A Closer Look at Data Augmentation and Preprocessing for Image Segmentation with Kernel Attention U-Nets

Shawn Carere*
Department of Medical Biophysics
University of Toronto
Ontario, Canada
shawn.carere@mail.utoronto.ca

Sayan Nag*
Department of Medical Biophysics
University of Toronto
Ontario, Canada
sayan.nag@mail.utoronto.ca

Abstract

U-Net and its variants are widely used in medical image segmentation tasks. In this paper, we propose a novel non-linear *kernel* based attention U-Net which gives 2.5% and 1.5% IoU improvements at the cost of just 0.5% increase in model parameters with respect to Attention U-Net. With such a kernel based multiplication between the attention matrices and input feature maps a strong locality constraint is enforced. We also demonstrate the often overlooked impacts of data-processing and augmentations on model training and evaluation. We show that small changes to preprocessing pipelines for validation and testing data can result in up to a 7% difference in IoU during model evaluation, hence highlighting the need for standardization. Furthermore we explore different implementations of data augmentation and show how they can effect model training. The source code used in the study can be found **here**.

*denotes equal contribution

Preprint. Under review.

1 Introduction

Medical image segmentation refers to the identification and extraction of regions of interest from medical image modalities. These modalities include Computed Tomography scans, Ultrasound images, Fluoroscopy, Magnetic Resonance Imaging, Positron Emission Tomography scans, etc. Identification of these regions of interest is important for clinicians to study specific organs and make accurate diagnosis. Segmentations were initially done by pathologists and radiologists. However, such manual labeling and annotation tasks can not only be tedious and time consuming but also prone to human errors. Human introduced errors might lead to unreliable and inefficient outcomes which can hinder crucial decision making tasks. Therefore, automated image segmentation has become a sacrosanct task in the recent years.

In the earlier days for medical image segmentation scientists often relied on basic image processing techniques such as edge detection, template matching methods, active contours, region-growing approaches. However, with the rise of Deep Learning (DL), scientific community has shifted to using DL frameworks for medical image segmentation. A major shift happened especially after the introduction of a fully-convolutional encoder-decoder based DL framework called U-Net (1) which became state-of-the-art in the Deep Learning based Medical Image Segmentation community. Further advancements in the field followed the footsteps of this well-renowned U-Net which led to the genesis of Attention U-Net (2), U-Net++ (3), DoubleU-Net (4), ResU-Net (5) and so on. Very recently, researchers have leveraged self-attention based Transformer framework introducing TransUnet (6) and Swin-Unet (7). To use such deep architectures, huge volumes of data is absolutely necessary. However, sample sizes of medical image datasets are very small when compared to those of natural images. Therefore data augmentation becomes one of the most handy and crucial tools in this regard.

Data preprocessing and augmentation techniques are prevalent in a variety of supervised learning tasks. Of particular interest are computer vision tasks, which are often heavily reliant on the amount of training data and thus utilize a wide variety of image augmentation techniques to prevent overfitting of deep and complex models. Although these augmentation techniques have been shown to be very effective in improving model generalization (8; 9), the choice of which augmentations to use and the magnitudes by which to apply them is often empirically (but loosely) justified. Furthermore, small differences in the implementation of data preprocessing and augmentation can often have disproportionately large impacts on model generalization.

This is especially true for supervised image segmentation tasks, where models predict a class for each individual pixel of the input image. Image segmentation models often take the 2 dimensional images as input, and predict 2 dimensional 'mask images' of the same shape as an output. The mask images contain the predicted class, often represented by an integer, for each pixel. Hence many modification made to the input images through data preprocessing and data augmentation must also be made to the ground truth mask images. This makes image segmentation models particularly sensitive to changes in data preprocessing and augmentation, consequentially reducing reproducibility within the field. In this paper we decide to focus on a subset of image segmentation, nuclei detection. Nuclei detection is a binary segmentation task for microscopic images of various types of cells. Pixels within the image are classified as either belonging to a cell's nuclei or not. Identifying the cell's nuclei is often the starting point for most analyses of microscopic images. This information allows researchers and health experts to identify all the individual cells in a sample and make inferences about how those cells are responding to various treatments and or experiments. Doing nuclei detection manually is often a repetitive and time consuming task. Automating this task is of great value to researchers and health experts as it allows them to focus on the underlying biological processes at work. From a computational perspective, nuclei segmentation is a task particularly well suited for intense data preprocessing and augmentation. Microscopic images vary greatly due not only to the large array of different cell types, shapes and sizes, but also due to the many differences between microscopic imaging procedures. These include different types of histology stains, imaging equipment and modalities, magnification, illumination etc. To demonstrate this, we show some example microscopic images in 1. The wide range of differences between samples allows for a wide variety of image augmentations and preprocessing approaches without compromising the integrity of the sample.

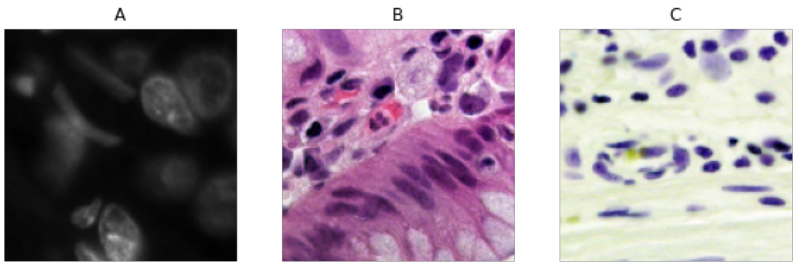


Figure 1: Examples of microscopic images. **A:** Fluorescent microscopy of cultured cells, primarily black and white **B:** Brightfield microscopy of a stained tissue sample, primarily pink and purple **C:** Brightfield microscopy of a stained tissue sample, primarily purple

In this paper we compare a novel Attention U-Net based architecture to existing methods as well as demonstrate the importance of data preprocessing and augmentation to model generalizability.

Contributions: In light of the aforementioned background, we state the main contributions of our paper:

1. We propose a novel *Kernel* Attention U-Net which has 2.5% and 1.5% IoU gains over a baseline Attention U-Net model on validation and test sets respectively without any data augmentation.
2. We also demonstrate the importance of data preprocessing and augmentations in the final outcomes. Showing, for example, how the method by which images are resized to common dimensions affects model performance.

2 Methods

2.1 U-Net

U-Net is an end-to-end architecture which is based on fully convolutional neural networks (1). It comprises of a contraction (encoding) path for capturing context information and a symmetric expansion (decoding) path for enabling precise localization. The contraction path has repeated 3×3 convolution block followed by a down-sampling block. The expansion path is composed of up-sampling followed by convolution (up-convolution) blocks. Skip-connections are used to connect encoder with decoder at all the levels for transmitting information at various spatial resolutions.

2.2 Attention Gates

Prior to the introduction of Attention Gates in U-Net, segmentation methods used object localisation modules for auxiliary tasks such as localisation besides segmentation modules. By integrating Attention Gates to an encoder-decoder based fully convolutional model such as U-Net, it has been demonstrated that similar performance can be achieved avoiding auxiliary models and thus decreasing overall model parameters (2). Attention gates have also been experimentally found to avoid irrelevant background areas therefore eliminating the need to crop the region-of-interests from the input images. Attention coefficients (aka Attention scores) act as masks identifying relevant regions and preserving those activations. These coefficients are being multiplied to the input feature maps (output of a convolutional layer in the encoding pathway).

2.3 Loss Function and Evaluation Metric

For training purposes we have used a hybrid loss function composing a combination of Dice Loss (10) and Binary Cross Entropy (BCE) loss. For reconstruction problems, BCE loss has been widely used and has been observed to perform superior to standard MSE loss. This is because for reconstruction problems, the activation function of the last layer is sigmoid, which essentially leads to loss saturation which hinders the learning process of stochastic gradient-based algorithms. Unlike MSE, BCE has a logarithm function which is responsible for undoing the exponential in sigmoid leading to better performance (11). On the other hand, Dice is a well-known metric to compare segmented outputs. In our work we have seen that combining these two losses together gives better performances as compared to their individual counterparts. Hence, our proposed loss function between ground-truth label (y) and predicted label ($\bar{y}$) is given as:

$$L(y, \bar{y}) = Dice(y, \bar{y}) + \lambda \; BCE(y, \bar{y}) \tag{1}$$

We have considered $\lambda = 1$ for all the experiments.

For evaluation purposes, we have compute the Intersection Over Union (IoU) as a metric to measure segmentation performance of our models. IoU is computed between the true segmentation labels and the predicted segmentation labels, mathematically given as:

$$IoU(y, \bar{y}) = \frac{y \cap \bar{y}}{y \cup \bar{y}} \tag{2}$$

2.4 Kernel Attention U-Net

As proposed by the authors in (2), in an Attention U-Net, attention score (α) is generated from the previous layer via an attention gate, and it gets multiplied with the skip connected output (x) from the encoding pathway in the same layer. This attention gate also comprises of a gating vector g which is used from the layer below containing contextual information to crop lower-level feature responses. Here, α has a dimension of $1 \times H \times W$ which gets multiplied with x which is of dimension $C \times H \times W$. Therefore, the same attention scores matrix (mask) is being multiplied (element-wise) with all the channels of x. We argue that this results in an inefficient masking scheme which can be further improved by introducing more channels in the attention masks. To this end, we propose

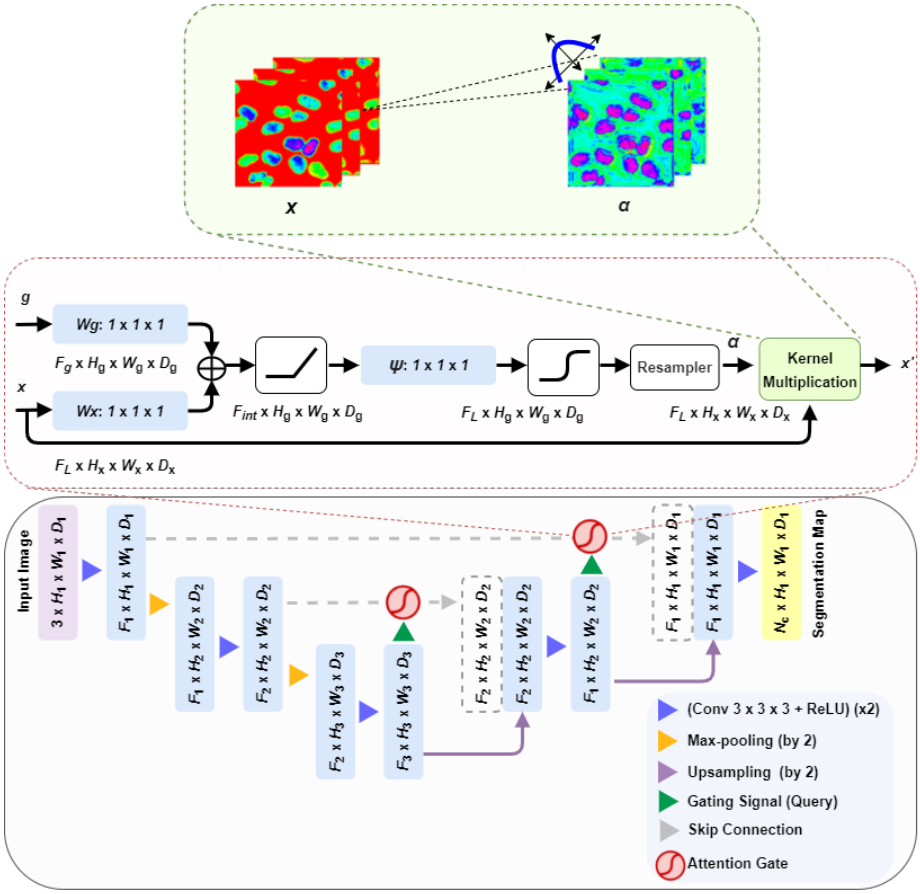


Figure 2: Block diagram of our proposed Kernel Attention U-Net. Inputs are passed through convolution and downsampling blocks in the encoding pathway which are then upsampled and convolved in the decoding pathway. Skip connections via attention gates connect encoding and decoding pathway. Input features x are scaled by the attention coefficient matrices α using a kernel multiplication. For simplicity we have chosen a non-linear decomposable cosine kernel (shown in blue) enforcing locality constraints.

multi-channel attention scores where the channel dimension is same as that of x. Further, we utilized *kernel based multiplication* between α and x. For simplicity purposes we have chosen a simple non-linear cosine kernel. With addition of such a decomposable non-linear cos-based re-weighting mechanism recency bias can be introduced. As a consequence, a strong locality is enforced weighting the neighbouring channels more with respect to the far-away ones. The aforementioned mechanism has been ilustrated in the Figure 2.

The kernel operation between x and α with cosine re-weighting is defined as:

$$k(x_i, \alpha_j) = x_i \alpha_j cos\left(\pi \frac{i-j}{2C}\right) \tag{3}$$

Using Ptolemy's theorem and decomposing the above expression further leads to:

$$k(x_i, \alpha_j) = \left(x_i cos\left(\pi \frac{i}{C}\right)\right)\left(\alpha_j cos\left(\pi \frac{j}{C}\right)\right) + \left(x_i sin\left(\pi \frac{i}{C}\right)\right)\left(\alpha_j sin\left(\pi \frac{j}{C}\right)\right) \quad (4)$$

where $i, j = 1, ..., C$, C is the number of channels.

Since, in this work we are predominantly using cosine as the only non-linear kernel we will alternatively call our model as Cosine Attention U-Net.

2.5 Data Augmentation and Preprocessing

For the purposes of evaluating the effects of data augmentation and preprocessing, we use a baseline U-Net architecture with the same architecture shown in 2 except without the attention gates.

2.5.1 Dataset

In this paper we use the Kaggle 2018 Data Science Bowl dataset from the Broad BioImage Benchmark collection (12). The dataset contains a diverse collection of microscopic images collectively containing over 10 thousand nuclei. The nuclei in the images are derived from various species including humans, mice, pigs and flies. The images were acquired under a variety of conditions including fluorescent and histology stains, several magnifications and varying quality of illumination. Additionally the nuclei appear in a variety of different contexts including tissues, mono-layered cultures, embryos, genotoxic stress, cell division and differentiation.

The dataset is separated into three sets, stage-1-train, stage-1-test and stage-2-test. In our experiments we use stage-1-train as our training set, stage-1-test as our validation set and stage-2-test as our test set. We will refer to these sets as train, validation and test respectively throughout the rest of this paper. It is important to note that the test set was never used during model training or hyperparameter tuning throughout our experiments.

2.5.2 Preprocessing

It is common practice in the field of computer vision to modify images in a variety of ways to make model training faster and more effective. These modifications include normalization, resizing, cropping etc. and are collectively referred to as preprocessing. Preprocessing techniques are a

powerful tool for computer vision, and in most instances can be used on validation and test sets without impacting the integrity of results. It is with segmentation tasks however, where their use can become problematic. For many preprocessing modifications, the same modifications made to the input images must also be made to the output masks. Therefore despite using the same dataset, various works might be testing their models with different ground truths, resulting in variation when trying to compare works and reproduce results. Furthermore, differences in ground truth masks can impact hyperparameter tuning as well as model training dynamics.

In this paper we focus on one of the most common preprocessing steps in computer vision, matching image dimensions. This preprocessing step is so common in fact, that dataset creators often take it upon themselves to match image dimensions before making their datasets publicly available. Matching image dimensions is the process by which images of different sizes, aspect ratios and resolutions are transformed in some way to all be the same size. Matching image dimensions has a variety of benefits. It makes model implementation more convenient as well as speeds up the forward and backwards passes by allowing mathematical operations to be broadcast onto the entire batch. However, many neural network layers (eg. Fully Connected) are specific to the size of the inputs, and therefore models that use these layers must be retrained in order to work at different input image resolutions. U-Nets are one of the few architectures that do not use layers dependent on input shape, and therefore are agnostic to the input image size. That being said, matching image dimensions is still useful for U-Nets due to mathematical broadcasting. Although this decreases computational time, it is usually only useful for model training. Yet it is still common practice to match the image dimensions of the validation and test sets as well, even though this is technically unnecessary for U-Net based architectures. In practice existing tools make the implementation of size agnostic U-Nets non-trivial which we discuss in more detail in our conclusion.

Some of the most common approaches for matching image dimensions include cropping and resizing the images. In this paper we explore the effects of these modifications on both the training and inference of nuclei segmentation models. Cropping removes information from the ground truth masks. Therefore, unless the exact same portions of each mask are cropped, then the ground truths across various works will not be identical. This leads to inconsistencies despite the original dataset being the same. Image resizing can also have various unintended effects. If the aspect ratio of the original image does not match the aspect ratio of the desired dimensions, then both the image and it's mask will be distorted. Not only does this lead to additional inconsistencies when testing but,

depending on the task, will either corrupt the training data or act as a form of data augmentation. Furthermore, resizing an image to new dimensions often requires the use of interpolation. Not only does this introduce artificial information into what is supposed to be the 'ground truth' masks, but often results in certain pixels within the mask having a value that does not correspond to one of the output classes. For example, in nuclei segmentation, a mask pixel has a value of 1 if it belongs to a nuclei, and 0 if it does not, hence the masks are referred to as being binary. After resizing the mask, some of the new pixels may have non binary values due to the interpolation process. Some researchers might recognize this, and threshold the masks before training. Alternatively one might choose to use the non-binary masks and threshold them only when necessary (eg. calculating IoU). In the worst case scenario, the masks might be truncated, resulting in all non-binary values being set to zero. Although these differences seem small and relevant only near segmentation boundaries, we show that they do indeed have a significant effect.

In this paper, we match the image dimensions using 3 different approaches, resulting in 3 distinct datasets or processing pipelines. We will reference these preprocessed datasets by their name throughout the paper, so we define them here for clarity. For all 3 datasets, the final spatial dimensions are 128x128 and the input images are normalized by dividing each pixel by 255. For the resampled and undistorted datasets, an interpolation order of 1 (linear interpolation) was used when resampling or rescaling. Lastly note that these processing pipelines were applied to our training, validation and test sets, each of which were only ever used for their respective purposes. For example, when we say we trained with the resampled dataset, this implies the training subset of the resampled data. The original testing and validation data are never used for model training, regardless of how they are preprocessed. Furthermore the original testing data was only used to obtain the final results from our models when writing this paper.

Resampled: For this dataset, the original samples were simply resampled to the desired resolution. As previously discussed, this results in distortion for samples that are not perfectly square. Additionally the masks contain non-binary values due to the interpolation of the resampling process. However, the entire image is preserved and no information is cropped out.

Undistorted: For this dataset, the original samples were resized to the desired dimension without distorting their aspect-ratio. This was done by cropping the images to the desired aspect-ratio before rescaling them to the desired resolution. Samples were cropped into multiple new samples when

possible in order to reduce the effects of cropping and preserve as much of the original image as possible. The masks contain non-binary values however due to the interpolation within the resizing process.

Downsize-Cropped: For this dataset, the original samples were downsampled to the desired dimensions. This was done by first cropping the images dimensions to the nearest multiple of the desired dimensions, and then downsampling by just omitting every nth row/column of the image (where n is the factor to downsample by). Not only did this preserved the aspect-ratio of the samples, but it converted them to the desired size without introducing new information via interpolation. Samples were cropped into multiple new samples when possible in order to reduce the effects of cropping. However due to the stricter dimensionality requirements, the downsize-cropped dataset is removes more information due to cropping than the undistorted dataset.

2.5.3 Rotation

Random rotations are a common form of data augmentation used for model training in many computer vision tasks. There exists various different ways to implement data augmentations which can have significant effects on model training and generalizability. Below we define several distinctions between various random rotation implementations. In this paper we explore the differences between batchwise and samplewise rotation augmentations, as well as the effects of the maximum magnitude of rotation on model training and generalizability. For all experiments in this paper we use a continuous range when selecting rotation angles. Additionally we implement all augmentations in an online manner as this is what is most common in the field of image segmentation.

Discrete vs Continuous: When applying a random rotation transformation to a sample, a random angle by which to rotate the image can be selected from either a continuous range (ie. $a_c \in (-r, r)$ where r is some maximum rotation magnitude), or a discrete set of values (ie. $a_d \in [0, 2, 4, 6, 8, 10]$).

Online vs Offline: Data augmentation can typically be implemented in one of two ways. Offline data augmentation refers to applying the transformations to the training set before training begins in order to artificially increase the number samples in your dataset. Both the unaugmented and augmented samples become part of the new augmented training set. Online data augmentation refers to randomly augmenting the batch of original data at the beginning of each train step. The augmented batch, which has the same number of samples as the original batch, is used only for one train step

and then thrown away. Since the augmentation is random, this results in samples being augmented differently each time they are seen by the model.

Batchwise vs Samplewise: There are many commonly used deep learning frameworks that allow for convenient implementation of online image augmentations. The current most popular frameworks are PyTorch and Tensorflow. Interestingly, each of these frameworks implements online data augmentations differently. If this is overlooked, it can result in inconsistencies between implementations of the same work using different deep learning frameworks. PyTorch uses batchwise data augmentations, which means that all the samples within a batch are augmented in the same way. For example, for a batchwise rotation augmentation, PyTorch would select a single random angle at the beginning of each train step, and rotate every sample in the input batch by that same angle. Tensorflow, however, uses samplewise data augmentations, which means that each sample is augmented individually. For a samplewise rotation augmentation, Tensorflow would select a different random angle for each sample within the input batch. Therefore every sample within the batch would be rotated by a different amount.

3 Experiments

All models throughout our experiments were trained using stochastic gradient descent (SGD) with nesterov momentum of 0.9 and a learning rate of 0.001. Unless stated otherwise, all models had a learning rate step down from 0.001 to 0.0005 at the end of the 10th epoch.

3.1 Comparisons with baseline models

In this section we compare the differences between a baseline U-Net, a state of the art U-Net with attention, and our novel U-Net with cosine attention. In order to focus on the improvements due to model architecture, we do not use any data augmentation for the results shown in this section. The used the undistorted processed scheme for training, and the downsized cropped processing scheme for validation and testing. Furthermore the undistorted training masks were thresholded by 0.5 before training in order to remove non-binary values introduced as a result of interpolation.

Table 1 contains the results where each model has been run 3 times with 3 different seeds and we have reported the mean and standard deviation over these 3 runs. It can be noticed that the mean IoU is the highest for our proposed Cosine Attention U-Net on both Validation and Test

datasets as compared to U-Net and Attention U-Net. Further, the standard deviation is also less for Cosine Attention U-Net as compared to those of others. This suggests that inclusion of multi-channel attention masks along with a locality preserving kernel (cosine) based multiplication indeed helped in improving the overall model performance. It is important to note that we have not used any learning rate step-down for either Attention U-Net or Cosine Attention U-Net since it did not result any improvement in performance.

Model Details				
Name	**Params (M)**	**GMacs**	**Validation IoU (%)**	**Test IoU (%)**
U-Net	∼34.52	∼16.37	75.66 ± 1.67	72.85 ± 2.44
Attention-U-Net	∼34.88	∼16.66	77.59 ± 0.78	77.46 ± 0.81
Cosine Attention-U-Net	∼35.06	∼16.80	$\mathbf{80.05 \pm 0.31}$	$\mathbf{78.68 \pm 0.33}$

Table 1: Baseline results (w/o data augmentations) for nuclei segmentation task using Data Science Bowl 2018 dataset.

Figure 3 shows the predicted segmentation output maps from the Attention U-Net and proposed Cosine Attention U-Net models. The red squares in each of the subplots highlights some interesting regions of the predicted maps (masks). It can be visually noticed that the output masks using Cosine Attention U-Net are more similar to the ground truth masks. This observation is also corroborated by the IoU values which have been computed and reported in the Table 2. Here, the nuclei images have been numbered from 1-4 in order from top to bottom as in the Figure 3.

Image Number	**Attention U-Net**	**Cosine Attention U-Net**
1	83.52	**84.04**
2	91.25	**91.32**
3	88.78	**88.92**
4	83.66	**87.29**

Table 2: IoU's (%) computed for 4 images (in order from top to bottom) shown in 3 using Attention U-Net and Cosine Attention U-Net.

Figure 4 shows the attention coefficient matrices (for a random channel and for the top-most level) across various training epochs along with the original training image. It can be observed that the proposed model gradually learns to focus on the cells with increasing epochs which is noticed from the intensity of the pink region.

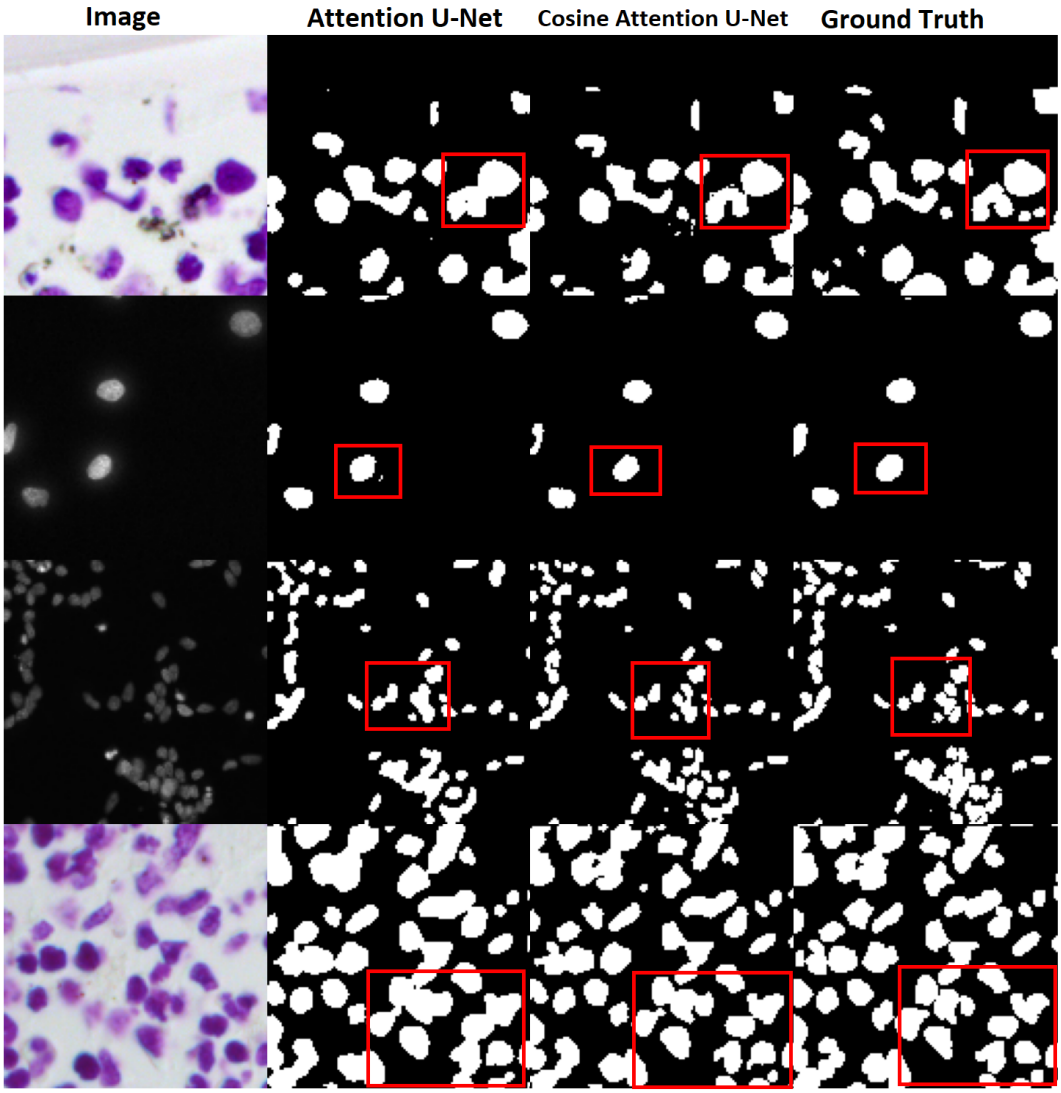


Figure 3: Segmented output masks (maps) along with original images and ground truth masks (maps).

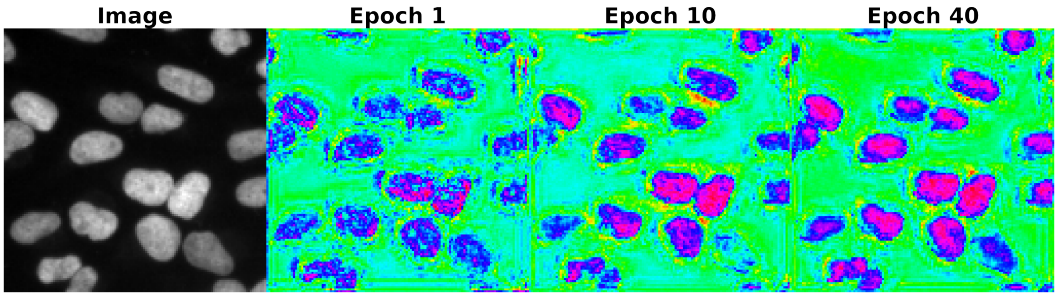


Figure 4: Attention coefficient matrices.

3.2 Data Augmentation and Preprocessing

3.2.1 Preprocessing

To test the effects of different preprocessing pipelines on model training and inference, we train the same baseline U-Net with the various preprocessed datasets defined in Section 2.5.2. Note that for the resampled and undistorted training sets, the training masks were thresholded by 0.5 before training in order to make them binary again.

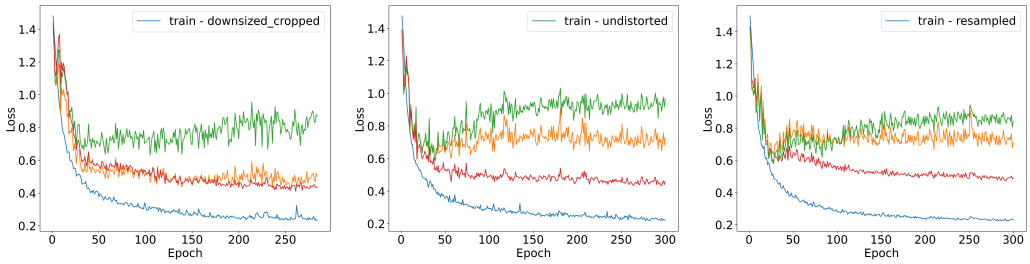
	Training		
Testing	resampled	undistorted	downsized cropped
resampled	82.44	80.75	**83.22**
undistorted	78.72	77.65	**79.52**
downsized cropped	75.53	74.54	**77.27**

Table 3: Test IoU's (%) of baseline U-Nets trained and validated using different types of preprocessing as discussed in Section 2.5.2

One can see from Table 3 that regardless of what training set is used, the models reported performance varies greatly depending on how the test set is preprocessed. Depending on the test set, we observe a difference in IoU of up to 7% despite the predictions being exactly the same. Testing with the resampled data consistently gives the most favourable results while testing with the downsized cropped data consistently gives the least favourable results. This is the case regardless of how the training data is preprocessed. This is problematic for the field of image segmentation as there is no agreed upon standard on what alterations to the test masks are permitted. Worse even, it makes it impossible to compare results between works without ensuring that every aspect of the processing pipeline is exactly the same.

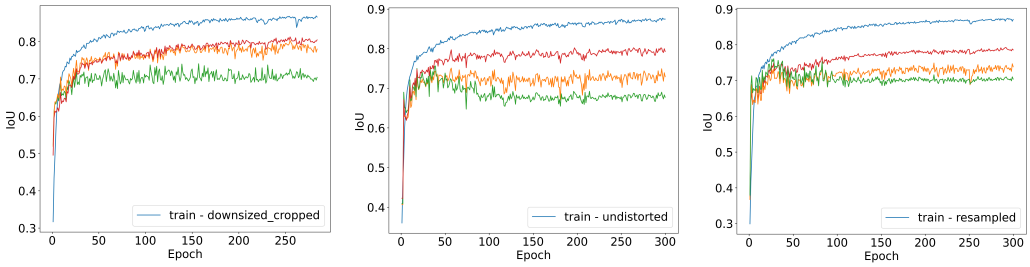
Furthermore we observe that using the downsized cropped training set achieves the best performance on all three variations of the test set. This was a surprise as we expected the distortions in the resampled training set to act as a form of data augmentation. Additionally the model trained with the undistorted training set achieved the worst performance despite only being slightly different from the downsized cropped dataset. What the resampled and undistorted datasets have in common is that they both use interpolation when resampling or rescaling the images. This seems to indicate that the information introduced by this interpolation makes it more difficult for the model to learn. We elaborate on this further when discussing our observations of Table 4. Additionally, the resampled training set achieves better performance than the undistorted dataset. Since the main difference

between these sets is the presence of distortions to the aspect ratios of the images, then it is likely that these distortions were indeed acting as a form of data augmentation, and improving model performance.



(a) downsized cropped (b) undistorted (c) resampled

Figure 5: Loss curves of U-Net trained and validated with different preprocessed datasets. The blue curve is training, the red curve is resampled validation dataset, the orange curve is undistorted validation dataset and the green curve is downsized cropped validation dataset. a) Trained with the downsized cropped training set b) Trained with undistorted training set c) Trained with resampled training set



(a) downsized cropped (b) undistorted (c) resampled

Figure 6: IoU curves of U-Net trained and validated with different preprocessed datasets. The blue curve is training, the red curve is resampled validation dataset, the orange curve is undistorted validation dataset and the green curve is downsized cropped validation dataset. a) Trained with the downsized cropped training set b) Trained with undistorted training set c) Trained with resampled training set

In order to qualitatively assess the effects of the differently preprocessed datasets, we validated our models using all three preprocessing schemes to observe how the calculated losses and IoU's change throughout training. These results can be seen in Figures 5 and 6. One can see that all three validation curves follow one another at the beginning of training and then progressively diverge from one another, with the downsized cropped validation set overfitting the most, and the resampled validation set seemingly not overfitting at all. The undistorted validation also seems to show overfitting, but not as strongly as the downsized cropped validation data. The one exception is in the model that uses the downsized cropped training data, which seemingly prevents overfitting on the undistorted validation data.

Testing	Training			
	resampled		undistorted	
	binary	non-binary	binary	non-binary
resampled	82.44	**82.54**	80.75	**82.39**
undistorted	78.72	**78.87**	77.65	**79.07**
downsized cropped	75.53	**76.16**	74.54	**76.20**

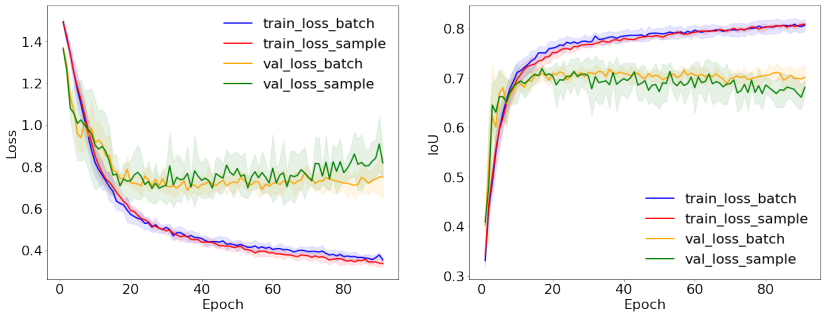
Table 4: Test IoU's (%) of baseline U-Net models for both thresholded binary training masks and non-binary masks. As discussed in section 2.5.2, only the resampled and undistorted preprocessing schemes introduce non-binary values to the masks

In the previous experiments we account for the non-binary masks generated by the resampled and undistorted preprocessing schemes by thresholding them before training. However in order to explore the effects of training with non-binary masks, we purposefully omit this thresholding and observe the differences. One can see from Table 4 that thresholding the masks before training consistently results in a lower test IoU. This indicates that there is some advantage to introducing uncertainty in our training labels. This supports our observations of Table 3 which showed that the training sets that use interpolation when processing the masks result in worse performance. This is likely because the non-binary pixels generated by the interpolation process are significantly less informative. If we assume the values of these interpolated pixels to be entirely noise, then by thresholding them, we are weighting them more heavily in the loss function and hence introducing more noise into the gradient calculated after each batch. The model is penalized for being uncertain of their class and predicting them incorrectly, even though their class is uncertain in the ground truth. We also note that the difference in performance between the binary and non-binary masks is lessened for the resampled training set. This further reinforces the idea that the distortions in this dataset are acting as a form of data augmentation, and making up in part for the negative effects of interpolation. Our results throughout this section indicate that the use of interpolation in preprocessing pipelines can significantly harm model performance, whereas unintentional distortions to the aspect ratio can act as a form of data augmentation and improve model performance. It is important to be aware and forthright with how one implements data preprocessing and augmentation so as not to confound the results where other aspects of model training and design are the independent variable.

3.2.2 Rotation

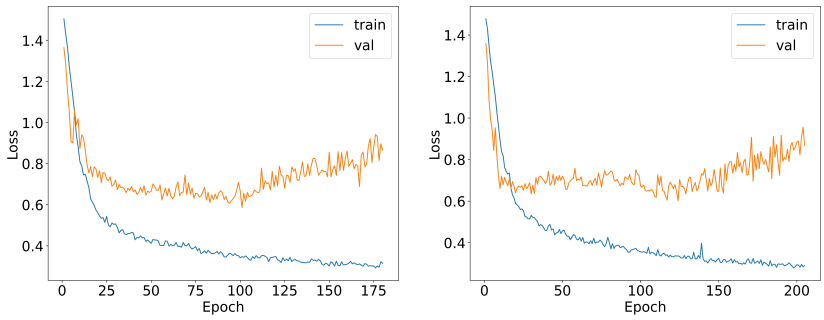
In this section we explore how different implementations of the same data augmentation can effect model training and generalization. We focus on the differences between batchwise and samplewise

augmentations using a random rotation during training of our baseline U-Net architecture. It is worth noting that we introduce a second learning rate step-down at epoch 20 from 0.0005 to 0.00025 when comparing batchwise to samplewise random rotations. One can see from Figure 7 that the model trained samplewise random rotations has more variance in throughout training than the model trained with batchwise random rotations. This is because the batchwise implementation takes longer to reach the overfitting state as seen in Figure 8. Before overfitting there is much less noise in the validation loss and IoU, once the model becomes overfit, the validation loss becomes nosier. Since the model with samplewise random rotations converges faster, it spends more time in the overfit state and hence has more variance in loss and IoU throughout training.



(a) Loss (b) IoU

Figure 7: Average loss and IoU of baseline U-Nets trained with an online continuous random rotation for various maximum angle magnitudes



(a) Batchwise (b) Samplewise

Figure 8: Loss of baseline U-Nets trained with an online continuous random rotation with a maximum magnitude of 30 degrees. a) Batchwise implementation b) Samplewise implementation

	Validation IoU (%)	**Test IoU** (%)
Batchwise	**75.02**	**80.71**
Samplewise	74.62	79.32

Table 5: IoU comparison between batchwise and samplewise implementation of a online continuous random rotation with a maximum magnitude of 30 degrees

One can see from Table 5 that a batchwise implementation for random rotations seems to work slightly better. However due to the large amount of noise in our validation metrics, and the small size of our testing dataset, further exploration must be done to confirm these findings.

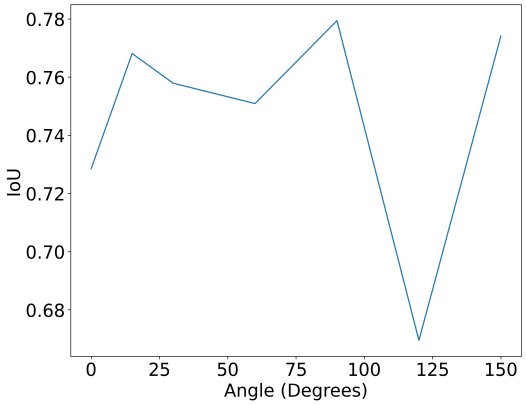


Figure 9: Test IoU's of baseline U-Nets trained with online continuous random rotation for varying maximum angles. The x axis represents the maximum rotation magnitude r which defines the range (-r, r) used when randomly selecting a random angle for rotation. The augmentation as implemented in a samplewise fashion

We also explored the effect of the maximum rotation magnitude on model performance. The results are shown in Figure 9. One can see that the effect size of changing the maximum magnitude of rotation was too small to be detected given the amount of variability due to the small size of our validation and testing sets.

4 Conclusions

In this study, we have proposed a novel *Kernel* Attention U-Net which performs better than the baseline Attention U-Net as well as U-Net by a significant margin (with respect to IoU) for nuclei segmentation tasks (on Data Science Bowl 2018 dataset). It is important to note that this approximately 2.5% and 1.5% IoU improvements over the state-of-the-art Attention U-Net comes at the cost of just 0.5% increase in model parameters with respect to Attention U-Net. We observed that non-linear kernel based multiplication improves the overall performance of the model by enforcing stricter locality between the neighboring channels than the far-away ones, as modulated by the kernel. We have used a decomposable cosine kernel for simplicity of computation. We have noticed that decomposing cosine using Ptolemy's theorem reduces the computational complexity to $\mathcal{O}(n)$ which is not possible with kernels such as Gaussian. If we allow for higher compute, a Gaussian kernel

with a learnable σ can be an interesting choice and might lead to even better performance. We will consider this as a future work.

The loss function used in our experiments suitably combines Dice and Binary Cross Entropy (BCE) losses with a weighting factor λ which we have considered to be unity for all our experiments. However, this factor can be treated as a vital hyperparameter and different values of λ can be used for further exploration to find the best value. Besides, different other hybrid losses can be used such as a weighted combination of IoU and BCE losses or Tversky loss and Hausdorff distance. This might lead to even better performance but we leave these as future works.

Furthermore, we show how differences in preprocessing can have a significant effect on model training and evaluation. For training, we show that the use of interpolation when matching image dimensions introduces noise to the dataset, which decreases model performance. In this paper we only use the default linear interpolation, but in future work it would be interesting to explore how changing the order of interpolation affects our observed results. We also show that unintentional changes in aspect ratio can act as a beneficial form of data augmentation. For evaluation, we show how seemingly minute changes in implementation when processing datasets can have large effects on validation and testing. This highlights the need for a standard method of evaluation that is unaffected by transformations to the output masks. One method of note is the encoded pixel notation used in the 2018 Kaggle Data Science Bowl. However switching to this method of evaluation is on its own not a perfect solution as it disadvantages models that do not make a prediction for every pixel in the mask due to the difference in dimensionality between samples. In theory, U-Net based architectures are actually agnostic to the input image size. However, given the currently available tools, designing and using size agnostic U-Nets is non trivial. Inference can only be parallelized for samples of the same size, as well as the spatial dimensions would need to be tracked during the convolution phase in order to reconstruct the right dimensions during the deconvolution phase (for example, if a filter of size 13x13 is pooled by a factor of 2 to a size of 6x6, then when it is upsampled by a factor of 2 in the deconvolution phase it will have a size of 12x12).

Lastly we explored the various different ways to implement random data augmentations such as rotation. We showed that a batchwise implementation seems to converge more slowly than a samplewise implementation. This indicates that samplewise implementations are more effective when it comes to computational cost, however further work should be done to explore if they reach the

same optimum. Additionally we also explored how the magnitude of rotation might affect model performance, but were unable to find a pattern due to the relatively small effect size when compared to the variance of our validation and testing metrics.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds.), (Cham), pp. 234–241, Springer International Publishing, 2015.

[2] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention U-Net: Learning Where to Look for the Pancreas," Apr. 2018.

[3] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation," *arXiv:1912.05074 [cs, eess]*, Jan. 2020. arXiv: 1912.05074.

[4] D. Jha, M. A. Riegler, D. Johansen, P. Halvorsen, and H. D. Johansen, "DoubleU-Net: A Deep Convolutional Neural Network for Medical Image Segmentation," *arXiv:2006.04868 [cs, eess]*, June 2020. arXiv: 2006.04868.

[5] F. I. Diakogiannis, F. Waldner, P. Caccetta, and C. Wu, "ResUNet-a: a deep learning framework for semantic segmentation of remotely sensed data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 162, pp. 94–114, Apr. 2020. arXiv: 1904.00592.

[6] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, "TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation," *arXiv:2102.04306 [cs]*, Feb. 2021. arXiv: 2102.04306.

[7] H. Cao, Y. Wang, J. Chen, D. Jiang, X. Zhang, Q. Tian, and M. Wang, "Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation," *arXiv:2105.05537 [cs, eess]*, May 2021. arXiv: 2105.05537.

[8] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: When to warp?," in *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–6, 2016.

[9] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, July 2019.

[10] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations," July 2017.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, Nov. 2016. Google-Books-ID: omivDQAAQBAJ.

[12] J. C. Caicedo, A. Goodman, K. W. Karhohs, B. A. Cimini, J. Ackerman, M. Haghighi, C. Heng, T. Becker, M. Doan, C. McQuin, M. Rohban, S. Singh, and A. E. Carpenter, "Nucleus segmentation across imaging experiments: the 2018 data science bowl," *Nature Methods*, vol. 16, pp. 1247–1253, Oct. 2019.