Semantic Segmentation of Bioimages Using Convolutional Neural Networks

Stiaan Wiehman* and Hendrik de Villiers*
*CSIR-SU Centre for AI Research, Department of Computer Science, Stellenbosch University, Stellenbosch, South Africa
Email: stiaan@aims.ac.za

***Abstract*—Convolutional neural networks have shown great promise in both general image segmentation problems as well as bioimage segmentation. In this paper, the application of different convolutional network architectures is explored on the *C. elegans* live/dead assay dataset from the Broad Bioimage Benchmark Collection. These architectures include a standard convolutional network which produces single pixel outputs, as well as Fully Convolutional Networks (FCN) for patch prediction. It was shown that the custom image processing pipeline, which achieved a worm segmentation accuracy of 94%, was outperformed by all of the architectures considered, with the best being 97.3% achieved by a FCN with a single downsampling layer. These results demonstrate the promise of employing convolutional neural network architectures as an alternative to ad-hoc image processing pipelines on optical microscopy images of *C. elegans*.**

I. Introduction

Bioimage analysis is a useful tool for researchers to quickly and accurately process the vast amount of image data that is obtained from high-throughput imaging technology. There are a wide variety of applications for bioimages, some of which include being able to distinguish between different cell types, to identify specific pathogens or disease, or to track certain cellular organisms or organelles. These applications are especially useful in the healthcare sector and pharmaceutical industry. A review of these applications is provided in [1].

A specific area of interest is to apply semantic segmentation methods on bioimages. By providing a semantic label for every pixel in an image, it is possible to not only determine the location of a desired biological object with high accuracy, but also to classify the object. Examples of this include differentiating between cell types in a tissue sample [2] or determining whether a microscopic organism is alive or dead. One important challenge with bioimages that doesn't occur with the more general image datasets like ImageNet [3] or the PASCAL VOC Challenge [4], is that the ground truth for bioimages needs to be annotated by an expert of the respective field or through use of specialized analytical labelling techniques which can be expensive and time consuming [5]. This usually results in fairly small datasets (a couple of hundred images or less) with incomplete or partial ground truth.

Neural networks have shown great promise in semantic segmentation tasks, delivering state-of-the-art performance on large segmentation datasets like PASCAL VOC [6], SIFT Flow [7] and Stanford Background [7], [8]. However, small datasets can be a challenge for neural networks, as they often have a large amount of trainable parameters, risking overfitting. This paper aims to explore architectures for the purpose of semantic segmentation on a small dataset, the *C. elegans* live/dead assay dataset. *C. elegans* is a useful model organism that is employed in a wide variety of biological fields, such as genomics and neuroscience. It is particularly interesting because of its simple anatomical structure, short life cycle and that its entire genome is known [9]. Automated image processing pipelines operating on images of this model organism therefore have high potential utility. The *C. elegans* dataset forms the core benchmark employed in this paper. Four different neural network architectures were considered, which included a standard convolutional network and three fully convolutional networks (FCN) [6].

The main contribution of this work is a demonstration of a variety of convolutional neural network architectures on this benchmark, showing that neural networks are a promising alternative to existing custom image processing pipelines [10].

A. Related Work

Convolutional neural networks have received much attention, often in the context of tasks involving large general image datasets such as ImageNet and Pascal VOC. Unfortunately, for bioimages, each dataset has its own specific type of output and degree of complexity and good performance on one dataset does not necessarily imply good performance on another dataset. In Hatipoglu et al. [11], a six layer convolutional neural network was shown to outperform both a k-nearest neighbour and a support vector machine approach in classifying cells in histopathology images. In Jain et al. [12], a convolutional neural network was used to restore electron microscopy images. In Kraus et al. [5], an FCN with a multiple instance learning (MIL) [13] layer was employed for classification of cells in mammalian and yeast datasets.

Two convolutional neural network architectures were identified as potentially suitable for semantic bioimage segmentation on the *C. elegans* dataset. The first model architecture is based on Ciresan et al. [14], which uses a 10-layer model comprised of four convolutional layers, each followed by a max pooling layer, followed by two fully connected layers. They validated their model on the ISBI 2012 EM Segmentation Challenge. The second model is based on the FCN developed by Long et al. [6]. They adapted the well-known deep convolutional neural network, AlexNet [15], into a fully convolutional network and used it to perform semantic segmentation. They verified their model on three different segmentation benchmark datasets:

978-1-5090-0620-5/16/$31.00

PASCAL VOC 2011, NYUDv2 and SIFT Flow. Variations of the base FCN managed to outperform state-of-the-art methods on all three benchmark datasets, showing that fully convolutional networks are a viable alternative to conventional classification convolutional networks. FCN's have also received some attention in bioimages. In Ronneberger et al. [16], a deep fully convolutional network with 23 convolutional layers was applied to three different bioimage datasets: the ISBI 2012 EM Segmentation Challenge and two from the ISBI 2014/15 Cell Tracking Challenge. Chen et al. [17] employed a FCN with a hierarchy of downsampling layers, where each level in the hierarchy contains a classification stream. The multiple classification streams are then fused to produce the final classification output. They validated their network on the ISBI 2012 EM Segmentation Challenge data and achieved state-of-the-art performance, outperforming both [14] and [16].

In previous work on the *C. elegans* live/dead assay, Wählby et al. [10] created a pipeline specifically tailored to segment and classify the worms. They preprocess the images by compensating for different levels of illumination. The images are then thresholded, which then acts as a binary segmentation mask. A skeletonization technique is then applied on the binary segmentation masks in order to separate individual worms and create single worm segmentations. Segmentation accuracy was determined based on these single worm segmentations. They considered a single worm segmentation to be correct if it has an F-measure of 0.8 or above. This lead to a segmentation accuracy of 81% using automated binary segmentation. Segmentation accuracy was improved to 94% with manually corrected binary segmentation. Lastly, using CellProfiler Analyst, the single worm segmentations were used to train a classifier to distinguish between the live and dead classes. This classifier was trained on a manually selected subset of the worms in the *C. elegans* dataset, but was evaluated on a larger, private dataset, reporting an accuracy of 97% and a precision of 83%.

In the following sections, a detailed description will be given of the dataset, the experiments and the results that followed.

II. Methods

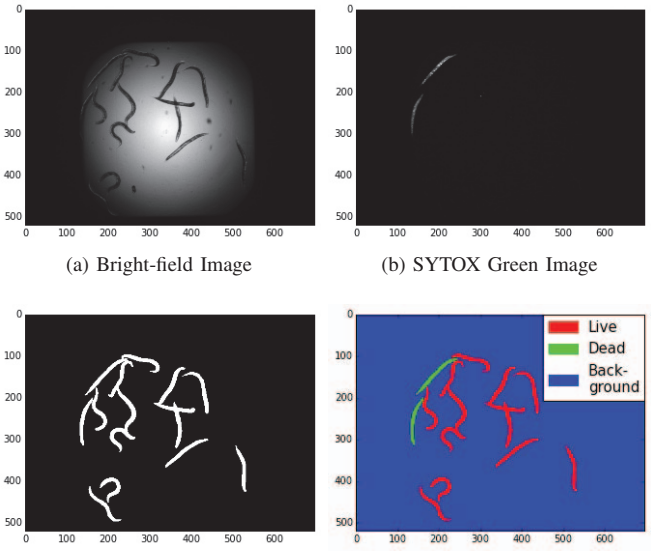
This section will give a brief description of the experimental setup that was used to create and train the various models. It will then give a detailed description of the dataset followed by the experimental procedures and models that were employed.

A. Setup

The models were trained on a high-end desktop workstation containing an Intel Core i7-4790 3.6GHz CPU, 16GB of main memory and an NVIDIA GeForce GTX980 Ti graphics processor with 6GB of memory. The models were implemented and trained using Theano [18], [19] in Python.

B. Dataset

The dataset employed was the *C. elegans* live/dead assay, version 1, available from the Broad Bioimage Benchmark Collection (BBBC) [20]. This dataset consists of 97 16-bit TIFF files of bright-field microscopy images. It is split into 52 images containing predominantly live individuals and 45 images containing predominantly dead individuals. In predominantly live images, most of the worms in the image should display the live class and similarly for predominantly dead images. The live worms tend to be curved in shape and smooth in texture. The dead worms have a rod-like shape and are uneven in texture. These are some of the visual cues that the neural networks could potentially learn to recognize.



(a) Bright-field Image (b) SYTOX Green Image

(c) Binary Ground Truth (d) Relabelled Ground Truth

Fig. 1. An example from the dataset. Figure 1a shows the bright-field image and 1b its corresponding SYTOX Green image. Figure 1c shows the binary mask and 1d the relabelled RGB mask. Best viewed in color.

Also included in the archive are 98 16-bit TIFF files of SYTOX Green stained fluorescent microscopy images[1]. Each of these images correspond to a single bright-field image. Both of these sets have a fairly simplistic background; however, as is often observed in microscopy images, they suffer from uneven illumination. This can, in some cases, make it difficult to observe the class of a worm. One such example can be seen in Figure 1a. Uneven illumination can be observed in both the bright-field and SYTOX images.

There are two sets of ground truth labels, available separately from the BBBC website. The first contains 100 binary segmentation masks that shows the location of all the worms in each image (example in Figure 1c). This is one of the main ground truth labellings that will be used in experiments and will henceforth be referred to as the binary masks. The second set also contains binary segmentation masks; however, in this case, each file shows the location of a single worm in an image. There are 1407 files available, with multiple files corresponding to each bright-field image. These files will henceforth be referred to as the single worm segmentations.

Due to the nature of the segmentation masks, no clear

[1]SYTOX Green stain chemically highlights dead individuals.

ground truth live/dead classification label were given to each worm. A predominantly live image could potentially contain dead worms; however, because no label is provided for each individual worm, this can lead to incorrect labelling during training and testing. A similar argument can be made for the predominantly dead images. This could cause potential confusion for the neural network, hence a new set of ground truth images was created as an attempt to reproduce the ground truth used in [10]. This was achieved by using the SYTOX images and the single worm segmentations.

The SYTOX images, such as Figure 1b, were thresholded in order to detect the highlighted worms. The single worm segmentations that correspond to highlighted worms in the SYTOX images were then relabelled as belonging to the dead class. The remaining single worm segmentations were relabelled as belonging to the live class.

Now that each worm had a designated class, they were merged to create a three channel image with the same resolution as the binary masks, where each channel represents a single class. A value of 254 in a respective channel indicates that the pixel at that location belongs to the corresponding class. There are also some cases where two or more worms would overlap, and should the overlapping worms be of different classes, the pixels in the overlapping region were given the value 127 in both the live and dead channel. This indicates that those pixels could be seen as either live or dead. These images were saved as 8-bit RGB images in PNG format and will henceforth be referred to as the relabelled masks.

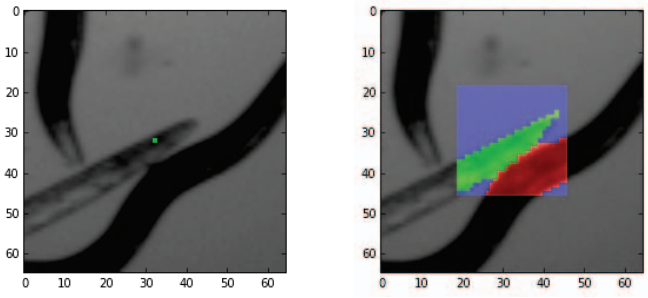
C. Experimental Setup

This section will explain further processing on the dataset for training purposes, the metrics employed to determine the performance on the data, how training was done and the main model architectures that were used.

1) Data: The only preprocessing that was applied to the data involves shifting and scaling the 16-bit bright-field images such that each image has pixel values in the range $[0, 1]$. The images, when loaded, are also cropped to 501×501 in size, ensuring that the lit section is roughly centred and allowing them to be stored within a single Numpy array.

This dataset does not have a dedicated validation or test set, and as such, a 5-fold cross validation was performed. With 97 images in the set, two randomly chosen images, one from each predominant class, were put aside and the remaining 95 images were randomly divided into 76 training images and 19 validation images. These splits were kept consistent for all experiments. Secondary experiments and tests were done using the first fold to be able to compare performance.

The images were loaded in batches, from which 2048 random patches were extracted. Square patches were randomly sampled based on the class of the center pixel. To ensure symmetry, the patches had to have odd length edges. The three classes were chosen with equal probability, which ensured that each set of 2048 patches were fairly balanced. The randomness introduced in the sampling of the patches ensured that each iteration was different from the next, and greatly reduced the chances of the network overfitting to the training data. Even though there are only 76 images in the training set, an input patch size of 101×101 allows enrichment to about 12 million training images. A single set of 2048 patches was considered to be one iteration, and for every batch of images, a certain number of iterations were performed. The number of iterations scaled according to the number of images in the current batch. One epoch is considered to be one pass through all the image batches. The number of iterations for the validation set is increased fivefold to compensate for the relatively small number of images in the set. The target label for each patch was generated as depicted in Figure 2.

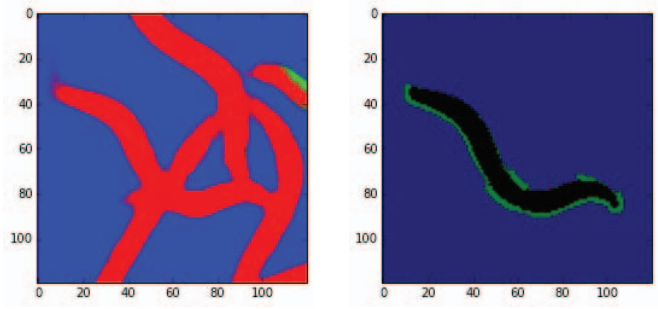


(a) Pixel Target (b) Patch Target

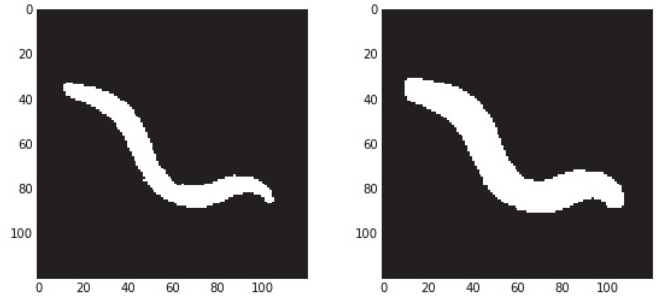
Fig. 2. Pixel target (Figure 2a) versus patch target (Figure 2b). The standard convolutional network receives the green pixel in Figure 2a as target. The FCN's will receive the RGB patch in Figure 2b as target. Red, green and blue correspond to live, dead and background, respectively. Best viewed in color.

2) Metrics: The metrics used in [10], which were accuracy, precision, recall and F-measure, will be reported. These will be given in four sets: mean pixel-level segmentation (MPS), mean pixel-level classification (MPC), mean worm-level segmentation (MWS) and mean worm-level classification (MWC). The pixel-level metrics are calculated over the pixels in the validation set, averaged by the total number of pixels. The worm-level metrics are calculated for each worm using the single worm segmentations, averaged by the total number of worms. To calculate the preceding metrics for segmentation, the positive class in the binary classification problem was considered to be both live and dead worms, and the negative class the background. For classification, the positive class was considered to be live and the negative class, dead. The background is ignored in this case. Furthermore, an extra metric is added to the MWS set to compare with [10], which employed an F-measure threshold of 0.8 to determine how many worms were segmented correctly. A worm is considered to be correctly segmented if it has an F-measure above 0.8.

The neural networks employed in this paper produce an output for each image pixel, making it relatively simple to calculate the pixel-level metrics. The results produced in [10], however, are produced in the worm domain. This difference in output domains poses the challenge to convert from the pixel-level domain to the worm-level domain. This is an important consideration in comparing the neural network results with [10]. This was done by dilating the binary single worm



(a) Network Output (b) Result after Dilation



(c) Single Worm Segmentation Mask (d) Dilated Mask

Fig. 3. The reason for dilation for the worm-level statistics. Figure 3a shows the probability output of model A (Figure 4). Figure 3d is the resulting mask after Figure 3c was dilated. Figure 3b shows why dilation is necessary. With reference to Figure 3b and the binary classification problem for MWS, black represents True Positive, green represents False Positive, red represents False Negative and blue represents True Negative. Best viewed in color.

segmentations (Figure 3c) using a 5×5 kernel of ones, which resulted in the binary segmentation shown in Figure 3d. For a single worm instance, all the pixels of the network output (Figure 3a) that were classified as either live or dead were considered to be a worm pixel should they intersect with the dilated mask. All other pixels, no matter the class, were considered to be background pixels. This results in the segmentation shown in Figure 3b. The green pixels show the amount of oversegmentation that the model produced, better known as false positives. Without these pixels representing oversegmentation, it would not have been possible to produce accurate worm-level results. Unfortunately, some worms do overlap with each other, causing pixels from neighbouring worms to be included. The consequence of using a dilated mask is that the results will indicate that the neural networks are slightly worse than what they truly are, as opposed to using the original mask, in which case the neural networks would seem better than what they truly are. For example, should the original mask be used to evaluate model performance for the worm in Figure 3, a precision of 100% will be obtained. To include the produced oversegmentation, the dilated mask must be used, which results in a much lower precision of 74.5%. A number of 3×3 dilations were tested using the network output that visibly caused the most oversegmentation. Upon visual inspection, it was concluded that two 3×3 dilations, or

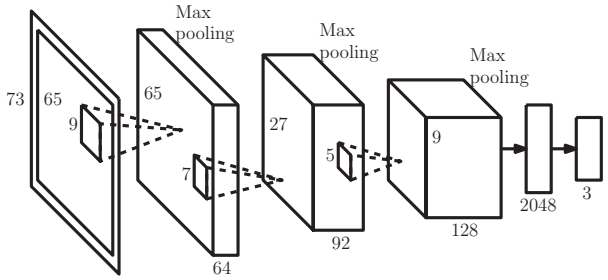


Fig. 4. Model A. The 65 × 65 input patch was padded to 73 × 73. All convolutional kernel strides are 1. All max pooling kernels are 2 × 2 with a stride of 2. This model produces single pixel predictions (see Figure 2a).

equivalently, one 5×5 dilation, managed to include most, if not all, of the oversegmented worm pixels without including a significant amount of pixels from neighbouring worms.

3) Training: All models were trained using the sampling strategy from Section II-C1. The models were trained for 200 epochs using the standard cross-entropy cost function, averaging over per-pixel errors. Training was done using stochastic gradient descent, a minibatch size of 256 and the adaptive learning rate, ADADELTA [21]. The number of iterations were chosen such that learning converged before 150 epochs. The time of convergence for each model can be controlled by changing the number of iterations per image batch.

4) Model Architectures: Four main neural network architectures were used. The first model (model A) is based on the model used in [14], and is illustrated in Figure 4. It has one less convolutional and max pooling layer, larger filters and more feature maps. It uses parametric rectified linear units (PReLU) as activation function, as it has been shown to have improved performance on image data [22]. The input layer was zero-padded by half the kernel size of the first layer. This model will provide a baseline for the rest of the network architectures.

The simplest FCN model, model B (depicted in Figure 5) has four convolutional layers. The first three use PReLU, the last uses softmax normalization over its feature maps. A convolutional layer that uses softmax normalization will henceforth be referred to as a ConvSoftmax. The simplicity of the network lies in the fact that it has no downsampling layers and is relatively shallow. The result of this is that it has a small contextual window compared to model A. The contextual window in this instance is the patch size required to produce a 1×1 feature map size in the deepest convolutional layer (before upsampling). With this model, the effect of a small contextual window as well as training on a patch rather than a single pixel is explored.

As an alternative to the patch size shown in Figure 5, a smaller as well as a larger patch was explored to determine what influence the patch size has on the performance of model B. The smaller patch, 27×27, was chosen in such a way that the output size was a single pixel. Performance was similar to that of the model in Section III-B, the difference being that it had significantly lower segmentation precision. This suggests that the model suffered from slight oversegmentation when

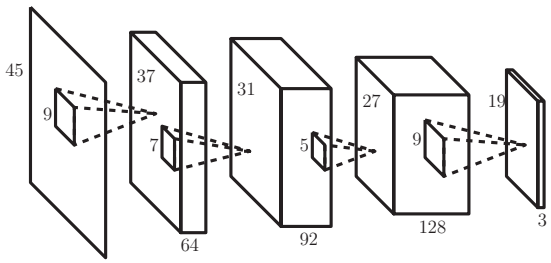


Fig. 5. Model B. All convolutional kernel strides, including the ConvSoftmax output layer, are 1. This model produces patch predictions (see Figure 2b).

trained on a single pixel target. The larger patch, 65×65, was chosen in such a way that most of the GPU memory was used. Performance was significantly worse when classifying worms. It also showed a significant decrease in segmentation recall. This suggests that the model is focussing on correctly classifying background pixels, which is reasonable considering the over-abundance of background pixels in the images, the size of the patch and that the cost function is not weighted.

Expanding on model B, a single downsampling layer was attached to the first convolutional layer, as depicted in Figure 6. The network was also made two convolutional layers deeper. Initially, manual upsampling through interpolation was explored. This yielded comparable classification statistics; however, segmentation was significantly worse. In light of this result, a trainable upsampling technique was considered, which resulted in a deconvolutional layer added just before the output layer. This leads to the third model, model C.

Model C (depicted in Figure 6) is also partially based on the FCN model used in Long et al. [6]. It has five convolutional layers using PReLU, with a max pooling layer attached to the first layer. Attached to the last layer is a deconvolutional layer, which upsamples the feature maps before it is given to a ConvSoftmax layer. Downsampling by use of a larger kernel stride in the first convolutional layer was also explored; however, it did not show a significant change in the results.

The architecture for model C was then augmented to be similar to the model in Ronneberger et al. [16], with one less downsampling layer as it was considered that the resulting increase in the contextual window was unnecessary. The network architecture with three max pooling layers has a contextual window which is roughly the size of an average fully grown worm. This leads to the final model, model D.

Model D is illustrated in Figure 7, and consists of a downsampling stream and an upsampling stream. The downsampling stream has six convolutional layers, with every second layer having a max pooling layer attached starting from the first layer. The upsampling stream has three deconvolutional layers, each followed by a convolutional layer and a ConvSoftmax layer as output. The output of the fourth convolutional layer in the downsampling stream is cropped to have the same feature map size as the output of the first deconvolutional layer. The output of these two layers are then stitched together and given as input to the first convolutional layer in the upsampling stream. A similar process follows for the input of the second convolutional layer in the upsampling stream. All convolutional layers use PReLU. Unlike the previous models, model D had to be trained on input patches with an even edge length. This approach was chosen in order to avoid loss of information on the borders when downsampling.

III. Results

This section will present the results obtained from experimentation with the described architectures. Model A (Figure 4) was subjected to a 5-fold cross validation using the binary masks as targets. After careful investigation of the output masks generated by the model, it was found that the model managed to detect both live and dead worms in a single image, while the corresponding binary mask provided as ground truth with the dataset, suggests that all the worms are of the same class. This occurrence raised the possibility that the model could be classifying certain worms correctly, but due to the nature of the binary segmentation masks, those worms were seen as incorrect. The opposite could also be true, that the model classified worms incorrectly, but it was seen as correct according to the ground truth. It was then decided that the ground truth labels needed to be relabelled using the procedure in Section II-C2. The results of the original experiment were included as reference in Table I, referred to as Standard Convolutional Network - Binary Pixel Target (SCN-BPT).. Results from model A through D on the relabelled ground truth are described in the corresponding sections that follow, and are shown in Table I as mean $\pm$ standard deviation.

A. Standard Convolutional Network - Pixel Target (SCN-PT)

First, a baseline was necessary. Model A (Figure 4) was again subjected to a 5-fold cross validation, this time using the relabelled masks. The model managed to perform even better than in SCN-BPT with regards to segmentation, but worse on classification. This difference can only be due to relabelled ground truth, as the hyperparameters of the model were kept the same. The model managed to achieve a segmentation accuracy of 95.88%, which outperforms the ad-hoc pipeline in Wählby et al. [10]. Classification accuracy was not as good, 88.72% compared to their 97%; however, model A did manage to achieve a significantly larger precision, with 87.95% compared to their 83%. A qualitative segmentation example is given in Figures 8c and 9c.

Given that model A managed to outperform [10] and that FCN's have shown great promise in semantic segmentation applications [6], [16], [17], it was thought that the following FCN's would improve on the result produced by model A.

B. Fully Convolutional Network - No Downsample (FCN-ND)

The simplest model, model B (Figure 5), was subjected to a 5-fold cross validation, using a patch target instead of single pixel target, as shown in Figure 2b. The results are shown in Table I, as well as Figures 8d and 9d. The model managed to improve in most of the segmentation metrics as well as the MWC metrics. It did, however, perform worse on the pixel

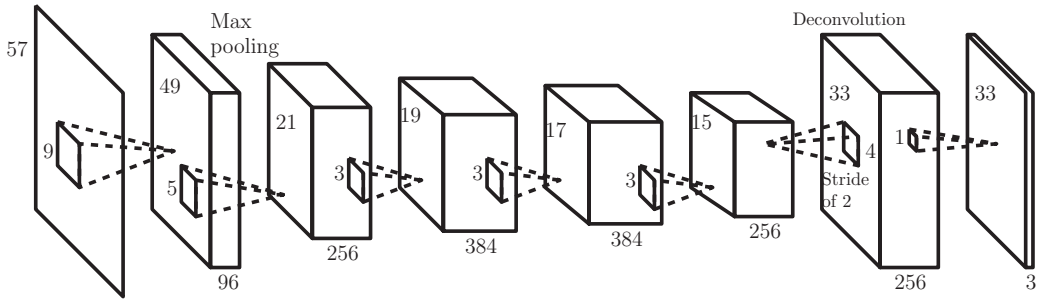


Fig. 6. Model C. All convolutional kernels including the ConvSoftmax kernel had a stride of 1. The single max pooling layer had a 2×2 kernel with a stride of 2. This model produces a patch of predictions (example in Figure 2b).

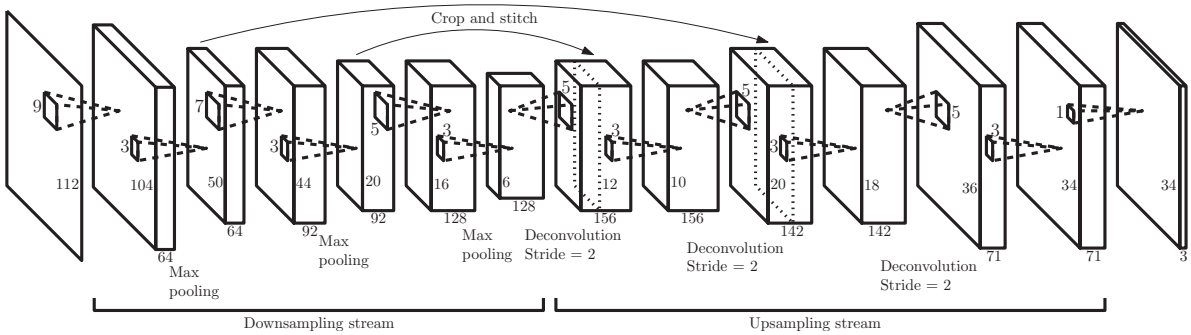


Fig. 7. Model D. All convolutional kernels including the ConvSoftmax kernel had a stride of 1. The three max pooling layers have 2×2 kernels and strides of 2. The first convolutional layer in the upsampling stream has skip-connections to the output of the fourth convolutional layer in the downsampling stream. Similar for the second convolutional layer in the upsampling stream. This model produces a patch of predictions (see Figure 2b).

TABLE I
THE RESULTS FOR THE PRIMARY EXPERIMENTS OUTLINED IN SECTION III.

		Pixel					Worm					
		SCN-BPT	SCN-PT	FCN-ND	FCN-1D	FCN-3D	SCN-BPT	SCN-PT	FCN-ND	FCN-1D	FCN-3D	[10]
Seg.	Accuracy	*0.9806±0.0012*	0.9820±0.0004	0.9902±0.0005	**0.9904±0.0008**	0.9898±0.0011	*0.9989±0.0000*	0.9990±0.0000	**0.9994±0.0000**	**0.9994±0.0000**	**0.9994±0.0001**	-
	Precision	*0.7955±0.0113*	0.8096±0.0038	0.9328±0.0122	0.9340±0.0101	**0.9362±0.0051**	*0.7937±0.0060*	0.8154±0.0035	**0.9285±0.0120**	0.9277±0.0097	0.9236±0.0051	-
	Recall	*0.9842±0.0044*	**0.9820±0.0049**	0.9201±0.0088	0.9296±0.0095	0.9425±0.0104	*0.9670±0.0081*	**0.9794±0.0065**	0.9190±0.0088	0.9281±0.0105	0.9297±0.0145	-
	F-Measure	*0.8798±0.0058*	0.8875±0.0039	0.9263±0.0046	0.9316±0.0064	**0.9393±0.0069**	*0.8698±0.0034*	0.8879±0.0044	0.9213±0.0048	**0.9254±0.0079**	0.9235±0.0111	-
	Seg. Acc.	-	-	-	-	-	*0.9409±0.0153*	0.9588±0.0150	0.9686±0.0067	**0.9730±0.0119**	0.9662±0.0219	0.9400
Class.	Accuracy	*0.8844±0.0176*	0.8619±0.0071	0.8338±0.0067	0.8437±0.0077	**0.8666±0.0272**	*0.8970±0.0169*	0.8872±0.0071	0.9015±0.0182	0.8984±0.0237	0.8894±0.0277	**0.9700**
	Precision	*0.8920±0.0420*	0.8582±0.0342	0.8265±0.0210	0.8341±0.0277	**0.8647±0.0305**	*0.9142±0.0427*	0.8795±0.0352	0.8798±0.0189	**0.8830±0.0193**	0.8706±0.0269	0.8300
	Recall	*0.9016±0.0233*	0.8569±0.0445	0.8312±0.0207	0.8482±0.0241	**0.8696±0.0503**	*0.8990±0.0336*	0.8806±0.0489	**0.9147±0.0236**	0.9025±0.0327	0.8973±0.0404	-
	F-Measure	*0.8962±0.0253*	0.8559±0.0097	0.8284±0.0101	0.8400±0.0092	**0.8668±0.0185**	*0.9057±0.0242*	0.8781±0.0089	**0.8966±0.0124**	0.8914±0.0159	0.8828±0.0212	-

classification metrics. This decrease in pixel classification is likely due to the small contextual window the model needs to work with, as it obscures the overall shape of the worms. This suggests that a larger contextual window is needed.

C. Fully Convolutional Network - 1 Downsample (FCN-1D)

Building on the results of model B, the effect of adding downsampling was explored. Model C (Figure 6) was subjected to a 5-fold cross validation. The model managed to perform the best overall with regards to the segmentation metrics, showing a significant improvement in mean segmentation accuracy with 97.30% (Table I). Pixel-level classification also improved marginally compared to that of model B, which could be as a result of the larger contextual windows; however, the difference in worm classification was negligible. Figures 8e and 9e provide qualitative examples of segmentation and classification performed by model C.

D. Fully Convolutional Network - 3 Downsamples (FCN-3D)

To explore the effect of having multiple downsampling layers, model D (Figure 7) was then subjected to a 5-fold cross validation. The model managed to significantly improve pixel-level classification, showing the highest mean metric values among the four models that trained on the relabelled masks. The differences observed in the segmentation metrics were negligible. Even though model D managed to produce the best mean metric values for pixel-level classification out of all the FCN experiments, it was still outperformed on the worm-level by both model B and C for both segmentation and classification. Qualitative segmentation examples can be found in Figures 8f and 9f.

IV. DISCUSSION

All of the network architectures that were tested in this paper managed to outperform Wählby et al. [10] with respect

to the segmentation accuracy of the worms and classification precision; however, the highest classification accuracy was lower than in [10]. Since the results of [10] were obtained on a larger, private dataset, it is necessary to assume that the larger dataset showed a similar worm diversity as the public *C. elegans* live/dead assay dataset that was used. Another assumption needed for comparison with [10], was that live was chosen as the positive class in the binary classification problem of classifying the worms. The model that showed both the best worm-level segmentation accuracy and classification precision was model C, with 97.30% and 88.30%, respectively, compared to the 94.00% and 83.00% of [10]. The model that showed the second best worm-level classification accuracy was model B, with 90.15% compared to the 97.00% of [10].

An interesting observation from Table I is that the standard convolutional networks (SCN-BPT and SCN-PT, Sections III and III-A) show significantly higher segmentation recall than precision. Regarding segmentation, precision is interpreted as a measure of oversegmentation and recall as a measure of undersegmentation. Ideally, the higher these metrics, the less over-/undersegmentation is occuring. The FCN models on the other hand, show a balanced distribution between these two metrics. This result is a quantitative example of why FCN's with patch-wise training are better for segmentation tasks.

Another interesting observation is that for all of the models that were trained on the relabelled masks, their classification metrics for the respective levels fall within a similar range. Pixel-level classification average about 85% for all the metrics and worm-level classification about 89%. Despite a number of adjustments to the architectures, these results did not improve further. It was then further investigated by inspecting the output of the models for various input images and it was found that the illumination level of the images had an influence on the classification performance of the models. Images with poor illumination make it difficult for the models to extract the textural information of the worms, and in some instances, the boundary between worm and background (Figure 9). This suggests that, should the illumination levels of the images be corrected, or the models are made invariant to differences in illumination, they should perform significantly better. In light of this result, it begs the question as to why SCN-BPT did so well on pixel-level classification compared to the rest of the models. A likely hypothesis is that, potentially, the model learnt to classify the center pixel as live based on the amount of live pixels it sees in the input patch, essentially applying a majority rule between the amount of live and dead pixels. This kind of approach does not work in SCN-PT, as it is expected of the network to distinguish between the class of the worm in the center and its neighbouring worms in the input patch.

Having multiple downsampling layers together with the skip-connection architecture to pass different level encodings to deeper layers seem to be crucial for the model to make pixel-level classification decisions. Two possibilities are immediately evident, the first being that this kind of architecture allows for the model to take the classes of neighbouring worms into consideration, much like in SCN-BPT and SCN-PT. The second possibility is that, since most of an adult worm could fit into the contextual window of model D, it takes the shape of the entire worm into consideration. Models B and C have a much smaller contextual window size, which would only allow a fraction of the entire worm to be in view. The difference in architecture, and inherently the difference in the contextual window size, does not seem to have a significant influence over segmentation at the pixel- and worm-level; however, it is clear from the results in Table I that it does influence the mean pixel-level classification performance of the models. This trend can be seen more clearly by ordering the models based on the size of their contextual window. The model with the smallest contextual window is model B (FCN-ND), followed by model C (FCN-1D), then model A (SCN-PT) and then lastly, the model with the largest contextual window is model D (FCN-3D). The same order can be observed by placing the pixel-level classification metrics in ascending order.

V. Conclusion

A variety of different neural network architectures were tested on the *C. elegans* live/dead assay dataset, all of which showed significant improvement of key metrics over the ad-hoc image processing pipeline in Wählby et al. [10]. Although the networks did not perform as well with regards to worm classification accuracy, it did manage to outperform the pipeline in both worm segmentation accuracy and worm classification precision. The FCN approaches performed significantly better than the standard convolutional model with regards to segmentation. It was noted that downsampling played a key role in enabling the FCN's to achieve higher pixel-level classification accuracy. It was found that the standard convolutional network architecture tends to oversegment the worms, based on its relatively low segmentation precision. Lastly, it was noted that low illumination levels could obscure key morphological information, such as texture or the border between worm and background.

In this work, a variety of neural network architectures were explored, all of which have shown a significant improvement over the existing ad-hoc pipeline on key metrics, suggesting that neural networks are a promising alternative to current methods operating in this domain. It was also shown that fully convolutional networks provide a significant improvement on segmentation tasks over standard convolutional networks.

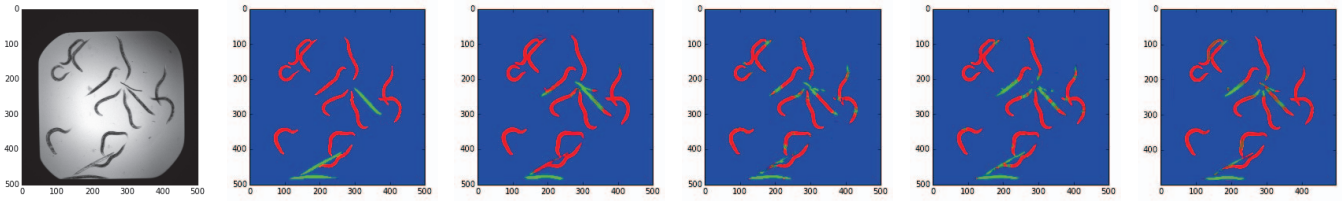
Acknowledgment

The authors would like to thank the National Research Foundation (NRF) of South Africa for their financial support.

References

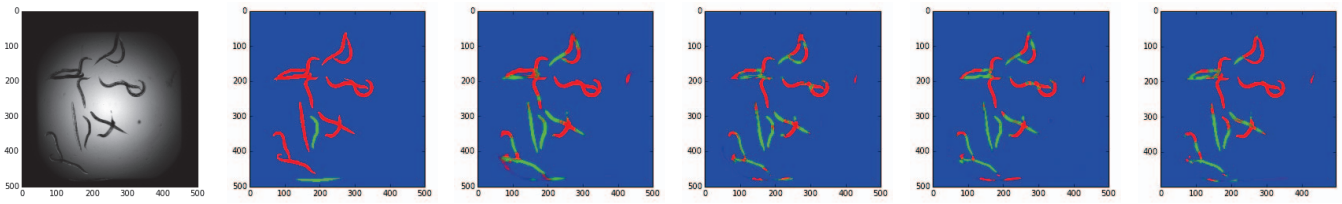
[1] H. Peng, "Bioimage informatics: a new area of engineering biology," *Bioinformatics*, vol. 24, no. 17, pp. 1827–1836, 2008. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/24/17/1827

[2] A. M. Khan, S.-E.-A. Raza, M. Khan, and N. M. Rajpoot, "Cell phenotyping in multi-tag fluorescent bioimages," *Neurocomputing*, vol. 134, pp. 254 – 261, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231214000988



(a) Input Image (b) Ground Truth (c) Model A (d) Model B (e) Model C (f) Model D

Fig. 8. A segmentation example showing the output probabilities generated by each model for the given input image (Figure 8a) and its corresponding ground truth label (Figure 8b). Figures 8c, 8d, 8e and 8f show the segmented output produced by models A, B, C and D respectively. Red, green and blue correspond to live, dead and background.



(a) Input Image (b) Ground Truth (c) Model A (d) Model B (e) Model C (f) Model D

Fig. 9. An example input (Figure 9a) where low illumination levels hampered classification by the models. The ground truth is given in Figure 9b. Figures 9c, 9d, 9e and 9f show the segmented output produced by models A, B, C and D respectively. Red, green and blue correspond to live, dead and background.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009, pp. 248–255.

[4] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[5] O. Z. Kraus, L. J. Ba, and B. J. Frey, "Classifying and Segmenting Microscopy Images Using Convolutional Multiple Instance Learning," *CoRR*, vol. abs/1511.05286, 2015. [Online]. Available: http://arxiv.org/abs/1511.05286

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3431–3440.

[7] P. H. O. Pinheiro and R. Collobert, "Recurrent Convolutional Neural Networks for Scene Parsing," in *Proceedings of the 31st International Conference on Machine Learning*, 2014. [Online]. Available: http://jmlr.org/proceedings/papers/v32/pinheiro14.pdf

[8] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 129–136.

[9] A. K. Corsi, B. Wightman, and M. Chalfie, "A transparent window into biology: A primer on Caenorhabditis elegans," *Genetics*, vol. 200, no. 2, pp. 387–407, 2015.

[10] C. Wählby, L. Kamentsky, Z. H. Liu, T. Riklin-Raviv, A. L. Conery, E. J. O'Rourke, K. L. Sokolnicki, O. Visvikis, V. Ljosa, J. E. Irazoqui *et al.*, "An image analysis toolbox for high-throughput *C. elegans* assays," *Nature Methods*, vol. 9, no. 7, pp. 714–716, 2012.

[11] N. Hatipoglu and G. Bilgin, "Classification of histopathological images using convolutional neural network," in *Proceedings of the International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Oct 2014, pp. 1–6.

[12] V. Jain, J. Murray, F. Roth, S. Turaga, V. Zhigulin, K. Briggman, M. Helmstaedter, W. Denk, and H. Seung, "Supervised Learning of Image Restoration with Convolutional Networks," in *Proceedings of the IEEE International Conference on Computer Vision*, Oct 2007, pp. 1–8.

[13] J. Foulds and E. Frank, "A review of multi-instance learning assumptions," *The Knowledge Engineering Review*, vol. 25, pp. 1–25, 3 2010. [Online]. Available: http://journals.cambridge.org/article_S026988890999035X

[14] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2843–2851. [Online]. Available: http://papers.nips.cc/paper/4741-deep-neural-networks-segment-neuronal-membranes-in-electron-microscopy-images.pdf

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Proceedings of Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[17] H. Chen, X. Qi, J.-Z. Cheng, and P.-A. Heng, "Deep Contextual Networks for Neuronal Structure Segmentation." [Online]. Available: http://appsrv.cse.cuhk.edu.hk/ hchen/research/2012isbi_seg.html

[18] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Jun. 2010, oral Presentation.

[19] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[20] V. Ljosa, K. L. Sokolnicki, and A. E. Carpenter, "Annotated high-throughput microscopy image sets for validation," *Nature Methods*, vol. 9, no. 7, p. 637, 2012.

[21] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *CoRR*, vol. abs/1502.01852, 2015. [Online]. Available: http://arxiv.org/abs/1502.01852