CS482/682 Final Project Report Group 30

Robot Tool Segmentation

ekote1 awang91 aaugur1 jyoun127

12/13/2023

1 Introduction

A. Background The da Vinci robot is a state-of-the-art machine for minimally invasive surgeries, but it still requires manual operation by surgeons. To automate surgeries, fast and precise vision-based tool segmentation is crucial. There are many segmentation models in the deep learning field with various architectures and performance specifications. In this project, our team seeks to apply three existing models (DeepLabV3, FCN ResNet-50, and TernausNet) to segment endoscopic camera images into three classes: tool, flesh, and background. We evaluate the chosen models to determine the best-performing model for deployment, and fine-tune it for our application. We also aim to propose model features that contribute to strong performance – information that could be used in creating new models.

B. Related Work In general, models that contain some encoder-decoder architecture have been shown to perform well on image segmentation tasks [3]. The issue of reduced feature resolution due to repeated down-sampling layers in traditional, deep architectures has been addressed with different approaches like atrous convolution [2] or fully convolutional layers [1]. Fully Convolutional Residual Networks (FCRN) and improvements upon U-Net using pre-trained encoder weights have also been shown to perform well in segmentation challenges like EndoVis [4].

2 Methods

A. Dataset We used a Kaggle dataset containing 8080 sets of endoscopic images with labels and masks. To speed up processing and reduce RAM consumption, all images are scaled down to 127x127. Then, we split our data into three sets: 70% for training, 10% validation, and 10% testing. Augmentation is then applied to the training set – random horizontal, and vertical flips.

B. Models We selected three models to test: DeepLabV3, FCN ResNet-50 (FCRN), and TernausNet. We downloaded pre-trained weights for DeepLabV3 and FCRN from PyTorch, and TernausNet from GitHub source [3].

1. DeepLab V3 A hallmark of DeepLab V3 is the atrous convolution. This allows control over the spatial context and resolution of feature reponse, which extracting more features and retaining the same number of parameters as regular convolution [2]. This makes it a powerful tool for segmentation.

2. FCRN A Fully-Convolutional Network with a ResNet-50 backbone, adapted from contemporary classification networks (e.g.: AlexNet, VGG, and GoogLeNet) [1]. Its residual blocks provides greater capacity to capture more complex features and more efficient backpropagation in the 50-layer network.

3. TernausNet [3] A modification of U-Net architecture that uses a VGG16 encoder pre-trained on ImageNet for image segmentation. This increases its performance compared to U-Net, while also considerably reducing training time.

C. Training Procedure For all three models, we applied the same data split (train, test, and validation) and pre-processing. We also chose the same optimizer (ADAM) and used learning rate schedulers. This is to ensure that the model comes to convergence. We research the types of training loss function that works best for each model. DiceLoss was used for DeepLabV3 and FCRN. Cross Entropy Loss was used for TernausNet. For details on hyperparameters used to train the model, refer to Table 1.

D. Evaluation Metrics and Loss Functions Our primary metrics for evaluating model performance are: test accuracy and test loss, because the accuracy of tool segmentation is foundational to reliable automated surgery. We also consider inference time, because the model cannot be prohibitively slow for field deployment. Inference for all models are executed on the same platform so the results are comparable. Test loss uses cross-entropy loss. Test accuracy

Table 1: Model Hyperparameters

Model	Epochs	LR	Scheduler Gamma	Step size
FCRN	5	.001	.1	5
DeepLabV3	3	.001	.8	5
TernausNet	8	.001	.8	4
TernausNet + DropOut	3	.001	.8	4

Table 2: Test Results from Initial Test

Model	FCRN	DeepLabV3	TernausNet
Test Loss (std.)	0.06(±0.01)	0.03(±0.003)	0.05(±0.021)
Test Accuracy (std.)	0.28(±0.002)	0.28(±0.004)	0.29(±0.005)
Inference Time(ms)	45752.07	56275.32	49227.80

is measured with IOU, which is a good measurement for segmentation alignment. For our use case, the segmentation of the tools is more important than the flesh or the background, so we implemented a custom-weighed IOU to reflect this:

$$\text{Total IoU} = 0.4\text{Flesh IoU} + 0.4\text{Tool IoU} + 0.2\text{Background IoU}$$

E. Attempt to Improve Performance After collecting test results (see Table 2), we determined that TernausNet was the most performant model. To fine-tune its performance, we applied dropout at a rate of 0.5 before each max-pooling layer (Refer to Discussion 4B for our reasoning in choosing dropout). We also noticed that TernausNet converged faster with fewer epochs, so we changed from 8 to 3 epochs in the fine-tuned TernausNet (see Table 1 for hyperparameters).

Since we already trained three models that perform similarly on the same task, we also tried model-ensembling to improve performance; we gave each model equally weighted votes.

3 Results

A. Results from Initial Test Table 2 contains the complete test results data (loss, accuracy, and inference time, with standard deviation). To qualitatively compare results, we visualized predicted masks of TernausNet and DeepLabV3, our best models (see Figure 1). Refer to Discussion 4B for visual analysis.

B. Results from Improvement Attempt *Note: All comparisons below are made against vanilla TernausNet (see Table 2 for statistics).*

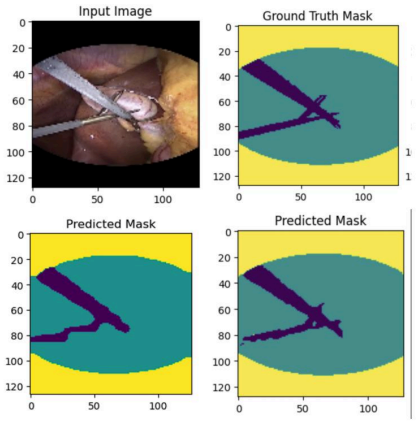

Figure 1: Masks For DeepLab (left) and TernausNet (right)

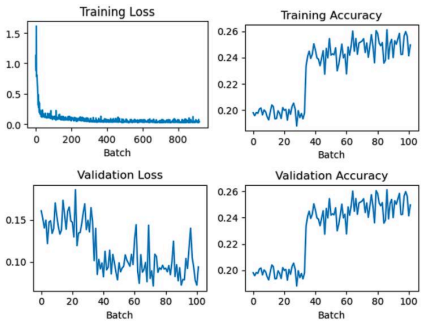

Figure 2: Ternaus with Dropout

Our TernausNet with added dropout had worse test accuracy and loss. Mean test accuracy is now 0.25 ± 0.005 (previously 0.29±0.005), and mean test loss is now 0.09 ± 0.020 (previously 0.05±0.021).

Ensembling our three models didn't change test accuracy, but made test loss worse. Mean test accuracy is now 0.29 ± 0.004 (previously 0.029 ± 0.005), and mean test loss is now 0.10 ± 0.025 (previously 0.05 ± 0.021).

4 Discussion

A. Initial Test of All Three Models While all three models had similar test accuracy (with TernausNet having the highest), DeepLab and TernausNet had noticeably better test loss than FCRN. Next, between DeepLab and TernausNet, TernausNet had 12.5% faster inference time (7.05s less). Therefore,

we chose to rank TerausNet first, DeepLabV3 second, and FCRN last.

The FCRN's lack of encoder-decoder architecture could explain its poor test loss, because the receptive field of FCNs in general only grow linearly with network depth. It is sometimes unable to capture long range contextual information effectively [5], making it a poor choice for our use case.

B. Model Improvement We decided to apply dropout to TernausNet because we noticed that our dataset, although large, had limited variation. We also decided to use model ensembling because we noticed that TernausNet was better at segmenting the tooltips (which contain more detail), while DeepLab and FCRN were better at the overall structure of the tools (see Figure 1). However, our modified models performed worse than our initial models (Results 3B). This makes sense for dropout, because it is known to sometimes cause variable or worse results. For our ensemble model, the test accuracy was worse than vanilla TernausNet. We believe this could due to DeepLab and FCRN having to the same weakness (lack of precision) while having twice the voting power than TernausNet (given equal weighted voting).

C. Summary and Next Steps In the visualized masks comparison of TernausNet and DeepLabV3 (Figure 1), we clearly see TernausNet is better at segmenting fine details (which is important for accurate tool segmentation). This tells us that its ImageNet pre-trained VGG-11 encoder likely outperforms DeepLab's atrous convolution in capturing high-frequency details. This hypothesis could be tested and, if verified, suggest that VGG based encoder should be used in creating new models. We can also try more ways of fine-tuning hyperparameters and adding regularization to improve TernausNet performance.

OneDrive Project Link Click here for our project OneDrive

References

[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *CoRR* abs/1411.4038 (2014). arXiv: 1411.4038. URL: http://arxiv.org/abs/1411.4038.

[2] Liang-Chieh Chen et al. "Rethinking Atrous Convolution for Semantic Image Segmentation". In: *CoRR* abs/1706.05587 (2017). arXiv: 1706.05587. URL: http://arxiv.org/abs/1706.05587.

[3] Vladimir Iglovikov and Alexey Shvets. "TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation". In: *CoRR* abs/1801.05746 (2018). arXiv: 1801.05746. URL: http://arxiv.org/abs/1801.05746.

[4] Max Allan et al. "2017 Robotic Instrument Segmentation Challenge". In: *CoRR* abs/1902.06426 (2019). arXiv: 1902.06426. URL: http://arxiv.org/abs/1902.06426.

[5] Li Zhang et al. "Dual Graph Convolutional Network for Semantic Segmentation". In: *CoRR* abs/1909.06121 (2019). arXiv: 1909.06121. URL: http://arxiv.org/abs/1909.06121.