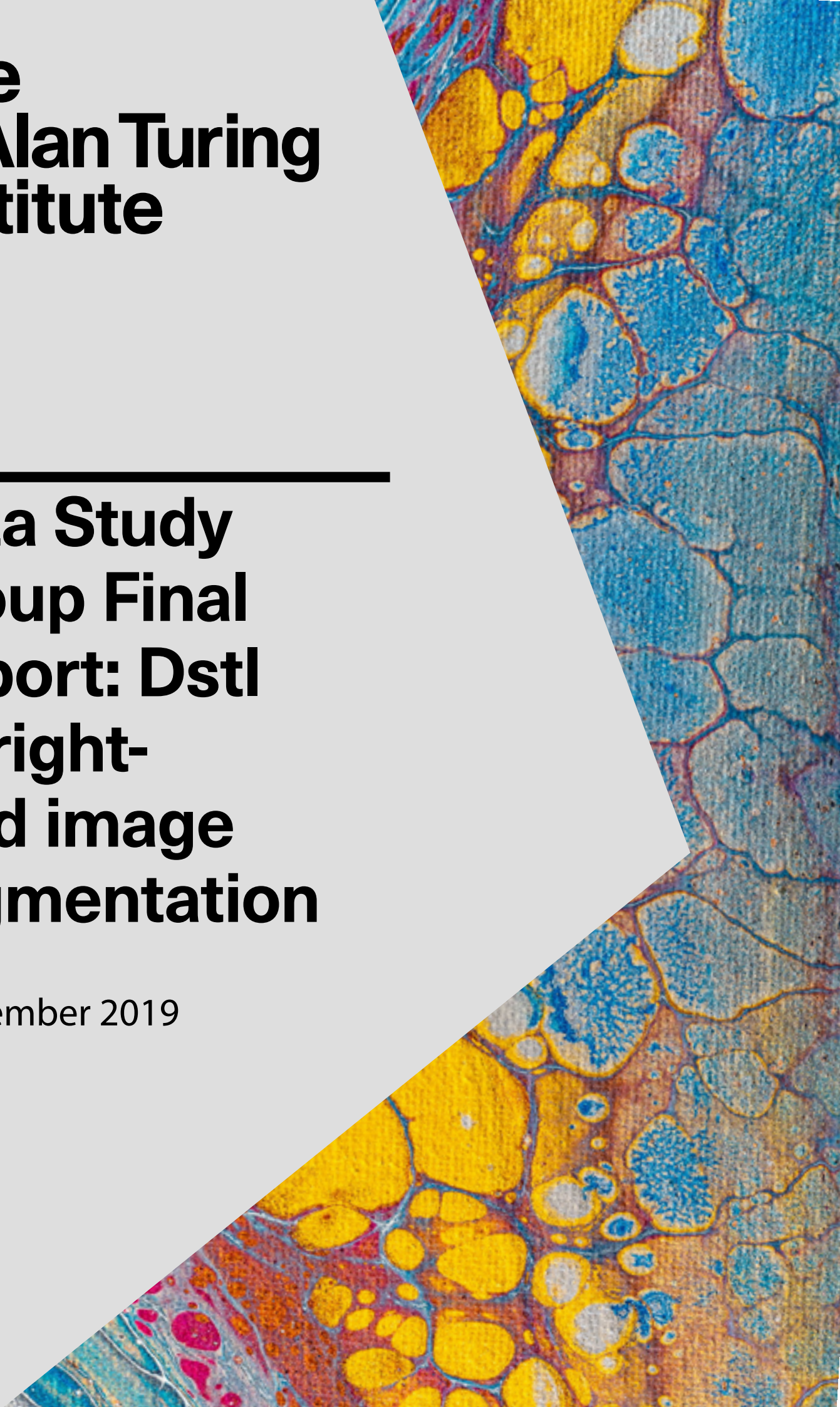
The Alan Turing Institute

Data Study Group Final Report: Dstl – Bright-field image segmentation

8 December 2019



Dstl – Bright-field Image Segmentation

Data Study Group

8 Dec 2019

https://doi.org/10.5281/zenodo.4452761

1 Summary

1.1 Challenge overview

The challenge was to test instance segmentation methods on transmitted light (bright-field) confocal microscopy images of cells. Instance segmentation is desirable because it allows for automatic cell counting and quantification of individual cell morphologies. Typically instance segmentation is done using fluorescence microscopy, and nuclear staining, but this has several drawbacks including photo-toxicity for live cell assays. Bright-field microscopy is used extensively in biomedical research hence the applications for the challenge are extensive. For example DSTL in particular is interesting in studying models of infection where segmentation of cells is an important step in the processing pipeline.

1.2 Data overview

Our main dataset consists of 100 transmitted light confocal microscopy images along with ground truth images that contain masks for each individual cell. The images are 1152×1152 and represent a significant segmentation challenge due to the low contrast and many touching cells.

1.3 Main objectives

The principle challenge is to develop segmentation protocols to identify individual cells (macrophages) from transmitted light, or bright-field images. This is a relatively simple task when cells are well separated but very difficult when they are densely packed together. This can be broken down as follows:

- Perform semantic segmentation
- Perform instance segmentation on individual cells
- Automatically count the number of cells in a given bright field image
- Use instance segmentation to provide quantitative measurements of cell morphologies

1.4 Approach

We explored the following avenues of research:

- Data Augmentation
- Generative Adversarial Networks (Pix2Pix) [1, 2]
- Self Organizing Maps [3]
- Discriminative Loss [4]
- UNET [5]
- Mask RCNN [6]

1.5 Limitations

The key limitations of each of our experiments are as follows:

- Pre-processing: The transformed data may produce shapes that no longer look like cells. This could potentially make the segmentation even harder.
- Generative Adversarial Network (Pix2Pix): Common caveats of deep learning approaches such as a lack of interpretability and long training times.
- Discriminative Loss: The feature dimension size requires tuning but increasing it incurs a large computational cost.
- Self Organizing Maps: There was too much noise present in the images for clustering.
- UNET: The network is designed for semantic segmentation so another step is required to achieve instance segmentation.
- Mask RCNN: Results were limited by the model architecture, which is designed for high contrast RGB images and commonly predicts low instance numbers.

1.6 Future work

The key pieces of future work are as follows:

- Generative Adversarial Network (Pix2Pix): Change the architecture so that it doesn't down-scale the prediction images.
- Discriminative Loss: Perform hyper-parameter tuning of the number of feature dimensions.
- Self Organizing Maps: Use dynamic re-organizing of the SOM units.

- UNET: Use the network for border prediction and then use an algorithm to fill in the bordered regions e.g. watershed algorithm.
- Mask RCNN: Fine-tune all the layers and lower the object detection threshold.
- Post-processing: Fine tune the threshholding algorithm and/or use morphological analysis to make decisions about when to split cells.

1.7 Conclusion

The group tested several different machine learning based approaches aiming to perform either semantic, or full instance segmentation, of cells within bright-field images. Two of these methods (self organising maps and descriminative loss) were unsupervised which is highly desirable for this challenge where the training set is very small. However although these methods showed some promise we consider it unlikely that they will compete with the tested supervised techniques even accounting for the small training set. With regards to semantic segmentation preliminary results showed UNET to be a powerful framework and this could be further improved by training to recognise boundary regions in addition to signal and background. Mask RCNN is a popular framework for instance segmentation and variations on this architecture showed some promise for this challenge but additional hyperparamter tuning and testing is needed to realise its full potential. Deep generative architectures, specially CGANs, gave promising results particularly when trained to produce distance maps. Future work improving the post-processing of these distance map outputs could potentially lead to robust instance segmentation of bright-field images.

2 Background

Confocal microscopy is a tool many bio-medical researchers use to gather data about their field of interest, including study of disease and infection. DSTL uses confocal microscopy to identify novel targets for anti-microbial therapies and to understand the effect of medical countermeasures. Confocal microscopy is a form of light microscopy which scans a laser point by point to build up an image. This is combined with a pinhole in the light path to remove out-of-focus light. DSTL uses a range of assays where cells are grown in-vitro (outside of the body) and imaged with a confocal microscope to study the effects of infection, or chemicals, on host cells such as macrophages. Macrophages are a type of white-blood-cell which fight infection by engulfing and digesting pathogens (bacteria, virus, fungi etc). Fluorescent bacteria are used to infect the cells which allows us to see where the bacteria go within the cell and what they do to the cell during infection. Fluorescent labels have the special property that they absorb the excitation light, here a laser beam, and then emit light at a lower frequency. By using a selective emission filter we can then produce high contrast images of the bacteria. However florescent labelling and imaging of the macrophages

Dish	**No. Images**	**No. Cells**
1	35	1092
2	16	886
3	49	835
Total	100	2813

Table 1: Number of images and cells from images where manual annotations were available.

can dramatically change the behaviour of these sensitive cells and should be avoided if possible. Therefore to identify the location of the cells a transmitted light image is captured. This simply captures the light which passes through the sample and produces a low contrast image of the cells which is much harder to process than florescent data.

Automated analysis of these images typically starts with cellular segmentation (the process of identifying individual cells within images). This is routinely done using high contrast fluorescent labels for either the cytoplasm or plasma membrane. Segmentation using label free modalities such as transmitted light/bright-field microscopy is advantageous because it is less phototoxic than fluorescent imaging and removes the need for labels which may affect the function of the cells. To the human eye cells are easy to identify on a transmitted light image, however due to the similarities in pixel values between the background and cell events segmentation by computational analysis is still a real challenge. This DSG (Data Study Group) challenge aimed to utilise the power of AI to design methods to segment cells from confocal microscopy datasets of human/murine immune cells infected with various pathogens.

3 Data overview

3.1 Dataset description

The image data used in the DSG was generated specifically for this challenge. The images are of a macrophage cell line (RAW 264.7) grown on 35mm tissue culture dishes, seeded at different densities and imaged with confocal microscopy (transmitted light). Macrophages are a type of immune cell which engulf pathogens and were used because they are of interest to DSTL for studying infection. The methods developed in this project could be applied to other cell lines and biomedical applications. The data was generated to take the challenge back to the very basics of segmentation; find and outline the cells within the images.

A subset of the data was manually annotated to provide ground truth segmentations. These images were taken from three experimental dishes and are detailed in Table 1. An example image and corresponding segmentation is shown in Figure 1.

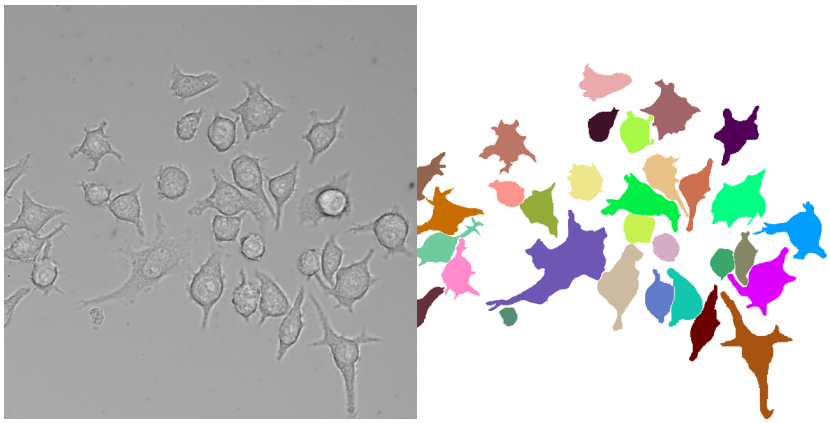


Figure 1: Example transmitted light confocal microscopy image and ground truth instance segmentation. The ground truth was produced by manual annotation. Image dimensions are 1152 × 1152 and are in grey-scale 8-bit format. Each cell in the ground truth image has a unique integer pixel value, background pixels have value zero.

3.2 Data challenges

The primary challenges with the data is that it is low contrast, which makes the segmentation problem significantly harder. In addition to this the cell morphologies can differ greatly, particularly in terms of size and circularity. A final issue is that cells can be touching which makes dissociating between distinct cells significantly harder.

4 Quantitative problem formulation

In this section we break the challenge down into quantitative objectives and propose metrics to evaluate performance.

1. **Can we use machine learning to semantically segment bright-field images?** As a first step we wanted to assess how accurately we could perform semantic segmentation on bright-field images. Semantic segmentation involves classifying each pixel in the image as either belonging to a cell or the background. Importantly this approach does not separate touching cells and treats all cells as belonging to the same class.

   To assess the accuracy of our approaches on the semantic segmentation problem we used the Jaccard index (intersection over Union), J, which is calculated as follows:

$$J = \frac{|\, S \cap R \,|}{|\, S \cup R \,|}$$

where R are the pixels belonging to cells in the ground truth reference masks and S are the pixels belonging to cells according to the reconstructed masks.

2. **Can we use machine learning to accurately segment individual cells?** Our second goal was to accurately count the number of individual cells present in a given image. To achieve this we need to perform instance segmentation, which not only identifies cells but also distinguishes between different cells by assigning them different values. Calculating cell counts then reduces to a count of the number of unique values in the image.

   To assess the accuracy of our approaches on the instance segmentation problem we used the Jaccard index again but at the cell level as proposed for the Cell Tracking Challenge [7]. For each reference cell, R_i, we calculate the union between reconstructed cells, S_i, and the corresponding matched reference cell, R_i. Cell S_i is said to match reference cell R_i if

$$| S_i \cap R_i | > \frac{| R_i |}{2}$$

   such that for each R_i at most one S_i can be matched. The calculation of the averaged cell level Jaccard index is as follows:

$$J_C = \frac{1}{N_{ref}} \sum_{i=1}^{N_{ref}} \frac{| S_i \cap R_i |}{| S_i \cup R_i |}$$

   where N_{ref} is the number of reference cells in the image. We also report the accuracy of our cell counts by taking the absolute different between predicted and reference cell counts and normalise by the number of reference cells,

$$A_c = 1 - \frac{| N_{pred} - N_{ref} |}{N_{ref}}$$

   where N_{pred} is the predicted number of cells in the image.

3. **Can we use machine learning to describe the morphology of individual cells?** The final research question is whether we can use instance segmentation to describe the morphology of the cells within the images. This involves outputting descriptive statistics on attributes such as cell height, width, size and circularity. The results of this research question are critically dependent on the accuracy of the instance segmentation methods.

5 Experiment: Data augmentation

5.1 Description

Creating a Python class that facilitates data augmentation through the implementation of the following geometric transformations: rotation, zoom, shear, crop and flip. Images of cells allow for all these transformations since no referential is required for them to make sense. Data augmentation is useful for both training and validation of the methods developed. The idea is to use the augmented images to increase the size of the training set.

5.2 Outcomes

We created a class (BFImageAugmentator) which creates a DataGenerator from Keras[1] with inputs for rotation range, shear range, zoom range, horizontal flipping (boolean) and vertical flipping (boolean). The class contains methods for coupled transformation of the raw image data and ground truth mask. Aug-mented data is automatically saved in a user defined directory. To use the class, one can run the following code;

```
from DataAugmentation import BFImageAugmentator

new_generator = BFImageAugmentator()
new_generator.runAugmentation(dish_number=3, number_new=5)
```

6 Experiment: Image Segmentation using Pix2Pix

6.1 Description

GANs are generative models based on deep learning. Generative modelling is a task where the model discovers patterns in the input data in order to generate new samples that could have been drawn from the original dataset. GANs are based on a min-max game between two networks, the generator and the discriminator, which try to fool each others. The generator learns how to generate fake samples while the discriminator learns how to distinguish fake samples from the samples drawn from the input data distribution. The input of the generator is just random noise while the output is a sample which should look like an example drawn from the employed dataset.

CGANs is a variation of GAN where instead of using random noise as input, the generator uses a sample from a different distribution. In the image-to-image translation scenario the generator takes as input the image that we want to translate and generates its translated version.

We tested a Pix2Pix model for segmenting cells in bright-field images. Pix2Pix is a model developed for image-to-image translation of paired images. It is based

[1] https://keras.io/preprocessing/image/

on conditional generative adversarial networks (CGAN) [1, 2]. We have pairs of images representing edges and objects. The edges represent the shape of the object. The model takes edges as input and produces the corresponding objects or vice-versa. In our scenario we used bright-field images as input and produce the segmented images as the output.

6.2 Rationale

Our task is based on finding cells in a bright-field image. From an image processing perspective this means assigning a different label to each cell in the image i.e., assigning different grey-scale values. Indeed, our ground truth images (known as masks) have a different grey-scale values for each cell. We decided to treat our task as an image-to-image translation as we want to map the bright-field image onto the annotated mask. Therefore, we decided to explore the Pix2Pix model for the bright-field image segmentation task given its success in other image-to-image translation scenarios.

6.3 Experimental setup

The dataset includes 80 paired images for training and 21 for testing. Each pair includes the bright-field image and its corresponding segmented annotation. The Pix2Pix repository was used in this experiment[2]. We considered three different representations of the segmented masks for the output;

1. A grey-scale value is assigned to each cell randomly. It means that every image has grey-scale values in random positions. We noticed some difficulties when training with this annotation. The reason is that there is no actual mapping between each cell in the bright-field image and the corresponding grey-scale value in the mask.

2. Grey-scale values are assigned to each cell in order from left to right. The leftmost cell is annotated with the lowest value, i.e., the darkest, while the rightmost cell is annotated with the highest value i.e., the brightest.

3. A Distance Transform means that we interpret the mask as a topographic map, and each point represents their height, in this case, their distance to the background or the cell-cell boundary. There is also a ridge, a line that separates two different parts, in our case, cells that will be close together. See Section 11 for post-processing of this representation to recover the original mask representation.

6.4 Outcomes

We trained the pix2pix model with different CGAN flavours, specifically with vanilla, mean square and Weisstein loss functions. Taking into account the three

[2]https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

different methodologies for generating the mask annotations as ground truth, the model has been been able to reproduce very good predictions even for those images where the complexity of the fragmentation was very high. The predictive power of the model is particularly promising when the Distance Transformation output is used as shown for an example in Figure 2 for the CGAN model with a Weisstein loss function.

(a) Input image (b) Target mask (c) Output image

Figure 2: Example CGAN prediction after a training cycle of 100 epochs. The generator was trained to output (output image) a Distance Transform representation of the ground truth annotation (target Mask).

7 Experiment: Self-Organizing Maps

7.1 Description

The Self-Organizing Map (SOM) is a simple network of neurons that takes values from an array and maps them onto output responses such that they are of the same topological order as the original array [3]. It is an unsupervised Artificial Neural Network that uses competitive learning as opposed to error-correcting learning to generate low-dimensional maps of real data. Competitive learning is an unsupervised leaning approach similar to statistical clustering whereas error correction is the conventional supervised learning approach where an error function between the system output and labelled training data is minimised. The rationale behind this approach was to generate a network capable of producing instance segmentation of cells from bright-field microscopy images without the need for any manual segmentation. The advantage of this method comes from the fact that the SOM does not need any manually labelled ground truth masks for training. Usually, the availability of manually segmented images in microscopy is the limiting factor for supervised learning methods. This is especially limiting in research applications as different cell types, experimental setups and microscopes all produce slightly different outputs that may affect the performance of a previously trained network.

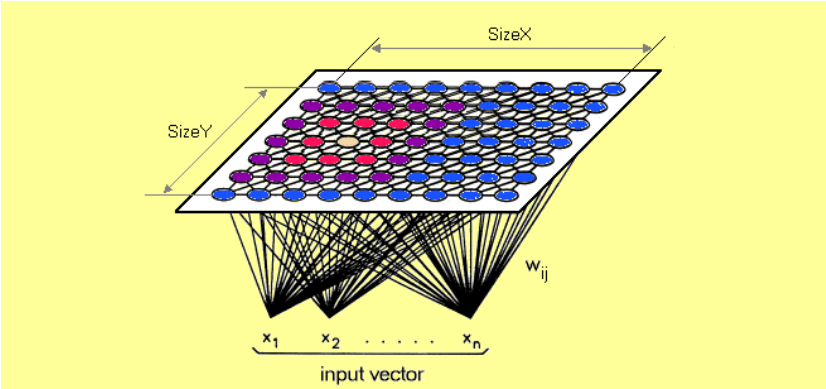


Figure 3: Self-Organising map structural illustration (from https://medium.com/@abhinavr8/self-organizing-maps-ff5853a118d4).

7.2 Experimental setup

The SOM is a neural network with input data $X \in \mathcal{R}^j$, $t = 1, 2, ..., n$. Initially, all the weights w_{ij} are small random numbers where each w_{ij} is associated with each neuron i. Each neuron i has position r_i on the grid of neurons and at the first iteration $k = 1$, we calculate the winning neuron v_k by using $v_k = argmin||x_k - w_{ik}||$. Then update the weights with $\Delta w_{ik} = \lambda_k \eta(v, i, k)[x_k - w_{vk}]$. For this project, the learning rate $\lambda_k = a/(1 + k/K)$ where a is a constant and K is the maximum number of iterations and the neighbourhood function is $\eta(v, i, k) = exp(-\frac{||r_v - r_k||^2}{2\sigma^2})$. Figure 3 demonstrates the structure of this network. Since the SOM is an unsupervised machine learning technique, no supervised training was needed and the clustering was applied directly to test images.

7.3 Outcomes

When an SOM with 9 feature neurons was trained and applied to the data, we obtain the result shown in Figure 4. From this, we can see that the lowest value feature value has mapped to the cell boundaries in the original image. However, due to the low contrast of the original image, much of the cell interior has been mapped onto the same feature value as the background.

In order to segment this mask into instances, we can take the result from the SOM and class every distinct, separate region of the mask and label them as shown in Figure 5a. When comparing with the ground truth results, the labelling of the pixels is not unreasonable. However, this is still not a full mask of the cells. From here we decided to test a variety of image segmentation methods within the skimage.segmentation module. Of the tested methods the Felzenszwalb efficient graph based image segmentation [8] worked the best and and an example is shown in Figure 5b. The segmentation step of this process

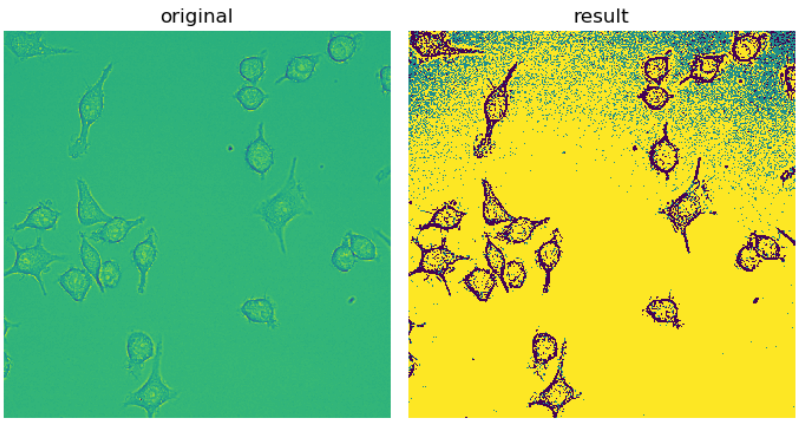


Figure 4: Self-Organising map result for 9 feature neurons.

still needs to be improved since we can see that although the SOM is able to detect most of the cell boundaries the segmentation of these boundaries into separate regions is not as effective. In order to reduce the amount of noise in the detection of boundaries, two SOMs were put in series to detect boundaries and then the segmentation protocol was applied (Figure 5c).

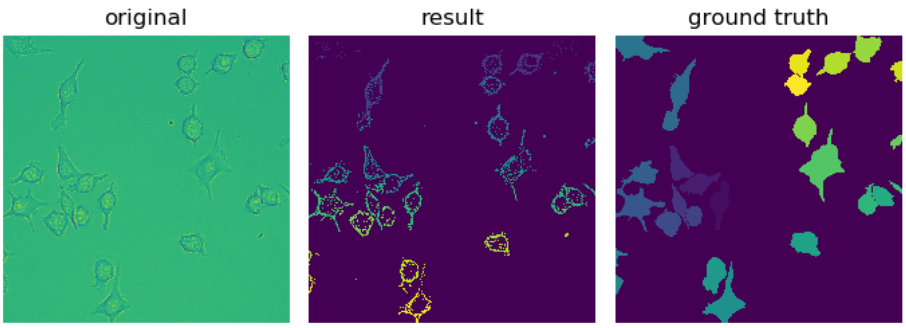
7.4 Conclusion

With relatively simple and fast neural network structures, we can detect the boundaries of the cells quite effectively. This provides a platform on which we can explore different methods for instance segmentation of cells in bright-field images. Deep learning methods could potentially be used in post-processing to determine which boundaries belong to each cell.

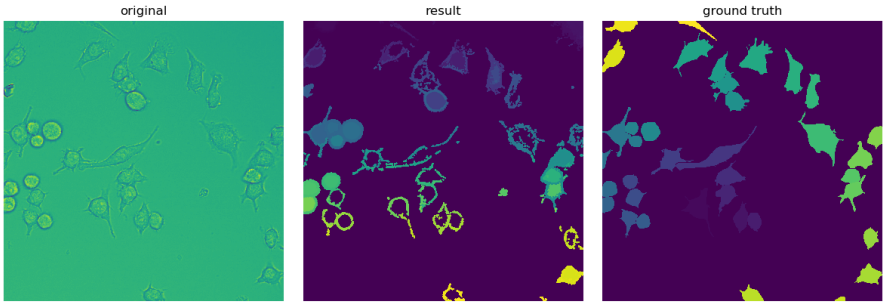
8 Experiment: Discriminative Loss

8.1 Description

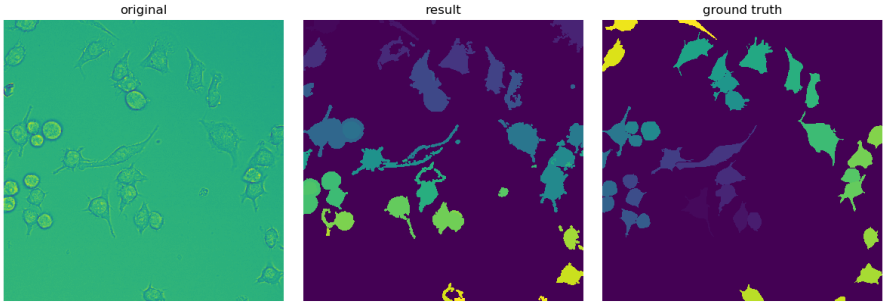
We can also treat the problem of segmenting cells as a vector embedding problem. For this we used a discriminative loss function that assigns pixels in an object to the corresponding vector in embedding space to identify objects with unsupervised clustering [4]. Recently, this approach has been successfully applied for accurate instance segmentation including with overlapping objects [9, 10].



(a) Each distinct component from the SOM output is labelled with a unique pixel value.



(b) Post-processing with the Felzenszwalb efficient graph based image segmentation from the skimage.segmentation module.



(c) Example results from two SOMs in series followed by graph based image segmentation.

Figure 5: Example results from Self Organizing Map experiments.

The task was to use the labelled images to train the neural network to semantically segment the images of bright-field microscopy. Then to segment these into separate features in an unsupervised clustering using a discriminative loss function.

8.2 Experimental setup

Note we used a TensorFlow implentation[3] with our own modifications. The images are transformed into dots, or embeddings, in a n-dimensional feature space, where n can be adjusted through hyperparameters. Intuitively, instances (cells) should clustered together as they are close in the feature space. The discriminative loss function can be defined as follows:

$$L_{var} = \frac{1}{C} \sum \frac{1}{N_c} \sum [||\mu_c - x_i|| - \delta_v]_+^2$$

$$L_{dist} = \frac{1}{C(C-1)} \sum_{C_A=1}^{C} \sum_{C_B=1}^{C} [2\delta_d - ||\mu_{c_A} - \mu_{c_B}||]_+^2 \quad where \quad \mu_{c_A} \neq \mu_{c_B}$$

$$L_{reg} = \frac{1}{C} \sum_{C=1}^{C} ||\mu_c||$$

$$L = \alpha \cdot L_{var} + \beta \cdot L_{dist} + \gamma \cdot L_{reg}$$

There are two repelling forces in the function, L_{var} and L_{dist}, which are the intra-cluster pulling force from a cluster of similar instances and the inter-cluster pushing force to separate different instances in the image (Figure 6). C is the number of clusters in the ground truth and N_C is the number of elements in the cluster, x_i is the pixel embedding in the n-dimensional space, μ_C is the mean embedding of cluster c, $||a-b||$ stands for the distance between a and b, and δ_v and δ_d are the margins for the variance and distance loss. The loss function is weighted by α, β and γ, where in default $\alpha = \beta = 1$ and $\gamma = 0.001$. See Figure 6 for an illustration.

The images are read in and resized to 512×512, different batch size, feature dimensions were tested during the experiment. The learning rate was set to 10^{-4}. All training images and testing images were selected and randomly sampled with different batch size. Here we improved the sampling method which allowed us to process multiple data in a single batch, although later we found out a single image performs better than larger batch size. This is probably because of the differences in background signal between images. Different feature dimensions were tested during the experiment and may affect the number of steps needed for convergence (Figure 7).

[3] https://github.com/hq-jiang/instance-segmentation-with-discriminative-loss-tensorflow

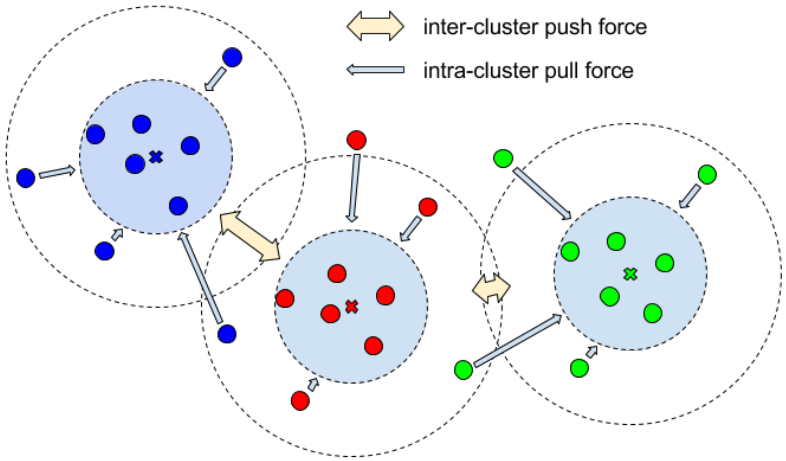


Figure 6: There are two repelling forces in the function which are the intra-cluster pulling force from a cluster of similar instances and the inter-cluster pushing force to separate different instances in the image. Figure adapted from [10].

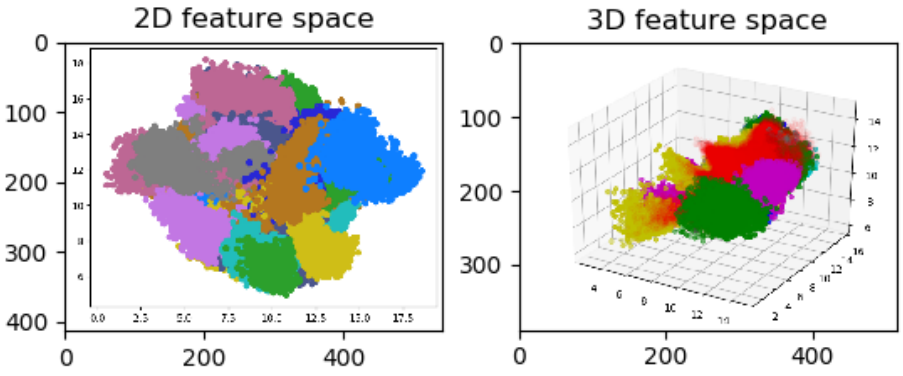


Figure 7: 2D and 3D feature spaces illustration.

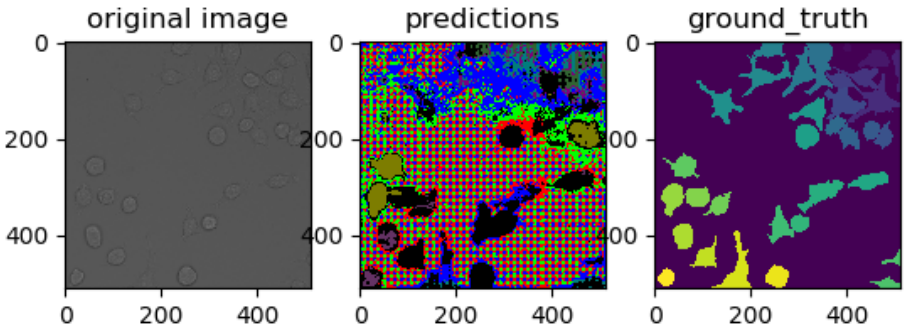


Figure 8: Example prediction for 2D feature space (4000 iterations, loss 0.4).

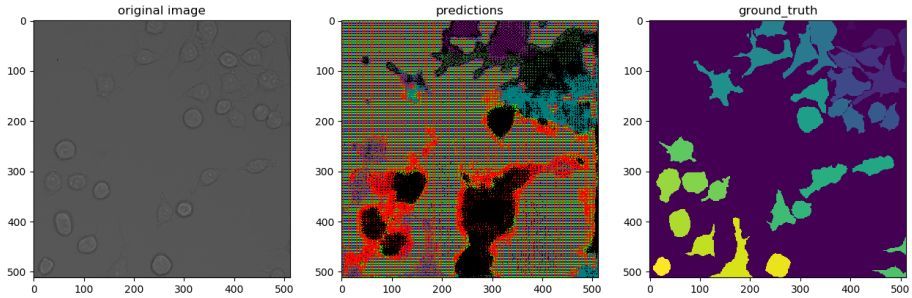


Figure 9: Example prediction for 3D feature space (700 iterations, not converged).

8.3 Outcomes

A feature dimension of 2 was used in the first setup, the network was well trained with loss reduced to 0.4 after 4000 iterations. However it failed to cluster the embeddings in the 2D feature space, with cells only partly recognized (Figure 8).

We then increased feature space to 3, as this gives more space for the clusters to be separated from each other (Figure 7). Unfortunately the training was interrupted by out of space error but even with limited training time (700 steps, not converged) this method yielded a better result (Figure 9).

The results in Figure 9 show that the increase in feature dimension can greatly increase the accuracy of the model, however given limited time we cannot try the model with larger dimension. Figure 10 shows the preliminary result for feature space set to 15 after 800 steps.

From the results in Figure 10 we can see that increase the space dimension has a great impact on model accuracy but significantly increased the time needed for training. It is also noteworthy to say that the model can be optimized in many ways, such as a better data feeding approach and/or validation pipeline.

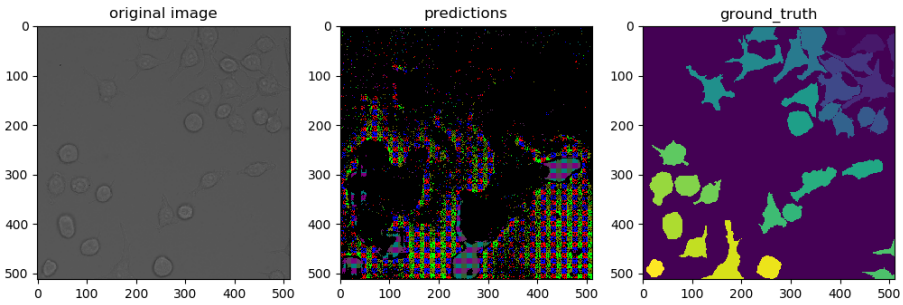


Figure 10: Preliminary result for feature space set to 15 after 800 steps.

8.4 Conclusion

The discriminative loss function provides an unsupervised way to do semantic segmentation. The image can be transferred into embeddings in n-dimensional feature space which facilitates use on more complicated problem. The advantages are that this method is unbiased, label free and treats the image holistically. However the training time can be demanding if the user chooses high feature dimension.

9 Experiment: Enc-Dec to U-Net

Starting with the common encoder-decoder architecture, we moved towards the structurally similar U-Net Artificial Neural Network [5]. The aim was to semantically segment the cells in the dataset as a precursor to instance segmentation.

The initial Enc-Dec network attempts to reproduce the mask from the input image using a per-pixel L1 loss. U-Net, on the other hand, attempts to predict what class a particular pixel belongs to (in our case one of only two classes) and uses a set of concatenation operations (skip connections) between the levels of convolution. In many cases it has been shown that U-Net performed much better than the naive Enc-Dec network.

9.1 PyTorch implementation

Task descriptions

The first task is to determine what is and what is not a cell. From this basic binary, semantic segmentation, one hopes to determine what is one cell and what is another. To begin with, we started by finding a suitable dataset with which to test my code. We chose the cell tracking challenge HeLA Cell set as a starting point [7]. Data need to be correctly paired, sized and processed before being used for training. We wrote a number of python classes to find the pairs of input image and output mask, resizing all to 512×512 grayscale images.

(a) Input image (b) Target mask (c) Output image

Figure 11: Example result from 4 block U-NET after 15 epochs.

In order to increase the dataset size, we performed basic augmentation by rotating each pair of images by 90, 180 and 270 degrees, thus tripling the amount of data available. With a clean dataset in place, we wrote both a basic Enc-Dec network and a U-Net network in PyTorch using code from an online repository[4]. All the code is written in Python, using PyTorch, Pillow, Python-OpenCV and related libraries.

Experimental set-up

The experiment starts by generating the datasets. A number of images have been specifically set aside as a test set. The remainder are augmented and added to the training set. We have found that around 60 epochs results in acceptable results. The architecture follows the standard U-Net patter, with 4 convolution layers and 4 upscale layers with concatenation between them. The number of channels at each level are 64,128,256 and 512. ReLU activation, maxpool and batch normalisation are also used.

Results

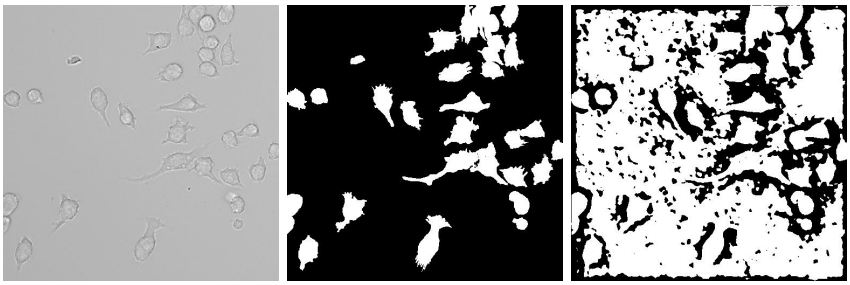
Our network implemented in PyTorch shows mostly good results with a few interesting failure modes. After 15 epochs the output begins to resemble the input masks quite closely (Figure 11). Failure modes are occasionally seen in the output as shown in Figure 12.

9.2 Keras implementation

U-NET Architecture

U-NET is a popular convolutional neural network architecture for biomedical image segmentation [5]. In particular, is has been successfully applied for numerous cell segmentation tasks in microscopy data. The implemented network is constructed of four down blocks and four up blocks. Each down-block consists

[4]https://github.com/milesial/Pytorch-UNet



(a) Input image (b) Target mask (c) Output image

Figure 12: Example result from 4 block U-NET after 15 epochs. Note the network has failed to produce a reasonable segmentation for this example.

of two convolution layers of filter size 3 with instance normalisation, followed by max pooling with a stride of 2 and and a ReLu activation function. The number of channels in the network is doubled at each downsampling step, ranging from 64 to 1024. Up-blocks are composed similarly, replacing max pooling by an upsampling step, and with filter size halving at each step up to size 64. The final convolution layer contains a filter of size 2 and the output layer employs a sigmoid activation function Figure 13.

Experimental set-up

Same padding and He normal kernal initialisers are employed. The Adam optimizer is used with a learning rate of 10^{-4} and binary cross-entropy is chosen as the loss function since we are dealing with a binary classification problem. Images were compressed to size 256x256 using the PIL.Image package. In order to reduce GPU memory the masks were transformed to binary form. The network was trained with 50 epochs and a batch size of 1. Due to the large size and complexity of the network and the limited training set size (81 images), a data augmentation step is crucial to facilitate generalisation to unseen data. This step was performed using the Keras preprocessing.image.Image.DataGenerator. The following transformations were performed: horizontal flips, zooming, shear intensity variations, rotations and shifts in width and height.

The code was adapted from a Keras implementation of U-NET[5] and is based on the architecture in [5]. A few modifications were made, such as including instance normalisation after the convolution layers.

Results

The U-Net was able to successfully identify cells in areas of low cell density. Examples are given in Figure 14 for some of the images in our test set. Images with a higher cell density were harder to predict. In some cases the network

[5]https:/github.com/zhixuhao/unet

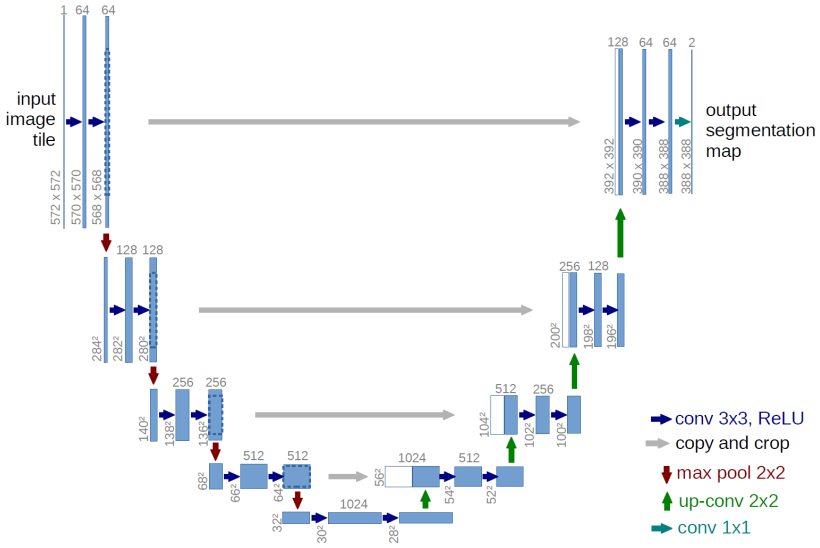


Figure 13: U-Net architecture for Keras implementation. Figure from [5].

performed badly at identifying cell masks (Figure 15). Averaged over the whole test dataset the network achieved a Jaccard index (Intersection over Union) of 0.556 with standard deviation 0.180.

Conclusions

U-NET seems to be capable of providing a semantic segmentation on the DSTL data-set though further experimentation is required in order to understand why the failures occur and how to avoid them. Accuracy measurements should be taken in order to measure further improvements.

Whilst this does not segment individual cells it should be possible to use U-Net to reproduce cell borders by changing the input data. With borders defined it might be possible to segment the image using some kind of flood fill to detect closed areas. A potential solution to provide instance segmentation would be to modify the network to output three classes (cell, boundary and background) that would allow to separate overlapping objects. Alternatively, a second network could be trained to identify cell centres using annotated centroids for training, followed by watershed segmentation on the recreated binary image using these centroids as a mask. The network would also benefit by initialising weights using a pre-trained model such as ImageNet or ResNet.

10 Experiment: Mask R-CNN

Mask R-CNN is an architecture for classifying, locating and segmenting visual objects. It can be trained end-to-end on all three tasks simultaneously. In

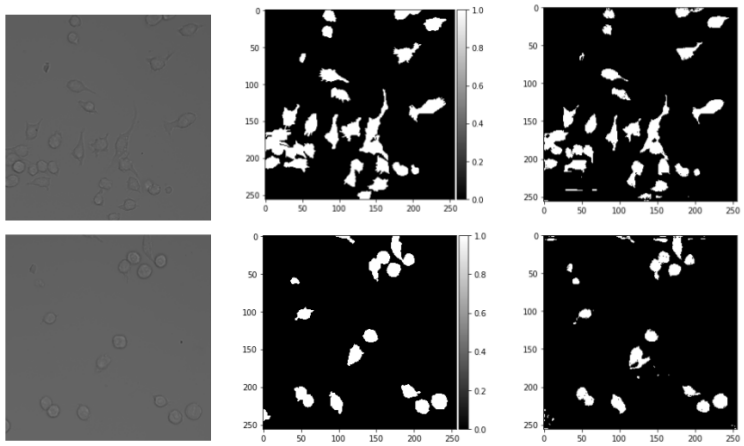


Figure 14: Example U-NET results where the segmentation has worked well. Each row corresponds to a single sample. Left to right: original microscopy image, ground truth binary label, and prediction.

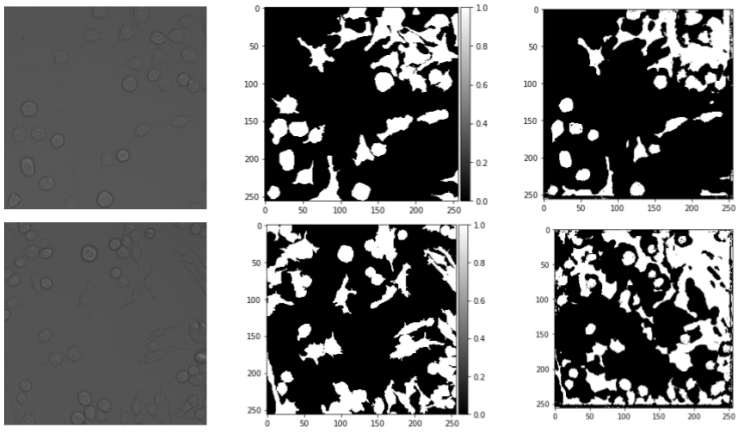


Figure 15: Example U-NET results where the segmentation has not worked well. Each row corresponds to a single sample. Left to right: original microscopy image, ground truth binary label, and prediction.

Mask R-CNN, we first take an image and extract features using the ResNet101 architecture. Next, we take the feature maps from the previous step and apply a region proposal network (RPN). RPN is predicting if an object is present in that region or not. Then these regions are passed through a couple of layers to predict the class label and bounding boxes. After we can compute the Intersection of Union (IoU) with the ground truth. If we have IoU greater than 0.5 we will consider this region as a region of interest. After we get all regions of interest we can get the segmentation mask for each region of interest and get the masks for all objects in the image.

10.1 Strategy

Two possible strategies involved:

- Train the model on external datasets and then check the performance on our data without any additional training step.
- Fine-tuning pre-trained weights from these models with our data and compare performance with the previous method.

10.2 Mask-RCNN 1

10.2.1 Description

In this experiment we used the Mask R-CNN implementation by Waleed Abdulla [6][6]. This is a Tensorflow and Keras based implementation of the Mask R-CNN model, based on FPN and ResNet101.

This repository provides a number of Jupyter notebooks to run the code using weights pre-trained on the MS COCO dataset [11], which is why we chose it. One of the notebooks provides a straightforward way of fine-tuning such a pre-trained model on the our data, which is particularly useful given the small number of annotated images that were provided for this challenge.

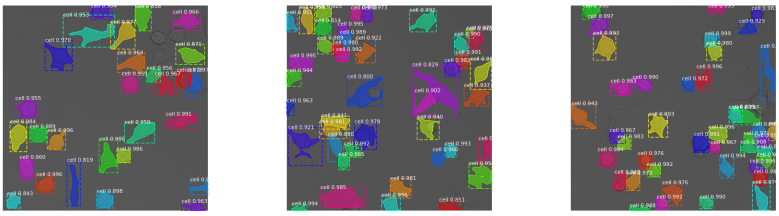
10.2.2 Experimental setup

Feature extractions: 10 epochs/100 steps with initial learning rate applied only to the last layer.

10.2.3 Outcomes

Averaged over all the testing images, we obtain the following results: $J = 0.7$, $J_C = 0.64$, $A_C = 0.9$ (Section 4) (Figure 16).

[6] https://github.com/matterport/Mask_RCNN



(a) (b) (c)

Figure 16: Three examples of cell R-CNN recognition, where masks are superimposed on the bright-field original image.

10.3 Usiigaci network

10.3.1 Description

In the work of Tai et al. [12] Mask R-CNN was applied to bright-field cell segmentation. They called their network Usiigaci. It is is based on the Keras implementation of Mask R-CNN[7].

10.3.2 Experimental setup

Usiigaci is pre-trained on the COCO image dataset [11], and then fine-tuned on a dataset of 50 cell images. Both the COCO dataset and the fine-tuning dataset are annotated with bounding boxes and masks for each cell. The plan was to load the weights from their model, apply a further fine-tuning step on our own training (and validation) dataset, and then apply inference to our test set.

10.3.3 Outcomes

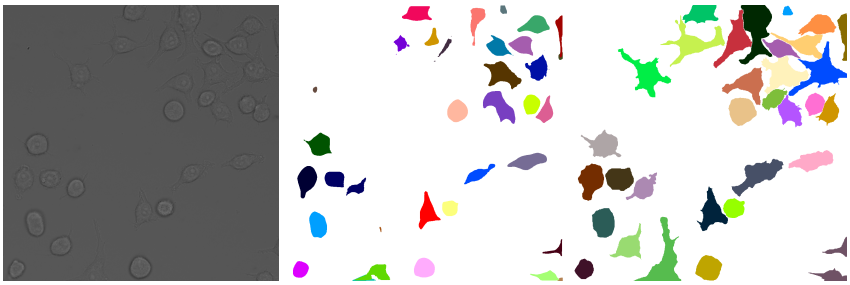
On our test set, our mean and standard deviation for the 3 specified metrics, are $J = 0.57 \pm 0.08$, $J_C = 0.57 \pm 0.10$ and $A_C = 0.85 \pm 0.10$ (Section 4) (Figure 17).

10.4 Conclusion

We have noticed a lot of false negatives, i.e. where the model fails to detect cells. It seems that the difference in appearance between the training data used by Usiigaci and our data have a very big influence and the pre-trained Usiigaci model fails in transferring its learning to our case.

Moreover, the Usiigaci code can be improved in terms of usability. For example, the authors could increase the number of comments to aid understanding

[7] https://github.com/oist/usiigaci



(a) (b) (c)

Figure 17: Example output from the Usiigaci network. From left to right: original microscope image, predicted segmentation and ground truth segmentation, respectively.

and improve the utility tools such that paths to the data or models do not needs to be manually specified.

11 Experiment: Post Processing of CGAN with Distance Transform as Input

11.1 Description

Most of the experiments thus far have been with the numeric masks, where each cell is represented as a value (all pixels of one cell share the same, unique to the cell, value). We hypothesised that if we feed a distance transform to the CGAN, we can achieve better instance segmentation after postprocessing of the output (Section 6 and Figure 2).

11.2 Experimental setup

We used the skimage package for the processing.

- Resizing the distance maps to the original 1152x1152 size.
- High threshold of the distance maps to obtain the centre of the cells.
- Low threshold of the distance maps to obtain the edges.
- Remove artefacts (cells below a certain number of pixels, generally are protrusions that get lost in the generative model).
- Watershed transform on the 'low threshold image' using the 'High threshold' image as the markers/seeds

11.3 Outcomes

While we were only able to carry out preliminary experiments due to time constraints, we believe it constitutes an interesting and potentially promising avenue for future investigation."

12 Future work and research avenues

12.1 Post-Processing the CGAN output

One of the limitations that we found as we tested the results with the CGAN models is that the generated images were in a different scale to the ground truth. Although this is not necessarily a problem, we have realised that instead of having sharp, clean boundaries around each cell, we have a gradient which can be limiting.

12.2 U-Net Borders

Rather than use U-NET to detect areas that are cell or not-cell, it might be advantageous to introduce a third class, specifically the borders of the cells. Converting the input masks to borders is a trivial task but in so doing, the network would create distinct areas, resulting in images similar to those produced by a watershed algorithm. More traditional image processing techniques could then be used.

12.3 Mask R-CNN

We had a look at another repository[8], which was produced by the BIOMAG GROUP. They worked on a similar problem in Kaggle's Data Science Bowl competition. Comparing to other participants they applied the image style transfer method. It is useful when one has a diverse range of microscopy images. This involved the generation augmented training samples using the image style transfer model. They combined this with Mask RCNN, which they used for segmentation, with U-NET for boundary correction and mathematical morphology for post-processing. Their method first rescales the image in order to make nuclear size approximately uniform, since they found out that doing that increase the performance of the model. Then they apply style models and finally they used the mask R-CNN segmentation model trained on rescaled and on augmented data. To get high-pixel accuracy on edges they used a U-NET based model. Details can be found in the paper of Hollandi and co-workers [13] and on the website www.nucleAIzer.org. Future work could be based on a similar approach.

[8] https://github.com/spreka/biomagdsb

Team members

- **Sam Blakeman**: Sam is a final year PhD student at Birkbeck, University of London. He is interested in reconciling machine learning methods with learning mechanisms in the brain, with a particular focus on reinforcement learning. He was the facilitator for this study group and coordinated the activities of the group throughout the week long challenge."
- **Tiejun Wei**: Tiejun is in his third year of PhD at Queen Mary Uni-versity of London and now full-time based at the Alan Turing Institute. Currently, he is leading a project focusing on the structural influence on carotenoid excited state properties, which potentially has a great impact understanding photoprotection. Both Daniel and Tiejun contributed the bulk of the work on the self organizing map and discriminative loss function projects.
- **Daniel Han**: Daniel is a second year PhD student at the University of Manchester. Through the use of mathematical modelling, data analytics and machine learning, he is quantifying the properties of intracellular transport. Tiejun and Daniel worked on the self-organizing map and discriminative loss methods to segment cells from the background.
- **Beatriz Costa-Gomes**: Bea is a final year PhD student at the Univer-sity of Manchester, doing an Enrichment Placement at the Alan Turing Institute. She developed a software for image analysis of Drosophila neuronal cultures, and is at the Turing to analyse the images using Machine Learning approaches. In this challenge, Bea wrote and ran both the preprocessing and the postprocessing experiments, as well as helping with the labelling of the ground truth images for the CGANs. Bea also prepared the final presentation.
- **Louis Mahon**: Louis is a first year PhD student in the Department of Computer Science at the University of Oxford. He researches deep learning, particularly its application to logical reasoning and knowledge extraction from unstructured data. He worked on the Usiigaci implementation of Mask R-CNN.
- **Bogdan Toader**: Bogdan is a postdoctoral research associate at the University of Cambridge, where he works on image deconvolution for microscopy as part of Cambridge Advanced Imaging Centre (CAIC). At the Data Study Group he worked on applying the Mask R-CNN model.
- **Yuki Asano**: Yuki is a PhD student in the Visual Geometry Group (VGG) working with Andrea Vedaldi and Christian Rupprecht at the University of Oxford. Prior to this he studied physics at the University of Munich (LMU) and Economics in Hagen as well as a MSc in Mathematical Modelling and Scientific Computing at the Mathematical Institute in Oxford.
- **Alessandro Ragano**: Alessandro is a PhD student at University College Dublin where he works on speech and audio quality prediction. At the DSG he worked with Cono on developing Pix2Pix model and on collecting the final results. Also, he contributed to the dataset annotation.

- **Natalia Garcia Martin**: Natalia is a second year PhD student at the University of Oxford working on Statistics and Machine Learning applications for Cancer Stratification by integrating genomics and imaging data. At the DSG, she worked on the U-Net Keras implementation.

- **Cono Di Paola**: Cono is a theoretical chemist and material scientist now working with his former group at King's College London as freelance/scientific collaborator. It has been a while now that he has been also involved in data science/ML projects. Cono has contributed with Alessandro to set the pix2pix/CGAN model up and run the model training and the prediction on the test data set on the GPU environment provided by the Alan Turing Institute.

- **Bekzhan Kerimkulov**: Bekzhan is a second year PhD student at University of Edinburgh. He studies stochastic optimal control problems and related numerics. He contributed to this report by providing a review on solutions to Kaggle's Data Science Bowl 2018 competition and an introduction to the Mask R-CNN.

- **Jeremy Pike**: Jeremy is a postdoctoral researcher and staff scientist at the University of Birmingham where he specialises in image processing and analysis for applications in light microscopy. Jeremy acted as PI for this DSG but all credit should go to the participants who drove the project themselves.

References

[1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[2] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[3] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[4] E. Moen, D. Bannon, T. Kudo, W. Graf, M. Covert, and D. Van Valen, "Deep learning for cellular image analysis," *Nature Methods*, vol. 16, no. 12, pp. 1233–1246, 2019.

[5] 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.

[6] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow." `https://github.com/matterport/Mask_RCNN`, 2017.

[7] M. Maška, V. Ulman, D. Svoboda, P. Matula, P. Matula, C. Ederra, A. Urbiola, T. España, S. Venkatesan, D. M. Balak, *et al.*, "A benchmark for comparison of cell tracking algorithms," *Bioinformatics*, vol. 30, no. 11, pp. 1609–1617, 2014.

[8] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004.

[9] C. Payer, D. Štern, T. Neff, H. Bischof, and M. Urschler, "Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018 - 21st International Conference, 2018, Proceedings*, Lecture Notes in Computer Science, pp. 3–11, Springer Verlag Heidelberg, 9 2018.

[10] B. D. Brabandere, D. Neven, and L. V. Gool, "Semantic instance segmentation with a discriminative loss function," *CoRR*, vol. abs/1708.02551, 2017.

[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 740–755, Springer International Publishing, 2014.

[12] H.-F. Tsai, J. Gajda, T. F. Sloan, A. Rares, and A. Q. Shen, "Usiigaci: Instance-aware cell tracking in stain-free phase contrast microscopy enabled by machine learning," *SoftwareX*, vol. 9, pp. 230 – 237, 2019.

[13] R. Hollandi, A. Szkalisity, T. Toth, E. Tasnadi, C. Molnar, B. Mathe, I. Grexa, J. Molnar, A. Balind, M. Gorbe, M. Kovacs, E. Migh, A. Goodman, T. Balassa, K. Koos, W. Wang, N. Bara, F. Kovacs, L. Paavolainen, T. Danka, A. Kriston, A. E. Carpenter, K. Smith, and P. Horvath, "A deep learning framework for nucleus segmentation using image style transfer," *bioRxiv*, 2019.

turing.ac.uk
@turinginst