul, Barbara Braden, Adam Bailey, Stefano Realdon, James East, Georges Wagnieres, Victor Loschenov, Enrico Grisan, et al. Endoscopy artifact detection (ead 2019) challenge dataset. *arXiv preprint arXiv:1905.03209*, 2019.

[10] Sharib Ali, Felix Zhou, Adam Bailey, Barbara Braden, James East, Xin Lu, and Jens Rittscher. A deep learning framework for quality assessment and restoration in video endoscopy. *arXiv preprint arXiv:1904.07073*, 2019.

[11] Sharib Ali, Felix Zhou, Barbara Braden, Adam Bailey, Suhui Yang, Guanju Cheng, Pengyi Zhang, Xiaoqiong Li, Maxime Kayser, Roger D. Soberanis-Mukul, Shadi Albarqouni, Xiaokang Wang, Chunqing Wang, Seiryo Watanabe, Ilkay Oksuz, Qingtian Ning, Shufan Yang, Mohammad Azam Khan, Xiaohong W. Gao, Stefano Realdon, Maxim Loshchenov, Julia A. Schnabel, James E. East, Geroges Wagnieres, Victor B. Loschenov, Enrico Grisan, Christian Daul, Walter Blondel, and Jens Rittscher. An objective comparison of detection and segmentation algorithms for artefacts in clinical endoscopy. *Scientific Reports*, 10, 2020.

[12] Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout, 2017.

[13] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond, 2019.

[14] Michael R. Zhang, James Lucas, Geoffrey Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back, 2019.

[15] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks, 2017.