PGR207 - Deep Learning Examination Report

Candidate nr.2, Candidate nr.24
School of Economics, Innovation, and Technology
Kristiania University College
Oslo, Norway

CONTENTS

***Abstract*—This report explores how various parameters affect the performance of deep learning models. Specifically, it evaluates the impact of data augmentation techniques, resolution, and training data size, given the same U-Net model architecture. Experiments were conducted on the Kvasir-Instrument dataset, a collection of annotated images for gastrointestinal endoscopy, to assess how these factors influence segmentation performance.**

The findings reveal that resolution changes and augmentation methods can enhance model generalization when appropriately applied, but excessive or unbalanced scaling may lead to overfitting. Moreover, while increasing training data size consistently improved performance, the returns diminished with larger datasets. The study highlights the importance of striking a balance between dataset size and augmentation strategies.

I. INTRODUCTION

This is an examination report in which the group was given a choice between five different experiments using the Kvasir-Instrument dataset. We decided to set a goal for ourselves and then choose experiments based on that. The goal we set for ourselves was to evaluate the extent to which additional data contributes to improved generalization versus overfitting. Before we continue, it's important to note that due to the data being from a single source, we cannot assess broader 'generalization' beyond this specific dataset. So when we talk about generalization we explicitly mean the test set from Kvasir-Instrument Dataset. With that in mind, the experiments chosen were to evaluate how data size (experiment III), augmentation methods (experiment II), and image resolution (experiment I) affect the models test performance. Each experiment is evaluated separately to assess individual performance, after which selected parts are combined to evaluate their overall effect on the model's generalization.

II. METHODOLOGY

In the Methodology section, each of the different technologies and tools relevant to the report is introduced, including an overview of the dataset experimented on and the different methods used in said experiments.

A. Kvasir-Instrument Dataset

The Kvasir-Instrument dataset consists of 590 annotated images of gastrointestinal (GI) endoscopic procedures, capturing various diagnostic and therapeutic tools such as snares, biopsy forceps, and balloons. The dataset includes the images themselves, ground truth segmentation masks, and bounding boxes to aid in the development of image segmentation and localization models. The images and their corresponding masks

share identical names, differing only in format: images are JPEG, and masks are PNG. Images vary in resolution from 720x576 to 1280x1024 pixels. A predefined train-test split is provided to support consistent evaluation. The dataset is openly accessible for research and educational purposes. [1]

B. Convolutional Neural Network

A Convolutional Neural Network (CNN) is a class of feedforward neural networks (FNN) designed for grid-like data structures such as images. One main difference between CNNs and FNNs is that CNNs do not fully connect each layer. This results in CNN models processing data in smaller regions at a time, maintaining spatial relations. The key distinguishing feature of CNN architectures are the convolution and pooling layers. Convolution layers extract features by using filters that are learnable matrices that detect specific features like shapes or textures. Pooling layers reduce spatial dimensions by summarizing regions of the input by using max pooling or average pooling. This preserves important features while discarding less relevant details.

This makes CNN models especially effective for image data, as they capture basic, local patterns like edges, basic shapes, and textures before progressively identifying more complex features. [2]

C. U-Net Architecture

The U-Net architecture is a variation of the CNN model introduced by Ronneberger in 2015 and was specifically designed for image segmentation tasks that require precise localization. [3] U-Net introduces an encoder-decoder structure that both captures both spatial information and broader context. This could allow the model to distinguish a surgical tool from surrounding tissues by recognizing the tool's curvature or lack thereof compared to its environment.

There are two things distinguishing U-Nets from CNN models; the encoder-decoder layers and the use of skip connections between corresponding layers. The encoder downsamples the input image and extracts feature representations by applying convolution and pooling operations. The decoder then upsamples these features back to the original image size using transposed convolutions or upsampling layers. Skip connections link corresponding encoder and decoder layers, helping to retain spatial details by directly transferring feature maps from the encoder to the decoder. This allows the network to produce detailed segmentation maps.

D. PyTorch

PyTorch is an open-source machine learning framework developed by Facebook's AI Research lab (FAIR). It includes modules like `torch.nn` for building neural networks, `torch.optim` for optimization algorithms, and `torch.utils.data` for data loading and preprocessing. Compared to competitors like TensorFlow and Keras, PyTorch includes built-in integration for GPU acceleration through CUDA, and uses easy to understand modular building blocks which streamline the model development process [4].

E. Data Augmentation

Augmentation methods in the context of image data are methods used to artificially expand the training dataset. Due to the nature of deep learning models, an image with one of its axis flipped will become a completely new data point. The same applies to if we rotate the image to some extent or crop parts of it out. To this end three different augmentation methods were used; rotation, horizontal flip, random crop. Rotation augmentation rotates an image by a random degree within a specified range, horizontal flip mirrors the image along its vertical axis, and random crop removes a portion of the image. Random crop specifically simulates partial occlusions or shifts in the object's position. The other two methods are self-explanatory.

F. Model Evaluation Methods

Each model is evaluated primarily on IoU (Intersection over Union) and Dice Coefficient; loss, training time, Precision, Recall, F1 Score, and Pixel Accuracy are also considered within this report.

IoU measures the overlap between the predicted segmentation and the ground truth. It is calculated as the ratio of the area of intersection to the area of union between the predicted and true masks.

Dice Coefficient evaluates the similarity between the predicted and ground truth masks. It is calculated as twice the area of intersection divided by the total number of pixels in both masks.

Precision represents the proportion of correctly predicted positive pixels out of all pixels predicted as positive. A high precision score means there are few false positive pixel predictions.

Recall measures the proportion of correctly predicted positive pixels out of all actual positive pixels. A high recall score means there are few false negative pixel predictions.

F1 Score is the harmonic mean of Precision and Recall.

Pixel Accuracy is the ratio of correctly classified pixels (both positive and negative) to the total number of pixels.

Time and epoch are also displayed in the graphs and tables below. While these metrics represent the duration and iterations recorded for the models, they should not be interpreted as definitive measures, as time and epoch counts varied greatly throughout the experiment. These metrics are included to give context to the reader. There are times where a model out perform the trends being observed due to running much longer than the rest. There has also been instances of models triggering early stopping up to 5 epochs earlier in one run compared to the other.

III. Experiments & Results

In this section the experiments themselves are explained as well as their results. It begins with a walkthrough of the base model, followed by details about changes made for each experiment and how they were conducted. At the end of each experiment section the corresponding results are displayed and

explained. Every experiment is built using the base model and its parameters unless specified otherwise.

A. Base Model

The base model is what each experiment are built upon and evaluated against. It has a basic U-Net architecture oriented towards binary segmentation.

1) Architecture:

– **Encoder (Contracting Path)**:
  - Consists of four encoder blocks.
  - Each block contains two convolutional layers with ReLU activations and batch normalization.
  - Followed by a max-pooling layer to reduce spatial dimensions.
  - The number of feature channels doubles at each subsequent block (64, 128, 256, 512).

– **Bottleneck**:
  - Contains two convolutional layers with 1024 filters each.
  - Includes dropout with a rate of 0.1 to prevent overfitting.

– **Decoder (Expanding Path)**:
  - Comprises four decoder blocks corresponding to the encoder blocks.
  - Each block begins with a transposed convolution (upconvolution) to upsample feature maps.
  - Features from the encoder are concatenated with the upsampled output.
  - Followed by two convolutional layers with ReLU activations.
  - Dropout layers are included in some larger blocks to enhance generalization.
  - The number of feature channels halves at each subsequent block (512, 256, 128, 64).

– **Final Output Layer**:
  - A single convolutional layer with a 1×1 kernel to produce the segmentation map.

The model uses `BCEWithLogitsLoss` for its loss function. This function was chosen due to its numerical stability, as it combines a Sigmoid layer and the Binary Cross Entropy Loss into a single class. It is well-suited for binary segmentation tasks because it allows for independent classification of each pixel, effectively handling the binary nature of the problem and enabling precise localization [5].

Adam (Adaptive Moment Estimation) was chosen as the base optimizer. It is an optimization algorithm that combines the benefits of AdaGrad and Momentum. It adapts the learning rate for each parameter based on estimates of the mean and variance (average of the squared gradients) of the loss gradients.

Lastly, `ReduceLROnPlateau` was chosen as the scheduler. It dynamically adjusts the learning rate based on validation loss. This scheduler monitors the validation metric and reduces the learning rate by a specified factor 0.5 if no improvement is observed for a set number of epochs. This is to prevent the model from being stuck in on local minima [6].

2) Code Structure: The base model is implemented using PyTorch and is organized into distinct components to building multiple different experiments easier. the components are as follows: imports, wanDB initialization, data handling, model architecture, training with validation, and evaluation.

- **Data Loading and Preprocessing**: A custom dataset class, `KvasirInstrumentDataset`, is defined to manage the loading and preprocessing of images and masks from the Kvasir-Instrument dataset. This includes resizing images to a target size of 576x576, normalizing pixel values, and applying padding transformations to prepare the data for training. The target size was chosen based on the smallest image size found in the base dataset.
- **Model Architecture**: The U-Net architecture is encapsulated within the `UNet` class. This class defines the encoder, bottleneck, decoder, and final output layers, incorporating convolutional layers, ReLU activations, dropout layers for regularization, and transposed convolutions for upsampling. Skip connections are implemented to make possible the transfer of spatial information from the encoder to the decoder.
- **Training Loop**: The training process is managed through a dedicated training loop that iterates over the dataset for a specified number of epochs. It utilizes the Adam optimizer and the `BCEWithLogitsLoss` loss function. (`ReduceLROnPlateau`) adjusts the learning rate based on validation loss. Early stopping is also incorporated to prevent overfitting by halting training when validation loss ceases to improve.
- **Evaluation Metrics**: Functions are implemented to calculate key evaluation metrics such as Intersection over Union (IoU), Dice coefficient, precision, recall, and pixel accuracy. These metrics provide a comprehensive assessment of the model's segmentation performance, ensuring that both the accuracy and the quality of the segmentation masks are thoroughly evaluated.

3) Regularization Techniques: To enhance the generalization capabilities of the model and prevent overfitting, the following regularization techniques were implemented:

- **Dropout Layers**: Incorporated within the bottleneck and specific decoder blocks with dropout rates of 0.1 and 0.2 respectively. This randomly deactivates a subset of neurons during training, preventing neurons from becoming overly specialized.
- **Weight Decay**: Applied a weight decay of 1×10^{-5} in the optimizer to penalize large weights.

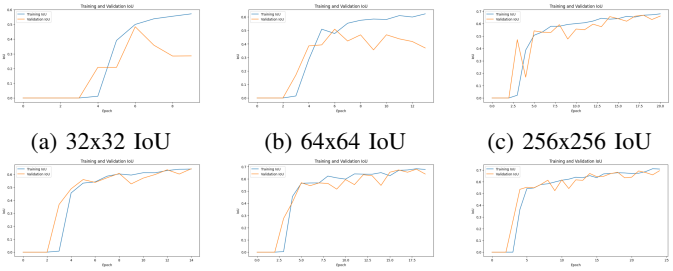
B. Experiment I – Image Resolution

In Experiment I, the effect that image resolution has on model performance is explored. Separate models were created for each resolution: 32x32, 64x64, 256x256, 720x720, 1024x1024, and 2048x2048. A combined model containing the base model, 64x64, 256x256, 720x720 and 1024x1024 data were also created. This was in part done to evaluate the effectiveness of resolution changes as a method of data augmentation.

TABLE I: Resolution Model Test Results

Model	IoU	Dice Coefficient	Epoch	Training Time
Base(576x576)	.698	.796	10	6m 50s
32x32	.344	.458	9	4m 5s
64x64	.434	.548	13	5m 45s
256x256	.735	.824	20	8m 56s
720x720	.728	.823	14	6m 45s
1024x1024	.717	.789	19	9m 29s
2048x2048	.771	.851	20	15m 32s
combined	.771	.820	12	27m 28s

The results vary to some degree, the models trained on lower resolution data tend to be less accurate and generally score slightly lower on both IoU and Dice. However going above the highest resolutions found within the base dataset, and effectively up-scaling these images resulted in a weaker model again.



(a) 32x32 IoU (b) 64x64 IoU (c) 256x256 IoU

(d) 720x720 IoU (e) 1024x1024 IoU (f) 2048x2048 IoU

Fig. 1: IoU Comparison Across Resolutions. Blue: Training IoU, Orange: Validation IoU.

The lowest-resolution models (32x32 and 64x64) were significantly less stable, exhibiting rapid overfitting after just a few epochs. There is a clear trend showing that a higher resolution increases the model performance. The only model slightly deviating from this was the 1024x1024 model, which dipped slightly compared to the rest, despite training for 19 epochs.

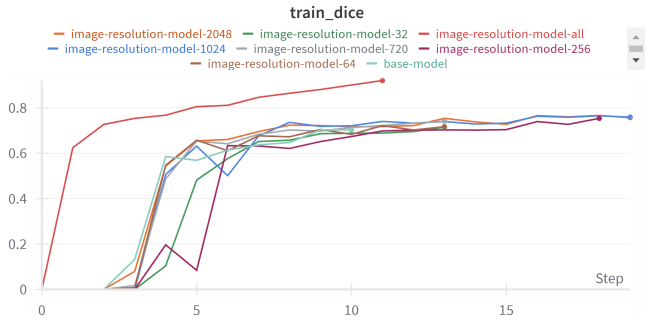


Fig. 2: Training Dice Coefficient Graph for resolution models

The combined model far outperformed the others, and generalized much better than other resolution models, which can be clearly seen in Figure 2. This indicates that resolution changes has significant performance influence as a augmentation method for this dataset.

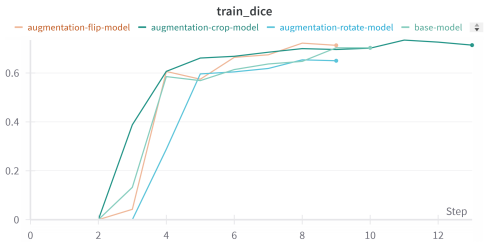


Fig. 3: Training Dice Coefficient for Data Augmentation experiments.

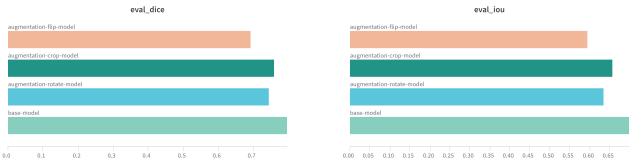
C. *Experiment II – Data Augmentation*

Experiment II tests the effects of various data augmentation techniques on a U-Net model. The following augmentations where chosen:

- **Rotation**: Random rotation of images within a range of ±360 degrees.
- **Horizontal Flip**: Horizontal flipping of the entire dataset.
- **Random Crop**: Random cropping retaining between 50% and 80% of the original images.

TABLE II: Augmentation Model Test Results

Model	IoU	Dice Coefficient	Epoch	Training Time
Base	.698	.796	10	6m 50s
Rotation	.635	.744	9	4m 29s
Crop	.657	.760	13	6m 13s
Flip	.593	.693	9	4m 21s



(a) Augmentation Dice (b) Augmentation IoU

Fig. 4: Figure showing evaluation results for augmentation models.

These augmentations were applied individually to evaluate how the base model would perform using only the augmented data. As can be seen in figure 3 these generally performed about the same as the base dataset did. Although with a slightly worse IoU and Dice on the evaluation set.

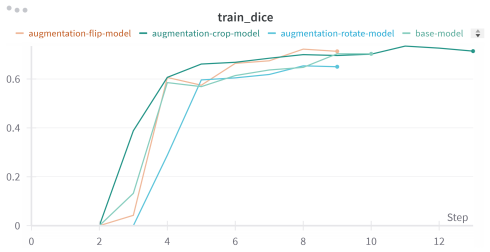


Fig. 5: Graph depicting augmentation model training dice scores.

Every augmentation model took at most three epochs to start learning and never spent more than three epochs before sharply tapering of towards diminishing returns. None of them deviated much away from the base dataset during training as can be seen in figure 5, which makes sense as they are for the most part replicas of said dataset.

D. Experiment III – Training data size

We investigate the impact of data size on model performance. The experiment involved training the model on different proportions of the training dataset. The results are shown below.

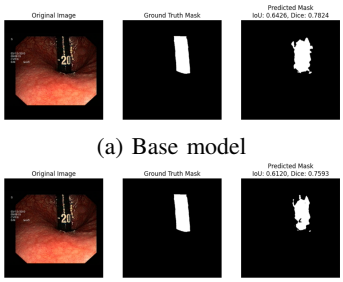
- **100% (Base model)**: With 424 images.
- **75%**: With 318 images.
- **50%**: With 212 images.
- **25%**: With 106 images.

TABLE III: Data Size Model Test Results

Model	IoU	Dice Coefficient	Epoch	Training Time
Base	.698	.796	10	6m 50s
75%	.680	.780	7	4m 23s
50%	.635	.748	10	5m 5s
25%	.534	.644	11	2m 29s

Fig. 6: Training Dice score for data size models

The results in general show that the model's trained on less data scores lower on IoU and Dice. An interesting observation shown in figure 6 here is that the models trained on fewer data points take a significant number of epochs to start improving.



(a) Base model

(b) 75%

(c) 50%

(d) 25%

Fig. 7: Example of predicted masks vs actual mask.

Figure 9 shows the predicted masks trend towards higher accuracy the more data points present in the training set. While this example visually highlights the trend, it is important to note that not all predictions were as clear-cut. But the trend generally seems to hold up.

There were also several models using both the base dataset and augmented data made. This was done to artificially increase the total amount of training data to test how the model would perform with more than just the base dataset. The results are as follows:

- **100% (Base model)**: With 424 images.
- **200%**: With 754 images. Using rotation augmentation to increase the training data.
- **400%**: With 1508 images. Using all augmentation methods explored in experiment II to increase the training data.
- **400%**: With 1650 images. Using all the different resolution changes in experiment I to increase the training data.
- **900%**: With 3960 images. Using 256, 576, 720 resolution changes from experiment I and also applying every augmentation from experiment II with each resolution.

TABLE IV: Data Size experiments for larger data size

Model	IoU	Dice Coefficient	Epoch	Training Time
Base	.698	.796	10	6m 50s
rotation + base	.753	.835	12	11m 7s
all augments	.809	.877	17	44m 40s
resolution	.766	.846	12	27m 28s
900%	.860	.893	10	48m 54s

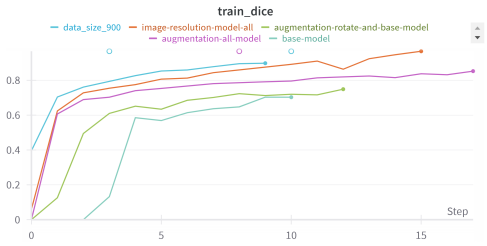
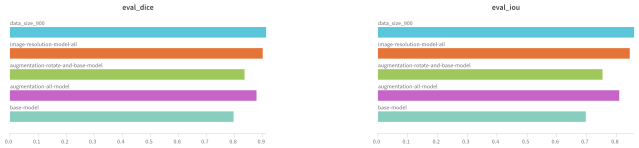


Fig. 8: Graph depicting the Dice during the training process of each model with increased training data.

The general trend seem to imply that more data will lead to a greater performance. Both dice and IoU are increasing with the data amount.



(a) Dice (b) IoU

Fig. 9: Graphs depicting the evaluation Dice and IoU of the models with increased training data.

IV. Discussion

The discussion section will be split into three parts, first two parts cover what were found in experiment I and II, then with the context from I and II, experiment III are covered in greater detail.

In experiment I, the results were generally what was expected, but when looking closer there are some unexpected finds. Going into this experiment the expectation was that lowering the resolution well below the smallest image in the dataset would lead to lost data and a generally weaker models. However by looking at the resolution experiment models scores It becomes apparent that while yes there are trends towards this, there are also significant outliers, best demonstrated by the precision scores shown in figure 10. Specifically the Base(576x576), 256x256 and 2048x2048 model. The 256x256 model significantly beat the trends set by and out performed every other model here. While 2048x2048, the largest of them saw much weaker results comparatively. The same can be said for the base model, however, it also ran for considerably fewer epochs and would likely catch up if trained longer. This is rather interesting due to the fact that a resolution of 256 is half the resolution of the lowest resolution image in the base dataset. If plotted on a graph with a logarithmic X representing resolution as done in Figure 11. The data seem to indicate diminishing returns or even a downwards facing asymptotic graph with a peak somewhere between a resolution of 256x256 and 567x567. This suggest there likely is an ideal resolution for this dataset situated between 256 and 576. That would imply the best approach would in fact be to lower the resolution of the dataset as a whole for training. This finding can be backed up by a study from 2020 that investigated the effect of image resolution on deep learning radiography models. One of their findings were that the best resolutions to work with were between 256×256 and 448×448 pixels for binary decision networks [7]. While binary decision and binary segmentation are separate fields, the findings are non the less interesting and seem to align with the observations made.

The Experiment II results were less interesting, but again with some unexpected observations. The models performed about the same, but with a slight hit to just about every metric compared to the base model. Which in itself is a bit interesting as one would expect next to no lost data on the rotation and flip models. but these two consistently performed worse than even the crop model which one would assume be the worse of the three as that model looses data during augmentation by design.

The Flip augmentation model specifically performed markedly worse than both the base- and other augmentation models. Investigating this further reveal that the flip model's train, and validation IoU and Dice scores start to diverge considerably after epoch 5. This might suggest that this specific model starts to overfit much earlier than the rest of the augmentation models. see figure 13. This find is rather unexpected and should probably not be trusted without further investigation. Theoretically the flipped dataset should perform exactly the same as the base model as it should contain all the same data points only mirrored. It might be caused by mistake or miss understanding of how the augmentation should be implemented. Or it could be that the base data has some degree of inherent asymmetry. We as a team do not have the expertise to evaluate if this is or ins't the case for gastrointestinal imagery. But a clear example of inherent asymmetry in metical imagery is the human heart. The heart is typically located on the left side of the chest and have a very asymmetrical design. Applying horizontal flip to such images can introduce unrealistic variations, potentially confusing the model during training.

Regardless, the results are more than comparable to the base model across all augmentations and most resolutions on both validation and evaluation. This is a clear indication that both resolution and other augmentation methods can be used to increase the models training data size, and by extension its generalization. However when saying that it is important to note that the dataset being used originates from a single source, therefor there is a risk of contamination by artifacts introduced during its creation. For instance, minor scratches on the camera lens or uniform encoding processes across all images could inadvertently bias the model. Models can also learn that specific things have been taken with a specific types of camera, and thereby cheat by learning the camera artifacts instead of the shown items. This is illustrated well by Philip T. Jackson in [8]. Consequently, when generalization is mentioned here it applies strictly to unseen data within the Kvasir-Instrument dataset. With that, in experiment III the expected results were that with increased training data, the

model improve and generalize better. However, it will also lead to greater resource requirements and time investments.

The findings of "The Shape of the Learning Curves: a Review" [9], show that training with a larger dataset will lead to improved performance of the model, but the result will give diminishing returns as the dataset gets larger. Furthermore, the results from "Deep learning scaling is predictable, empirically." [10] show that models trained on larger dataset are able to utilize the data available to it much better, resulting is a much deeper understanding.

The results from experiment III shows that using the entire training dataset improved performance across the board. In figure 6 the models training on fewer data points spent a longer time before starting to learn at all. This is a good indicator that the model simply didn't have enough data to understand the data as there likely wasn't enough correlations to cling onto. The models performance is shown to rapidly increase with more data. However it also shows a trend towards diminishing returns with training data increases. This is illustrated in figure 12.

This is also fairly well visualized within figure 9. Here the actual predictions vs the ground truth is visualized, The larger models all seem to know roughly where surgical tool is and to some degree its outline, however they have a hard time understanding shape of it and exactly where the edge of it is. The diminishing returns can likely to some degree can be attributed to the fact that the models with larger training data seem to fill out the predictions more.

Going beyond the base model's data size also increased the performance considerably. However there are some interesting observations to make, especially in regards to the model containing just different image resolution sets. By training metrics it easily out performed every other model tested with a staggering 0.96 dice score achieved, however, looking at the models evaluation scores it scores considerably worse indicating the model started over fitting. This is displayed the models training vs validation loss score graph shown in figure 14. This is likely a result of the resolution images being to similar to each other, especially the higher resolution images. When an image is downscaled from 1024 pixels to 256 pixels, it loses a significant amount of detail, effectively transforming it into a new data point with altered characteristics. In contrast, upscaling an image, regardless of its original resolution, does not add any new information beyond what was present in the source image. This results in duplication of images at higher resolutions, particularly in the case of 2048x, where all images are effectively duplicates. This increases the likelihood of the model memorizing specific images rather than learning general patterns or trends, in other words this easily leads to over fitting. The same trend can be seen when investigating the model trained on 900% of the data. as shown in figure 15.

These overfitting issues could potentially be mitigated to some degree by combining the different augmentation methods used. For example, one could apply a resolution change and rotate the same data. thereby creating new data that might not share any characteristics with the source images.

V. Conclusion

This study explored the effects of data size, image resolution, and augmentation techniques on the performance of a U-Net model for binary segmentation using the Kvasir-Instrument dataset. Experiment I demonstrated that while higher resolutions generally improve performance, excessive resolution scaling may lead to overfitting and diminishing returns. Experiment II showed that basic augmentation methods, though effective in expanding the dataset, provided limited improvement when used in isolation. Finally, Experiment III highlighted the importance of adequate training data in achieving higher IoU and Dice scores, though the performance gains diminish with excessive data scaling. The results underline the necessity of balancing dataset size and quality while employing diverse augmentation techniques to enhance generalization without risking overfitting or excessive resource use.

VI. Appendix

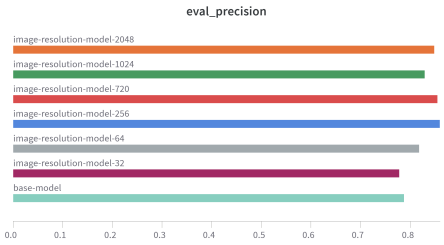


Fig. 10: The recall score each resolution based model got after evaluation on the test set.

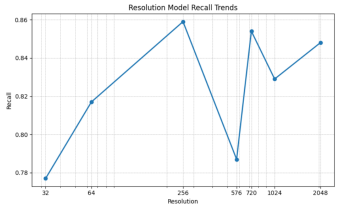


Fig. 11: Displaying the precision of the resolution models with a logarithmic X axis representing resolution.

Fig. 14: Training vs Validation Loss for model containing all resolution data

Fig. 15: Training vs Validation Loss for the model training on every augmentation set, as well as the 256x, base set and 720x set.

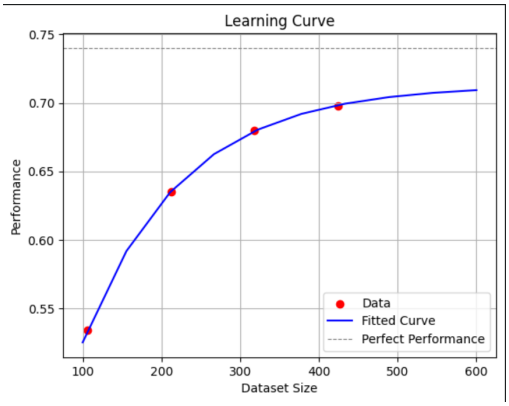


Fig. 12: Learning Curve showing diminishing results

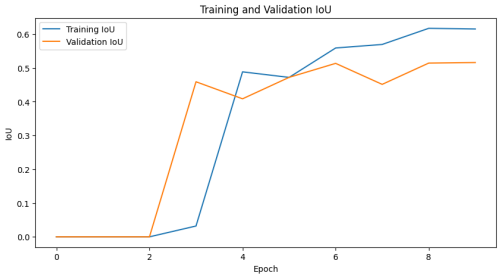


Fig. 13: Graph depicting the train and validation graphs for the Flip augmentation model

References

[1] D. Jha, S. Ali, K. Emanuelsen, S. A. Hicks, V. Thambawita, E. Garcia-Ceja, M. A. Riegler, T. de Lange, P. T. Schmidt, H. D. Johansen, D. Johansen, and P. Halvorsen, "Kvasir-instrument: Diagnostic and therapeutic tool segmentation dataset in gastrointestinal endoscopy," in *International Conference on Multimedia Modeling*. Cham: Springer, 2021, pp. 218–229.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998. [Online]. Available: https://ieeexplore.ieee.org/document/726791

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.

[5] PyTorch, "Bcewithlogitsloss documentation," https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html, 2023.

[6] ——, "Reducelronplateau documentation," https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html, 2023.

[7] C. F. Sabottke and B. M. Spieler, "The effect of image resolution on deep learning in radiography," *Radiology: Artificial Intelligence*, 2020. [Online]. Available: https://doi.org/10.1148/ryai.2019190015

[8] R. Geirhos, A. A. Müller, M. Bethge, F. A. Wichmann, and W. Brendel, "Camera bias in a fine grained classification task," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2021, pp. 1810–1820. [Online]. Available: https://ieeexplore.ieee.org/document/9534097

[9] T. Viering and M. Loog, "The shape of learning curves: a review," https://arxiv.org/abs/2103.10948, 2021.

[10] H. Joel, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. M. A. Patwary, Y. Yang, and Y. Zhou, "Deep learning scaling is predictable, empirically," https://arxiv.org/abs/1712.00409, 2017.