A SUBMISSION NOTE ON EAD 2020: DEEP LEARNING BASED APPROACH FOR DETECTING ARTEFACTS IN ENDOSCOPY

Vishnusai Y, Prithvi Prakash, Nithin Shivashankar,

Mimyk Medical Simulations Pvt Ltd, Indian Institute of Science, Bengaluru, India

ABSTRACT

Deep neural network-based methods are becoming popular for disease diagnosis and treatment in Endoscopy. In this paper, we discuss our submission to Endoscopic Artefact Detection Challenge (EAD2020). The competition is part of grand challenges in Biomedical Image Analysis and consists of three sub-tasks[1]: i) Bounding box-based localisation of artefacts ii) Region-based segmentation of artefacts, and iii) Out of sample generalisation task.

For the first sub-task, we modify the Faster R-CNN object detector by integrating a powerful backbone network and a feature pyramidal module. For the second sub-task, we implemented a U-Net based autoencoder with a modified loss function to construct the semantic channels. For the third sub-task, we used ensembling techniques along with a data-augmentation technique inspired by RandAugment to boost the generalisation performance.

We report a Score$_d$ of 0.1869 ± 0.1076 for the first task, sscore of 0.5187 with a sstd of 0.2755 for the second task and mAP$_g$ of 0.2620 and a dev$_g$ of 0.0890 for the third task on the test data-set. Our method for the third task, achieves the third position on the leaderboard for the mAP$_g$ metric and also surpasses the results obtained by many methods in the previous EAD2019 challenge.

***Index Terms*—** Endoscopic Artefact Detection Challenge, Faster-RCNN, RandAugmentation, U-net.

1. INTRODUCTION

Endoscopy is widely used as a clinical procedure for early detection of numerous cancers (e.g., nasopharyngeal, oesophageal adenocarcinoma, gastric, colorectal cancers, bladder cancer, etc). It is also used for therapeutic procedures and minimally invasive surgeries (e.g. Laparoscopy). During this procedure, an endoscope which is a thin, long and flexible tube with a camera and a light source located at its proximal tip is used which helps to visualise the internal organs and helps for further diagnoses by the clinicians. A major drawback of the video frames obtained from this process, is that they are corrupted with multiple artefacts (for e.g. motion blur, pixel saturation, bubbles, fluid, debris, specularity reflections, etc) even though, the videos might be captured at a very high resolution. These artefacts prevent effective diagnoses of pathologies, post-analysis with respect to retrieving frames for report generation and video mosaicking for follow-ups. Thus, it becomes essential to use frame restoration algorithms, which helps to restore the frame to its highest quality. The frame restoration algorithms require accurate detection of the spatial location of multi-class artefacts in the corresponding frames. But present endoscopy workflow supports the restoration of only one type of artefact class, which is insufficient for high-quality frame restoration. So, it becomes essential to build multi-class artefact detectors which can lead to the development of artefact correction and frame restoration algorithms for each specific artefact class.

Endoscopic Artefact Detection challenge 2020[1], aims to address the key problem inherent in endoscopy. There are three sub-tasks in this particular challenge. They are: i) Bounding box localisation of multi-class artefacts. In this task, we are required to identify the class of the artefact, along with its spatial location by identifying the closest bounding box co-ordinates around the artefact. ii) Semantic segmentation of artefacts, where we are required to identify the class of the artefact along with its accurate region in the frame. Semantic segmentation is more effective than bounding box based localisation, because the region of interest, i.e. the artefact region is accurately marked in this task. iii) Out of sample generalisation task, where we are required to identify the type and the region of the artefacts through bounding boxes, from frames not captured for training purposes.

In our work, we demonstrate the following novelty:

- We modify the Faster R-CNN [1] module for the object detection task. We use a powerful version of the backbone ResNeXt-101 [2] for effective extraction of aggregated features. Additionally, we apply a feature pyramidal network (FPN) [3] module for multi-scale feature representation.
- In-order to improve generalisation, we came up with an augmentation technique inspired by RandAugment [4]. RandAugment provided one of the highest boost in AP on the COCO [5] and ImageNet [6] dataset. By

[1] https://ead2020.grand-challenge.org

Class labels	Number of instances
Specularity	11856
Imaging Artefacts	8681
Bubbles	5345
Contrast	1866
Saturation	1423
Blur	747
Instrument	603
Blood	528
Total	**31049**

Table 1. Statistics of the training data provided of EAD2020 challenge

using this technique, we achieved a significant increment in performance for Task 3. Using an ensemble of the improved Faster R-CNN and RetinaNet module along with the augmentation techniques, we achieved the third position in the leaderboard with respect to the mAP_g metric. We also demonstrate in Section 4.3 that our model surpasses the results obtained by methods used in the previous EAD2019 challenge. More details regarding the augmentation technique are provided in Section 3.3.

- For the second task, we use a U-Net [7] and similar augmentation techniques along with adopting Binary Focal Loss which is further detailed out in Section 3.2.

2. DATASETS

For the EAD 2020 challenge, two types of data-sets [8, 9, 10] were provided. These data-sets correspond to the two kinds of tasks, i.e. bounding box based localisation and semantic segmentation. The details of them are given below:

2.1. Dataset for bounding box based localisation task

The data-sets for this task were provided in three phases. There were eight classes of artefacts i.e. specularity, bubbles, saturation, contrast, blood, instrument, blur and imaging artefacts. In the first phase, we received 2200 endoscopic frames. In the second phase, we received 99 frames and in the third phase, we received five sets of sequential frames, totally adding to 232 in number. Table 1 provides the class-wise split of the total number of artefacts present in the total data provided.

From the table, it is can be inferred that there is data-imbalance between the classes. Specularity has the highest number of instances equal to 11856 whereas blood has the lowest number of instances equal to 528. Also, one more challenge observed with respect to this data-set is the non-uniformity of the image size/aspect ratio across the training images.

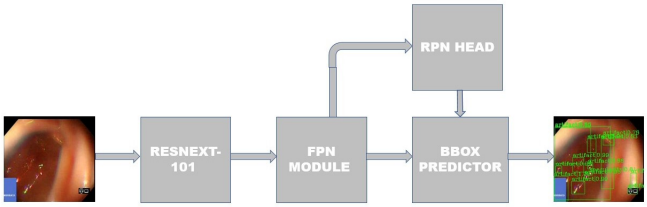

Fig. 1. Framework of the object detector used

2.2. Dataset for semantic segmentation task

The data-sets for this task were provided in three phases as well, with the first release having 474 samples, followed by the second release of 70 samples, capped off with a final release of 99 samples amounting to a total of 643 instances. With segmenting out the artefacts from images as the goal, each instance had an RGB Image of an arbitrary size paired up with a corresponding five-channel TIF mask file. The five channels from the masks represented Instruments, Specularity, Artefact, Bubbles and Saturation in that order. Each image had the possibility of overlapping masks.

3. METHODS

3.1. Multi-class Artefact Detection

Faster R-CNN module is a two-stage object detector containing a backbone with a feature extractor and a prediction module. Fig 1 shows the different components of the Faster R-CNN module. We discuss below the improvement made to the Faster R-CNN module to boost the performance towards the tasks.

3.1.1. Backbone Network

Backbone networks are used for the low-dimensional representation of input data. Usually, they are fully convolutional layers. The choice of a backbone network is crucial in determining how well the input data, in our case an image is encoded into a low-dimensional space. Typically, a stronger backbone network extracts effective features from the input image which leads to better accuracy of the output predictions.

In the EAD2020 train data-set (as extension of EAD2019), as observed, the class objects can be very small in size and difficult to differentiate from the background [10]. So, a need for a strong object detector module becomes necessary. The standard backbone networks used in literature

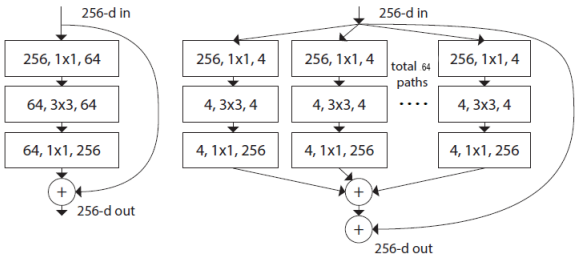

Fig. 2. a. The building of a ResNet module and b. The building block of a Resnext module with C=64

[11, 12, 13, 14, 15, 16, 17] are ResNet-50/101, VGG-16, Inception models, etc. To design a stronger backbone, we make use of the simple architectural design exhibited by ResNet-50/VGG-16. The number of hyper-parameters like the filter size and strides are fixed for each convolutional/residual block. Each block is followed by a downsampling step and after every stage of downsampling, the width of the blocks are multiplied by a factor of two. We build on this simple architectural design and improvise it by making use of the concept of simple split-transform-merge, as described in [2]. The input at each stage after the downsampling step is sent independently to *C* parallel residual blocks, where *C* refers to a hyper-parameter termed as cardinality. The output across each parallel residual block is concatenated before sending it to the next downsampling step. This model achieved state-of-the-art results on the imagenet dataset.

Given the complications of our data-set, we choose ResNet-101 having a cardinality *C* of 64, i.e. a ResNeXt-101 with a cardinality of 64. Fig 2 shows the building block of the ResNeXt-101 architecture. Inception based modules are based on a similar concept to the ResNeXt modules and achieve very good results. But it has complications with respect to hyper-parameter tuning. The filter size and strides need to be tailored for each stage and it is unclear as to how to adapt the architecture to new data-sets.

3.1.2. Feature Pyramidal Network module

FPN constructs an image pyramid by fusing intermediate layers from the backbone network. It is a top-down pathway consisting of lateral connections so the network efficiently constructs a rich, multi-scale feature pyramid from a single resolution input image. Since, the data-set in our case consists of both small and large-sized objects and also objects which are difficult to detect, using an FPN module builds a high-level semantic representation of the input image at both high and low resolutions, which helps for better predictions. To achieve this, we build an FPN module on top of the ResNeXt-101 backbone. We construct a pyramid with levels P3 through P7, where l indicates the pyramid level (Pl has resolution 2l lower than the input). Also, all pyramid levels have 256 output channels in concordance with [3].

3.1.3. Output prediction module

The output prediction module consists of two sub-components, the RPN head and the bounding box prediction module. The RPN-head proposes regions of interest from the intermediate feature representations coming from the FPN module. The bounding box prediction module is again a convolutional neural network. We chose to go with the standard modules for the RPN and the bounding box neural network as stated in [1].

3.2. Region based segmentation

The technique being employed for the Semantic Segmentation task was the U-NET Architecture Autoencoder [7]. Prior to training, we apply augmentations on the images and masks such as flipping, zooming, and rotating to increase the train sample size. The backbone networks loaded into our models are weights pre-trained on the ImageNet dataset. For the loss function, we make use of the Binary Focal Loss [18] as opposed to the traditional Binary Cross-Entropy Loss given that in most masks the negative pixels significantly outnumber the positive pixels. To gauge the performance of our models we make use of the Intersection over Union metric (IoU).

$$IoU = \frac{TP}{(TP + FP + FN)}$$

To prepare the data for training, the images and masks are scaled down to the uniform size of 256x256. Then, split in the ratio 80:20 of training to test data, followed by the aforementioned augmentations applied at random to both pools. We were able to accomplish this by making use of the inbuilt Keras [19] ImageDataGenerator class which provides high-level APIs to apply these in batches and the Segmentation Models Librar [20] for the various backbones for the U-NET.

3.3. Out-of-sample generalisation

The out-of-sample generalisation task requires us to detect artefacts from images different from what was obtained for training purposes. This requires that the neural network does not get over-fit to the training data while maintaining its performance on the test data.

Data-augmentation is a standard technique which performs label-preserving transformations to the input data-set. It is a well-known fact that data-augmentation improves the generalization capability of the deep neural network. Choosing the right transformations and adopting an effective methodology for implementing the transformations is crucial to achieve a high generalization score. Recently, a lot of research has gone into identifying effective data-augmentation methodologies. Cubuk et.al. [21] uses a technique termed

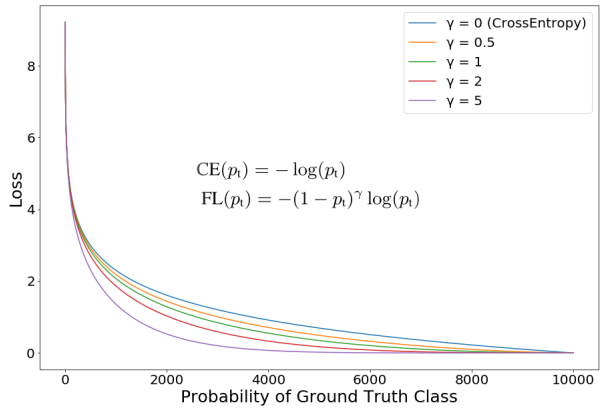

Fig. 3. Focal Loss [18].

as Auto-Augment, wherein a reinforcement learning based algorithm is used to select the most effective transformations suitable to a given data-set from a total pool of 16 operations. This technique achieved very high mAP on the COCO data-set. But, implementation of the methodology is complex and requires high computational capacity. In-order to design an effective augmentation technique which is also simple to implement, we make use of the idea used in [4].

The main idea is to randomly select N transformations from a total pool of T operations and apply it to the image sequentially with a magnitude M that can be varied in the range $[1, 10]$. This algorithm requires less computational power to implement and surprisingly achieved similar results as [21] on the COCO data-set. We fine-tune this algorithm to our data-set. By experimentation, we found that effective and original augmentations were being produced for *M = 4,5*. Also, we restricted the number of sequential augmentation techniques N to 2, due to computational limitations. For the data-set given, we selected the transformations based on a simple intuition that the output image should be label-preserving. The chosen augmentation techniques are *Equalize, Sharpness, Brightness, Rotate, Cutout, Translate-X/Y* and *Shear-X/Y*. Figure 4 shows examples of the augmented data.

To further increase the generalisation capability, we created an ensemble of the improved Faster R-CNN module from Section 3.1 along with a RetinaNet [18] object detector using Weighted Boxes Fusion technique [22]. The networks were allocated with equal weights. The threshold and intersection over union parameters were set at 0.0000001 and 0.6 respectively. The results of the above methods, along with a comparison with other networks are discussed in Section 4.

4. RESULTS

The results from the trained model on the test data-set are discussed next. The models were trained on NVIDIA GTX 1080Ti and RTX 2070 GPU's.

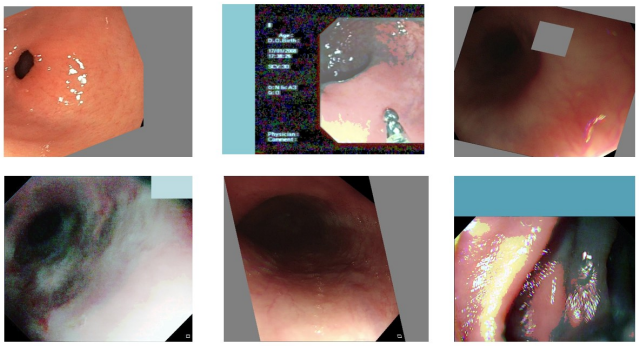

Fig. 4. Examples of the images generated through the augmentation technique

Backbone	**Score**$_d$
Modified ResNeXt-101 backbone	0.2319 ± 0.1005
ResNext-50	0.2004 ± 0.1540
ResNet-101	0.1456 ± 0.3564

Table 2. Comparison of the backbone used with other standard backbones

4.1. Task 1: Bounding box localisation based multi-class artefact detection

The test data-set was provided in two phases. The first phase contained 150 images and the second phase contained totally contained 317 images including the first phase images. As discussed in Section 3.1 we use an improved Faster R-CNN module for this task. To establish a comparison of our backbone with other standard backbones we trained the same Faster R-CNN with different backbones like ResNeXt-50 and ResNet-101. We compared the performance of the backbones by using only the first phase of the test data-set. Table 2 summarises the results. It can be observed that the modified ResNeXt-101 algorithm gives the highest Score$_d$ score over other backbones, thus demonstrating its effectiveness over other standard backbones.

Also, to establish a comparison with other state-of-the art object detection techniques, we also train a Cascade R-CNN and a RetinaNet detector with the same configuration of the ResNeXt-101 backbone. The training set images were resized to (1300,800) and the learning rate was set to 0.01. We also applied the augmentation techniques discussed in Section 3.3. We stopped the training after 12 epochs because the models showed signs of over-fitting. Table 3 summarises the obtained results on the final test data-set. It can be observed from the results that Faster R-CNN performed way better than the RetinaNet and Cascade R-CNN. However, Faster R-CNN model showed higher error of ± 0.1076.

Method	Score_d
Faster R-CNN	0.1869 ± 0.1076
RetinaNet	0.1725 ± 0.0989
Cascade R-CNN	0.1686 ± 0.0907

Table 3. Results obtained for Task 1 by different models

Architecture	Backbone	Train IoU	Validation IoU
U-Net	None	0.9513	0.1823
U-Net	ResNet50	0.9728	0.2141
U-Net	ResNext50	0.9642	0.2501

Table 4. Results on Train and Validation set

Method	sscore	sstd
U-Net+ResNext50	0.5187	0.2755

Table 5. Results obtained for Task 2 on Test Set

4.2. Task 2: Semantic Segmentation of Artefacts

As discussed in Section 3.2, we used the U-NET Architecture with different backbones to train on our augmented data. All models were trained for 150 epochs and implemented using the Segmentation Models [20] framework. On our test and validation data we had the following results:

We used the best performing Model on the validation data-set (U-Net with the ResNext50 Backbone) on the test Dataset which had 162 samples for our submission. The results are shown in Table 5.

4.3. Task 3: Out of Sample Generalisation

The test data-set for this task consisted of 99 frames. As discussed earlier in Section 3.3, we implemented the augmentation technique and produced meaningful label-preserving images. To demonstrate the effectiveness of the augmentation technique, we compare the performance of the improved Faster R-CNN model trained using the augmented images with the model trained without using the augmented data. Both the models were tested on the first phase of the test data-set to establish comparisons. Table 6 summarises the results. It can be clearly observed that the model trained using the augmented images easily surpasses the model trained without the augmented images by achieving a difference of 5.9% on the mAP_g metric.

We created an ensemble model of Faster R-CNN and RetinaNet to boost the performance for the generalisation task. To prove the effectiveness of the ensemble model, we provide comparisons with non-ensembled models. As shown in Table 7, the ensembled model achieved a mAP_g of *0.2620* and a dev_g of *0.0890*. Even though the Faster R-CNN module achieves the lowest dev_g score, the mAP_g score is higher for the ensemble based model. This score is recorded as the third highest in the leaderboard.

Also, we compare our entire methodology (Augmentation + Ensembling) adopted for this task, with methods used in the previous EAD2019 challenge. Mohammad et.al. [16] uses a RetinaNet with a Resnet-101 module. Gao et.al. [13] uses a Fast R-CNN-NAS module. The comparison for the same is provided in Table 8. It can be observed that our technique achieves the highest performance.

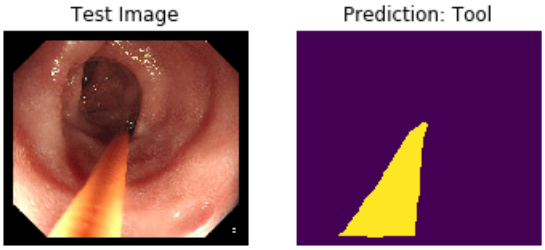

Fig. 5. Sample Result obtained from the U-NET Model for the Tool Detection Channel of the Semantic Segmentation Task

Method	mAP_g	dev_g
With the augmented images	0.2583	0.0680
Without the augmented images	0.1987	0.0533

Table 6. Comparison results of the improved Faster R-CNN model trained with and without the augmented data

Method	mAP_g	dev_g
Ensemble(Faster R-CNN + RetinaNet)	0.2620	0.0890
Faster R-CNN only	0.2583	0.0680
RetinaNet only	0.2393	0.0722

Table 7. Comparsion of results obtained for ensemble modeling

Method	Data-set	mAP_g	dev_g
Ensemble model	EAD2020	0.2620	0.0890
Faster R-CNN only (Ours)	EAD2020	0.2583	0.0680
Gao et.al. [13]	EAD2019	0.2515	0.0728
RetinaNet only (Ours)	EAD2020	0.2393	0.0722
Mohammad et.al. [16]	EAD2019	0.2187	0.0770
Yang et.al. [15]	EAD2019	0.1931	0.0478

Table 8. Comparison of our methods with EAD2019 [10] models

5. DISCUSSION & CONCLUSION

In this paper, we discuss the methods used for the EAD 2020 challenge and also present the corresponding results obtained. For the first task we used a improved Faster R-CNN module with a powerful backbone and a FPN module. For the second sub-task we used a U-Net architecture by modifying the loss function. For the third sub-task, we designed an augmentation technique inspired by RandAugment [4]. We also used an ensemble of Faster R-CNN and RetinaNet to further boost the results.

We demonstrate that the modified ResNext-101 backbone achieves better results than the standard backbones. For the first task, we achieved a Score_d of 0.1869 ± 0.1076. We also compare the results against other state-of-the-art techniques. For the second task, we achieve a sscore of 0.5187 and sstd of 0.2755. For the third task, we first demonstrate that the model trained on the augmented performs better than the model not trained on the augmented images. Then, we demonstrate that the ensembled based model performs better than non-ensembled models. Lastly, we show that our model achieves the third position in the leaderboard with respect to the mAP_g metric and that it performs better than a few models used in the EAD2019 challenge. We believe more research can be done towards identifying better semantic segmentation algorithms and better ensembling techniques in the future.

6. ACKNOWLEDGEMENT

We thank the EndoCV2020 organisers for the opportunity. We would also like to extend our thanks to Shanthanu Chakravarthy, Raghu Menon and Varun Seshadrinathan from Mimyk team for all the support during the competition.

7. REFERENCES

[1] Ren et.al. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS)*, page 9199, 2015.

[2] Saining Xie, Ross Girshick, Piotr Dollàr, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016.

[3] Tsung-Yi Lin et.al. Feature pyramid networks for object detection, 2016.

[4] Ekin Dogus Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical data augmentation with no separate search. *ArXiv*, abs/1909.13719, 2019.

[5] Tsung-Yi Lin et.al. Microsoft coco: Common objects in context, 2014.

[6] J. et.al. Deng. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241, 2015.

[8] 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.

[9] 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.

[10] 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.

[11] Suhui Yang and Guanju Cheng. Endoscopic artefact detection and segmentation with deep convolutional neural network. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[12] Ilkay Oksuz, James R. Clough, Andrew P. King, and Julia A. Schnabel. Artefact detection in video endoscopy using retinanet and focal loss function. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[13] Xiaokang Wang and Chunqing Wang. Detect artefacts of various sizes on the right scale for each class in video endoscopy. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[14] Maxime Kayser, Roger D. Soberanis-Mukul, Shadi Albarqouni, and Nassir Navab. Focal loss for artefact

detection in medical endoscopy. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[15] Shufan Yang and Sandy Cochran. Graph-search based unet-d for the analysis of endoscopic images. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[16] Mohammad Azam Khan and Jaegul Choo. Multi-class artefact detection in video endoscopy via convolution neural networks. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[17] Xiaohong W. Gao and Yu Qian. Patch-based deep learning approaches for artefact detection of endoscopic images. In *Proceedings of the 2019 Challenge on Endoscopy Artefacts Detection (EAD2019), Venice, Italy, 8th April*, volume 2366 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[18] Tsung-Yi Lin et.al. Focal loss for dense object detection, 2017.

[19] François et.al. Chollet. Keras. https://github.com/fchollet/keras, 2015.

[20] Pavel Yakubovskiy. Segmentation models, 2019.

[21] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data, 2018.

[22] Roman Solovyev and Weimin Wang. Weighted boxes fusion: ensembling boxes for object detection models, 2019.