Cell Detection for 2-D Microscopy Images with U-Net based CNNs

Gianluca Bencomo GB5435@PRINCETON.EDU
Department of Computer Science
Princeton University
Princeton, NJ 08544, USA

John Yang JY1682@PRINCETON.EDU
Department of Computer Science
Princeton University
Princeton, NJ 08544, USA

Abstract

Accurately counting the number of cells present in 2-D microscopy images is a fundamental yet challenging task required in biological experiments and many medical diagnostic processes. Present methodologies for cell counting use either a hemacytometer (2-D microscopy image) or a Coulter counter (electronic counter). While a hemacytometer is tedious, time-consuming, and prone to subjective errors, live-dead discrimination is unreliable with electronic counters. Automatic counting methods for 2-D microscopy are preferred, but challenged by low contrasts, complex backgrounds, large cell count and cell shape variance, and heavy cell occlusion in their images. Herein, we adapt the segmentation network, U-net, to perform cell localization and counting for 2-D microscopy images across a variety of cell types. We evaluate the method on four public cell counting benchmarks - synthetic fluorescence microscopy (VGG) dataset, Modified Bone Marrow (MBM) dataset, human subcutaneous adipose tissue (ADI) dataset, and Dublin Cell Counting (DCC) dataset. Across all four datasets, we report performance comparable to the current state-of-the-art. The source code is available at `https://github.com/gbencomo/JYGB`.

Keywords: Cell counting, neural networks, microscopy images

1. Introduction

In cell and molecular biology, experiments require cell counts to determine cell concentrations in samples, study growth rates, and measure cell viability [1]. In medicine, the number of cells in a microscopy image can determine a disease diagnosis [2], help differentiate tumor types [3], assist in understanding cellular and molecular genetic mechanisms [4, 5], and provide useful information to many other applications [6, 7]. Manual cell counting is the most reliable method, but is tedious, time-consuming, prone to subjective errors, and not suitable for the high-throughput processes of biomedical applications. In this paper, we demonstrate the capacity for an adapted U-Net to be trained with a small dataset and achieve good performance on unseen samples.

Early attempts in the space of cell counting and cell detection utilized the concavity-based algorithm [8] and the distance transform-based algorithm [9]. Both of these algorithms, however, cannot handle irregular cells and are not robust to the irregularities that occur within 2-D microscopy images. Now, automated cell counting attempts are dominated by supervised learning approaches that perform counting without explicit image processing.

This involves mapping image features to a predicted density map, and casting the counting problem as a regression problem. Microscopy images are dot-annotated, meaning that a single pixel is used to represent cells and the center of the cell is set to 1 otherwise 0 (see Figure 1). Integrating over a density map, or any region in the density map, gives a cell count for that image or region.

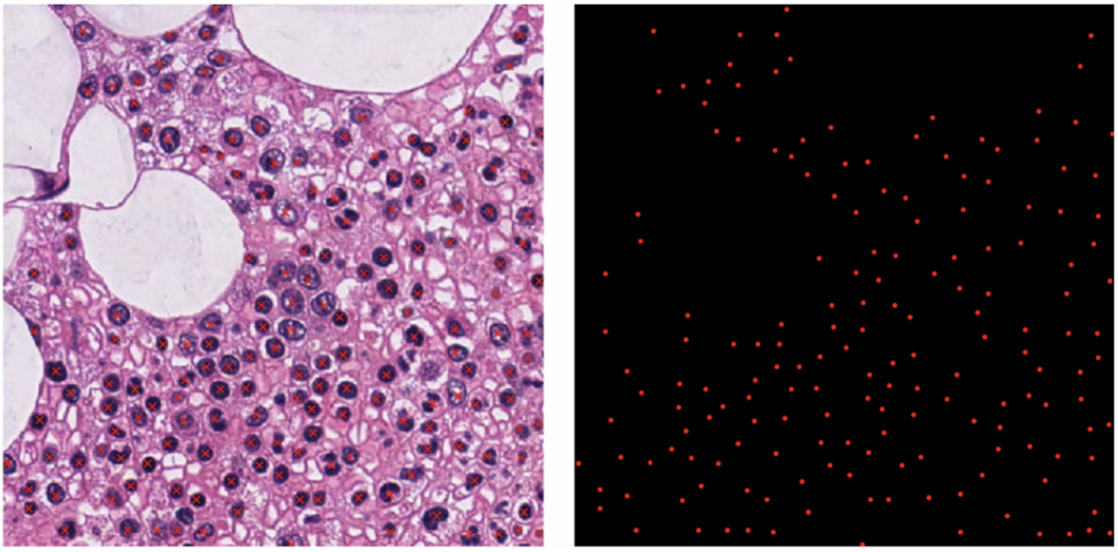

Figure 1: A sample image from the Modified Bone Marrow (MBM) dataset [19] with dot annotations shown as red cross overlays (left image) and the corresponding density map used as a label in training (right image).

Machine learning models that require explicitly-defined features as parameters, like support vector machine (SVM), have struggled to perform this counting task due to the incredible amount of variance between images. This has popularized the non-specificity of the CNN-based model, where the many convolutional layers can automatically and adaptively extract semantic features in an extremely high-dimensional space. The CNN-based model has proven to be robust against the various image acquisition techniques, low image contrast, complex tissue backgrounds, large variations in cell sizes, shapes and counts, and significant inter-cell occlusions 2-D microscopy images.

Following the CNN-based approach with density maps, we used the encoder–decoder network architecture of the U-Net to map images features to pixel-wise probabilities of a cell's presence. We explored adaptations to the network's architecture, such as an atrous spatial pyramid pooling (ASPP) module prior to the upsampling layers, and experimented with different values and methods for loss functions, optimizers, and learning rate schedulers. We validate the effectiveness of this method on four publicly available benchmarks, and receive results competitive with the existing state-of-the-art.

The rest of this report is organized as follows. Section **2** reviews the most important objects counting methods of the past several decades, especially for cell counting. Section **3** discusses the network architecture and implementation in detail. A description of our four

datasets is included in Section **4**. Our results are discussed in Section **5**. Section **6** draws a conclusion to this report.

2. Related Work

Cell counting methods can generally fall into one of two categories: detection-based and regression-based. Detection-based approaches typically segment objects from the background image and count the objects, or the level set [10, 11]. These segmentations are robust for individual cells with clear edges, but struggle separating overlapping cells that are low-contrast or edge-blurred, often classifying them as a single cell. The distance-transform and concave points on the edge where objects overlap have both been used to count cells in detection-based approaches [8, 11].

Early regression-based approaches learned a mapping from dense local image features to density maps. Fiaschi et al. performed cell counting with patch-wise prediction and structured labels [12]. Lempitsky and Zisserman used linear regression with dense SIFT features to predict the density map [13]. Arteta et al. built upon their work by utilizing the same SIFT features but within a tree structure of candidate cell-like regions. Through SVM, these cell-like regions were mapped accordingly and an interactive counting system [14] also based on Lempitsky's previous work was applied.

More recently, Xie et al. used a CNN-based approach to learn the mapping between images and filtered density maps [15]. Due to the breakthrough success of this network, research has since followed this direction. Zhu et al. built a network with a similar construction but, instead of integrating the density map to determine cell counts, the non-maximum suppression (NMS) algorithm was used to detect cells on the density map [16]. Pan et al. utilized receptive fields of varying sizes at the first convolution layer to obtain more local information, and consider cell types with high variance in cell size [17]. Walach et al. boosted their CNN-based approach by allowing each layer to learn the bias between the ground truth and the prediction of the previous layer [18]. Cohen et al. found success by predicting cell counts with density maps built from the receptive fields of the networks [19]. This represents a redundant counting approach based on patches, but is extremely non-intuitive. Their method involved first filtering the dot annotations with a square kernel and then summing the value in a sliding window fashion. Each window corresponds to a receptive field in the network, and the redundancies improved accuracy but also increased the model's tendency to over-fitting. Realizing the importance of image features at varying levels of abstraction, Rad et al. proposed a novel pyramid network to extract more global information [20]. Density-based models offer superior counting performance with crowded cells and segmentation-based models perform better with cells containing distinct edges. Thus, Akram et al. combined two CNNs to perform both, detection with density maps and cell segmentation [21].

Notable work relevant to cell counting has also occurred in the general object counting space. Heatmaps, HOG-based head detectors, and CNN-based regressors have all been used to count people [22]. Zhang et al. used Long Short-Term Memory (LSTM) networks to count vehicles [23]. CNN-based models have been used to effectively count anything from penguins to consumer products. The object counting space has also produced breakthrough work in concepts such as transfer learning, feature extraction, and learning without

forgetting (LwF). Bilen et al. trained a CNN model to learn a universal vision representation that has the ability to achieve strong performance on an array of unrelated tasks by including domain-specific scaling and normalization layers throughout the network that, in a sense, direct the network to the task at hand [24]. Rebuffi et al. extended this work, increasing its robustness, by proposing a sequential training procedure that allows the CNN to learn new tasks over time without discarding any of knowledge gained from previous functions [25].

3. Method

In this section, we describe the core components of our U-Net implementation. All code was written in python and heavily utilized the numpy and pytorch libraries. The full implementation is available at `https://github.com/gbencomo/JYGB`.

3.1 Overview

We begin with a description of the more salient mathematical aspects of the density regression-based counting process implemented in this study. For any given two-dimensional microscopy image $X \in \mathbb{R}^{M \times N}$ with N_c cells, the corresponding density map can be represented as $Y \in \mathbb{R}^{M \times N}$. $Y_{i,j} \in \{0, 1\}$ for any $i \in M$ and $j \in N$, thus representing the presence of a cell at each corresponding pixel of X. We denote $\phi(X)$ as a feature map of X and define the density regression function $F_\phi(\phi(X); \mathbf{w})$ as a mapping from $X \to Y$:

$$Y = F_\phi(\phi(X); \mathbf{w}),$$

where vector $\mathbf{w}$ parameterizes F_ϕ. Since the dot annotations shown in Figure 1 are sparse, making the regression problem more difficult, we follow previous works [15, 19] and take a normalized Gaussian function G to smooth the dot annotations. The density maps used for regression are defined as:

$$g(X) = \sum_{i=1}^{N} G(x_i, \sigma),$$

where x_i is the i^{th} cell dot label, σ is the standard deviation of the 2-D Gaussian, and $G(x_i, \sigma)$ represents the normalized dot annotations at every cell in the density map. For simplicity and continuity, we used a kernel bandwidth of 3 and $\sigma = 1$ to smooth the dot annotations across every image of all four datasets used in this study. All smoothing was done prior to any pre-processing and data augmentation. With smoothing in mind, we can arrive at our final mapping from $X \to Y$:

$$Y = F_\phi(\phi(g(X)); \mathbf{w}),$$

where Y is a continuous density map, rather than a discrete representation. We can compute the number of cells in an image, or region of an image, by:

$$N_c = \sum_{i=1}^{M} \sum_{j=1}^{N} Y_{i,j} = \sum_{i=1}^{M} \sum_{j=1}^{N} [F_\phi(\phi(g(X)); \mathbf{w})]_{i,j}.$$

In our U-Net implementation, we attempt to learn $F_\phi(\phi(g(X); \mathbf{w})$ and the corresponding $\mathbf{w}$. Since the U-Net is fully convolutional, and generates our feature map, we are attempting to learn the target function $F(g(X); \mathbf{w})$.

3.2 Network Architecture

The baseline model used in this study is a U-Net, which is a encoder–decoder-style neural network consisting of 26 hidden layers (see Figure 2) [26]. This CNN features a contracting path and an expanding path. During the contracting path, the encoder takes an image tile and computes successive feature maps at multiple scales. This yields a feature representation that is multi-resolution and has multiple levels of abstraction. During the expanding path, the decoder using this feature representation to assign class probabilities to each pixel at every level of abstraction. In U-Nets designed for segmentation, the decoder gradually synthesizes the segmentation by updating the class probabilities for each pixel at each layer. It starts with low-resolution feature maps, corresponding to large structures, and finishes with the full-resolution feature maps, representing structures that could be as small as a single pixel. In the adapted U-Net used in this study, each pixel is given a single class probability.

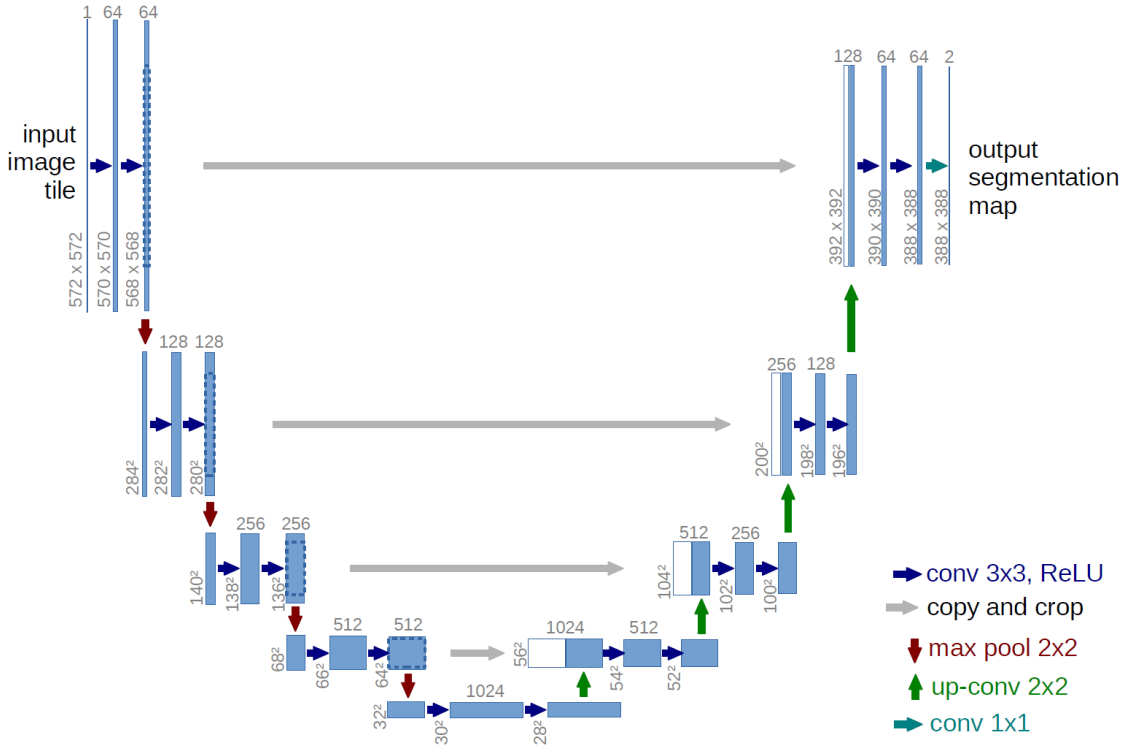

Figure 2: An example image passing through the U-Net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. Figure shown is not original, and was pulled directly from the U-Net paper [26].

The encoder performs two 3x3 convolutions (without padding), each followed by batch normalization and a rectified linear unit (ReLU). After every two convolutions, a 2x2 max-

pooling operation with stride two is performed, effectively halving the resolution of the resulting feature map. The network only uses the valid part of each convolution.

The decoder repeatedly up-samples with a scale factor of two, which halves the number of feature channels. This up-sampling, or up-convolution, ends with a concatenation with the cropped encoder feature map for each corresponding resolution and two convolutions, each followed by 2-D batch normalization and ReLUs.

The final layer is a 1×1 convolution that maps feature vectors to the probability of a cell's presence at each pixel.

A variation of the U-Net model was also tested that included an atrous spatial pyramid pooling (ASPP) layer to make a 27 layer model. This model placed this ASPP directly before the first upsampling layer.

ASPP consists of several parallel atrous convolutions with different rates [27]. It is a combination of atrous convolution and spatial pyramid pooling, and it can capture the contextual information at multiple scales (see Figure 3). The ASPP module used consists of one 1×1 convolution and three parallel 3×3 convolutions with rates of 6, 12, and 18, respectively as well as an image-level feature that is produced by global average pooling. The U-Net has shown an extraordinary ability to implicitly represent scale, and account for information at many levels of abstraction. However, by explicitly accounting for object scale and information at several levels of abstraction, we hope to improve the U-Net's ability to successfully handle a wide range of objects. We report on the performance of both the baseline adapted U-Net structure and the U-Net + ASPP.

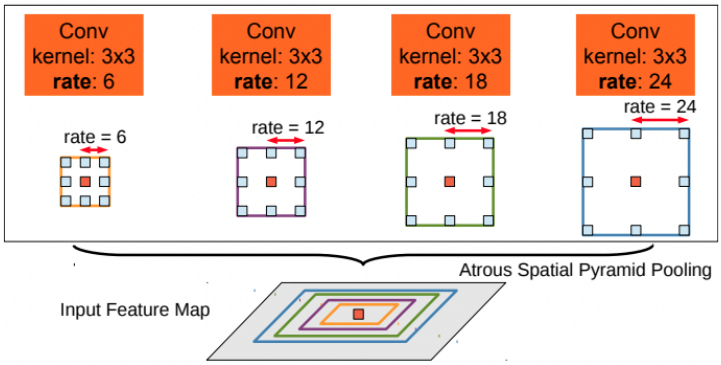

Figure 3: Atrous Spatial Pyramid Pooling (ASPP) example. The center pixel (orange) is classified by using multi-scale features with varying effective Field-Of-Views. Figure shown is not original, and was pulled directly from [27].

3.3 Implementation Details

3.3.1 Data Augmentation

During training, we randomly crop 87.5% region of the images and apply random horizontal flipping and vertical flipping. We utilize data augmentation to add disruptions to the distributions that can prevent over-fitting and improve model robustness. This is crucial with small datasets [26].

3.3.2 Pre-processing

The U-Net architecture implemented can work with raw images without sophisticated pre-processing. Regardless, we perform a few essential operations. Since the DCC dataset has varying image sizes, we downsample every image and density map to 256×256, preserving the mapping between input images and dot annotations by linearly scaling the density map. We also zero-pad the image edge from 150×150 to 152×152 in ADI dataset, giving more convenience to the max-pooling operations of the network. The VGG and MBM datasets did not receive modifications to the raw image inputs. This follows the pre-processing pipeline of previous works [15, 19].

Prior to training and testing, the datasets receive a random train-test split corresponding to the numbers featured in Table 1. Both the training and test sets are zero-centered by subtracting the mean of the training set and normalized by dividing by the standard deviation of the training set. Data augmentation is only performed on the training set.

3.3.3 Optimization

To minimize the loss function, we made attempts with three optimizers: stochastic gradient descent (SGD), RMSprop, and the Adam optimizer with decoupled weight decay. This variation of Adam explicitly enforces weight decay instead of implemented as L_2 regularization [28]. For all three optimizers, an initial learning rate of 0.001 was used, and a weight decay of 0.001. Momentum was set to 0.9 for SGD. At the start of training, the weights of the network are randomly initialized by the Glorot method. We always trained with 300 epochs and saved the well-trained parameters.

3.3.4 Loss Functions

The mean square error (MSE) is the colloquial loss function for regression tasks and it measures the Euclidean distance between predicted labels and ground truth labels. In our implementation, it is defined as:

$$L(Y, F(X)) = \frac{1}{N} \sum_{i=1}^{N} \|Y_i - F(X_i)\|_2^2$$

where Y_i and $F(X_i)$ respectively are the ground truth density map and estimated density map of the input image X_i, N is the number of training/test images.

We constructed our U-Net implementation with MSE loss as well as Huber loss, which is defined as:

$$L_\delta(Y, F(X)) = \begin{cases} \frac{1}{N} \sum_{i=1}^{N} (Y_i - F(X_i))^2 & \text{for } |Y_i - F(X_i)| \leq \delta, \\ \frac{1}{N} \sum_{i=1}^{N} \delta \left(|Y_i - F(X_i)| - \frac{1}{2}\delta \right), & \text{otherwise.} \end{cases}$$

where δ defines a threshold to switch from mean absolute error (MAE) to MSE. The idea is to balance MAE's superior handling of outliers with MSE's computational savings. In our experiments, we test the performance of both of these loss functions.

3.3.5 Learning Rate Scheduling

We tested two learning rate schedules in our experiments. The first learning rate schedule was standard: every 30 epochs the learning rate was multiplied by a factor of 0.5. The second learning rate we tested was cosine annealing with warm restarts (see Figure 4). This method, in theory, prevents the learner from getting stuck in a local loss minima with periods of fast learning but also allows the learner to converge to a near-true-optimal point during extended periods of slow learning. We performed the first restart after 10 epochs, and doubled the time until the next restart after each restart.

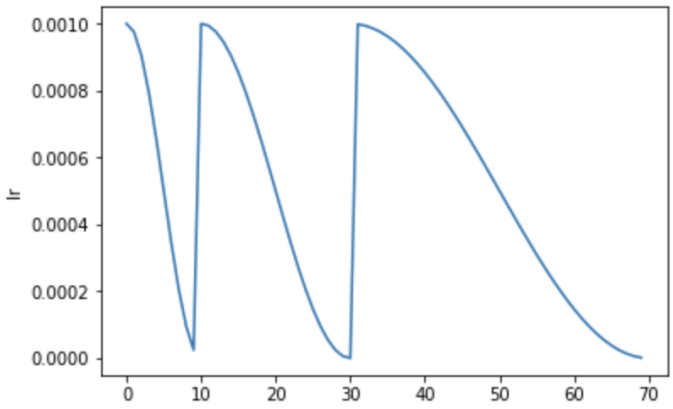

Figure 4: Cosine annealing with restarts as used in U-net implementation.

4. Datasets

Four microscopy image datasets were considered in this study, which include synthetic images of bacterial cells (VGG) [13], experimental images of bone marrow cells (MBM) [19], human subcutaneous adipose tissue cells (ADI) [19], and the Dublin Cell Counting (DCC) dataset [30], which features microscopy images from a wide array of tissues and cell species. Table 1 illustrates the data details. Sample images from the four datasets are shown in Figure 5.

Dataset	**VGG**	**MBM**	**ADI**	**DCC**
Image size	256 x 256	600 x 600	150 x 150	varied
Train (#) / Total (#)	64 / 200	15 / 44	50 / 200	100 / 176
Cell Count	174 ± 64	126 ± 33	165 ± 44	34 ± 22
Type	Synthetic	Real	Real	Real

Table 1: Dataset statistics for the four datasets employed in this study. Image size is represented by pixel, and count statistics is represented by mean and standard variations of cell numbers in all the images in each dataset.

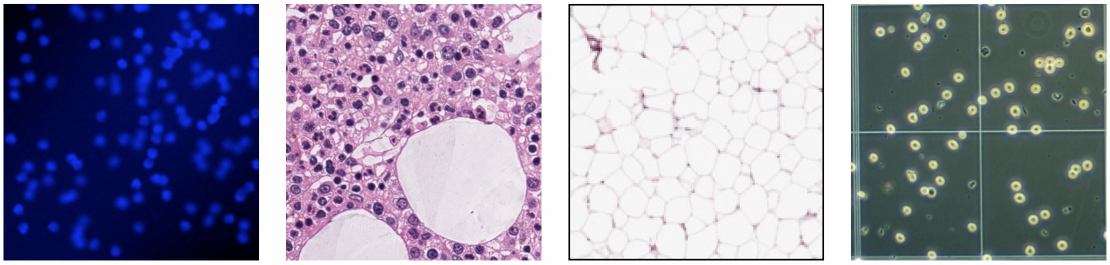

Figure 5: Example images of the four datasets used in this study. From left to right: synthetic fluorescence microscopy (VGG) dataset, Modified Bone Marrow (MBM) dataset, human subcutaneous adipose tissue (ADI) dataset and Dublin Cell Counting (DCC) dataset.

4.1 Synthetic fluorescence microscopy (VGG)

Lempitsky and Zisserman [13] synthetically generated highly-realistic emulations of fluorescence microscopic images of bacterial cells. These synthetic images are similar to the real cell images in that they feature cell overlapping, defocus, and intensity variability. Cell images synthesized with varying focal lengths. Reference Table 1 for dataset statistics.

4.2 Modified Bone Marrow (MBM)

Cohen et al. [19] adapted the bone marrow (BM) dataset introduced by Kainz et al. [29] to correct several wrong annotations of foreground and ambiguous objects. Through cropping, Cohen et al. converted the original 11 image dataset into 44 images. These images are full of staining impurities, which makes the counting task significantly more difficult. Reference Table 1 for dataset statistics.

4.3 Human subcutaneous adipose tissue (ADI)

This dataset was constructed from the Genotype Tissue Expression Consortium [31] with densely packed adipocyte cells. 200 regions of interest were sampled from high resolution histology slides by using a sliding window of 1700 x 1700. Images were then down sampled to 150 x 150, representing a suitable scale in which cells could be counted using a 32×32 receptive field. Adipocytes can vary in size dramatically (20-200µ) [31] and they are densely packed cells due to their gap junction connections. This represents an extremely difficult test-case for automated cell counting procedures. Reference Table 1 for dataset statistics.

4.4 Dublin Cell Counting (DCC)

This dataset consists of a wide array of varying tissues and cell species. Examples include stem cells derived from embryonic mice, isolated human lung adenocarcinoma and examples of primary human monocytes isolated from a healthy human volunteer. Throughout the dataset, there are significant variations in cell density, cell morphology, and cell size. The objective lens used during imaging was also varied and the diameter of the diaphragm was variably adjusted to change the amount of light that hits the sample. Finally, the hemacytometer saw varying grid sizes and configurations. This dataset was built to demonstrate

all of the variability that can arise in microscopy and represents one of the most difficult object counting tasks. Reference Table 1 for dataset statistics.

5. Results

To explore and validate the proposed model, we first performed an ablation study with the various interchangeable features to determine the optimal set up. We then ran comparisons between the optimal set up and the highest performing results published in literature. Each model reported was tested exactly 5 times, with a randomized training-test split. As previously mentioned, every training session included 300 epochs. Optimization was performed in mini-batches, and we report results received with a batch size of 16. Since the MBM dataset included only 15 samples in the training set, every epoch had a single batch of 15 samples. Across the varying image sizes between the four datasets, and varying model configurations, training times ranged from 4 - 8 hours on a standard CPU. With test images, predictions are made in roughly 3 seconds. We only show plots for one training run (Figure 6), since we wanted to keep this report concise and found the density maps (Figures 7 - 10) and tables significantly more descriptive.

5.1 Evaluation metric

Following previous works [10 - 23], the mean absolute error was the evaluation metric in this study. It is defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|P_i - T_i|,$$

where N is the number of test images, P_i is the predicted number of cells for image i, and T_i is the true number of cells for image i. The MAE is calculated for every image not included in the training set of an experiment.

5.2 Ablation study

To optimize the model, we tested the following configurations on the VGG cell dataset:

- U-Net baseline: Adapted U-Net with SGD as the optimizer, standard learning rate scheduling, and MSE loss.
- U-Net + AdamW: Adapted U-Net with AdamW as the optimizer, standard learning rate scheduling, and MSE loss.
- U-Net + RMSprop: Adapted U-Net with RMSprop as the optimizer, standard learning rate scheduling, and MSE loss
- U-Net + AdamW + CA: Adapted U-Net with AdamW as the optimizer, cosine annealing with warm restarts as the learning rate schedule, and MSE loss
- U-Net + AdamW + CA + Huber: Adapted U-Net with AdamW as the optimizer, cosine annealing with warm restarts as the learning rate schedule, and Huber loss

- U-Net + AdamW + CA + ASPP: Adapted U-Net with AdamW as the optimizer, cosine annealing with warm restarts as the learning rate schedule, and the ASPP module included

Method	MAE	MSE
U-Net baseline	2.4 ± 0.3	9.8 ± 1.9
U-Net + AdamW	2.3 ± 0.3	9.7 ± 1.9
U-Net + RMSprop	2.4 ± 0.3	10.1 ± 2.2
U-Net + AdamW + CA	2.3 ± 0.3	9.4 ± 1.3
U-Net + AdamW + CA + Huber	3.1 ± 0.4	12.2 ± 2.3
U-Net + AdamW + CA + ASPP	2.2 ± 0.2	7.9 ± 0.9

Table 2: The results of the ablation study on VGG cells

We found that AdamW offered the fastest convergence and performed better at finding minima than the other optimizers. Cosine annealing with warm restarts also offers a very small performance boost. MSE significantly outperformed the Huber loss, and the downside of Huber loss was exasperated by slightly longer run times. We also found that the ASPP module offers a performance boost to the U-Net. This is likely due to the additional levels of abstraction added to the feature map, and image features that are not accounted for by the native U-Net architecture.

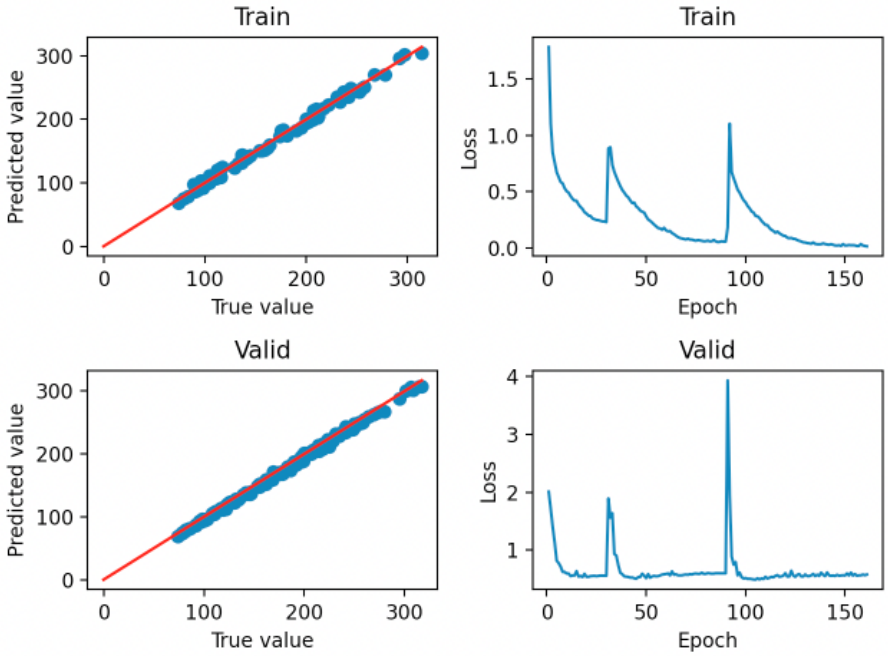

Figure 6: U-Net + AdamW + CA + ASPP over 160 epochs (VGG).

5.3 Comparison with the state-of-the-art

5.3.1 Synthetic fluorescence microscopy (VGG)

The VGG cell dataset represents the most commonly used dataset to benchmark cell counting methods. As shown in Table 3, the method implemented outperforms many previously published results, and performs close to the state-of-the-art. For all experiments with the VGG cell dataset, we randomly sampled 64 out of the 200 images as a training. We performed data augmentation and pre-processing on those 64 samples, and used the remaining 136 images as a test set. Using a test set that was significantly larger than the training set was deliberate. In settings in which cell counting is used, biologists or medical experiments would ideally need to annotate as few images as possible to generate an automatic counting system for their tissue or cell culture.

Method	**MAE**	$\mathbf{N}_{train}$
ResNet-152 (R), Xue et al. [29]	7.5 ± 2.2	100
GMN, Lu et al. [33]	3.6 ± 0.3	32
FCRN-A, Xie et al. [15]	2.9 ± 0.2	50
Count-Ception, Cohen et al. [19]	2.3 ± 0.4	50
Cell-Net, Rad et al. [20]	2.2 ± 0.5	50
MM-Net, Jiang et al. [34]	2.1 ± 0.1	50
U-Net + AdamW + CA + ASPP	2.2 ± 0.2	64

Table 3: The comparison results on VGG Cells dataset with state-of-the-art works.

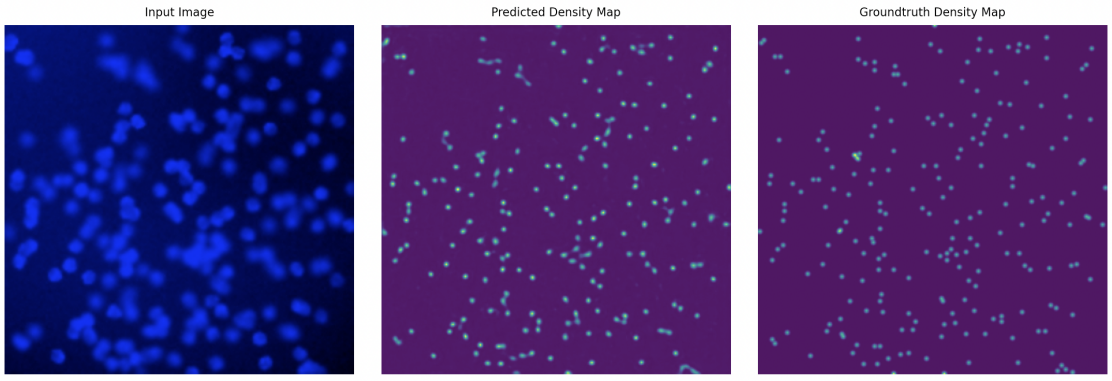

Figure 7: Example of a predicted density map (middle) based on the input image (left). The groundtruth density map is shown on the right. Predicted cell count: 222.306. True cell count: 223.0.

5.3.2 Modified Bone Marrow (MBM)

Our results with the MBM cell dataset produced similar outcomes as with the VGG cell dataset. For every experiment, we randomly sampled 14 images as a training test and 29 images as a test set, and performed data augmentation exclusively on the training set. With only 15 training images, we were able to localize and count cells with an impressive accuracy considering all of the impurities and variance in this dataset. The visualization of the predicted density map (Figure 8) demonstrates how the impurities and variance create uncertainty. While the model was able to localize cells, it was not able to localize cell centers. Much of the landscape within the predicted density map carries a probability of a cell at each pixel that is non-zero. This means the predictions are weaker than other datasets in this study, but after integrating the density map we still receive relatively good prediction. The method implemented outperforms many previously published results, but is significantly outperformed by the state-of-the-art (Table 4).

Method	**MAE**	$\mathbf{N}_{train}$
FCRN-A, Xie et al. [15]	21.3 ± 9.4	15
Marsden et al. [30]	20.5 ± 3.5	15
Count-Ception, Cohen et al. [19]	8.8 ± 2.3	15
Cell-Net, Rad et al. [20]	9.8 ± 3.2	10
MM-Net, Jiang et al [34]	7.5 ± 0.7	15
U-Net + AdamW + CA + ASPP	8.4 ± 0.8	15

Table 4: The comparison results on MBM Cells dataset with state-of-the-art works.

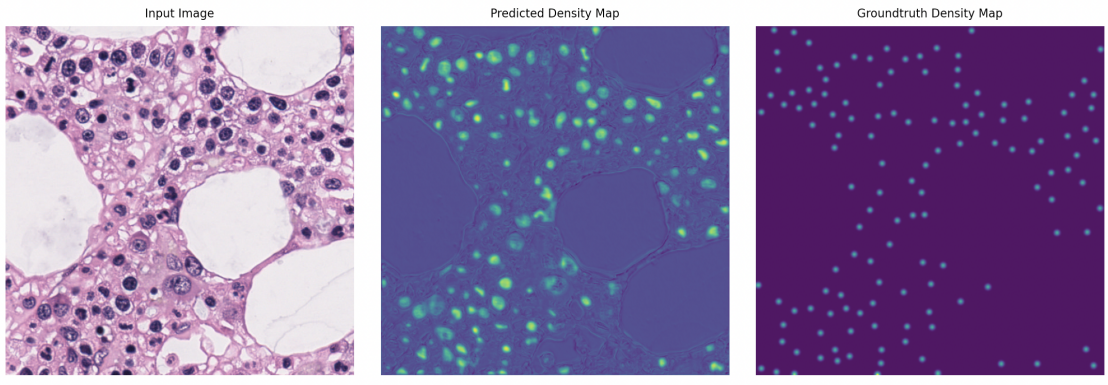

Figure 8: Example of a predicted density map (middle) based on the input image (left). The groundtruth density map is shown on the right. Predicted cell count: 177.934. True cell count: 180.0.

5.3.3 Human subcutaneous adipose tissue (ADI)

The ADI cell dataset represents a less commonly used dataset to benchmark cell counting methods, but previous results have been reported. We trained on 50 random, augmented images and tested on the remaining 150 images. The method implemented in this study was able to significantly outperform an older method (Table 5). In Figure 9, the predicted density map makes both strong and weak predictions for cell centers, and we found that predictions got weaker the father a cell was from average cell shape and average cell size. Since cell shape and cell size follow a normal distribution [35], we still receive decent predictions for cell counts after we integrate the density map.

Method	**MAE**	$\mathbf{N}_{train}$
Count-Ception, Cohen et al. [19]	19.4 ± 2.2	50
U-Net + AdamW + CA + ASPP	12.2 ± 1.4	50

Table 5: The comparison results on ADI Cells dataset with state-of-the-art works.

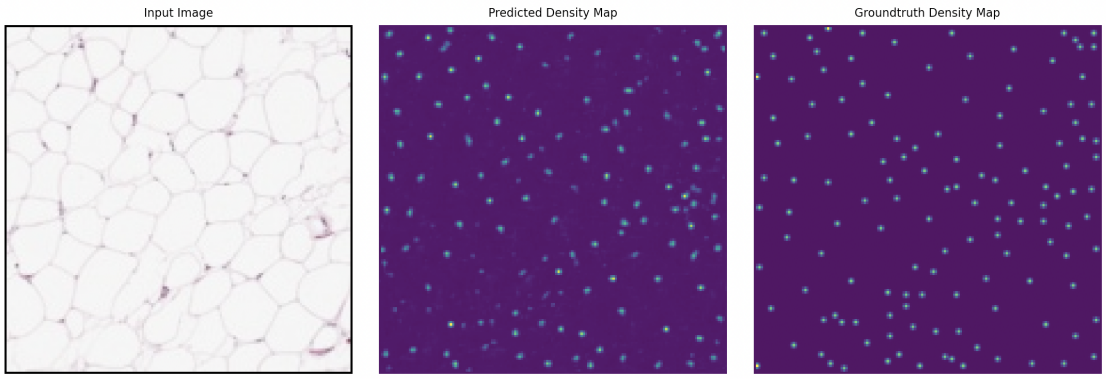

Figure 9: Example of a predicted density map (middle) based on the input image (left). The groundtruth density map is shown on the right. Predicted cell count: 128.678. True cell count: 132.0.

5.3.4 Dublin Cell Counting (DCC)

The DCC cell dataset is another less commonly used dataset to benchmark cell counting methods, but carries a dataset diversity ideal for measuring the generalization of a method. Following the procedures of Marsden et al. [30], we trained with 100 images and tested with 76 images, randomly sampled for each experiment. We performed data augmentation exclusively on the training set. The method implemented in this study was able to significantly outperform an older method (Table 5) and match groundtruth density maps almost perfectly (see Figure 10). It is important to note that these experiments had the largest training set sizes, to which we attribute the high accuracy predictions. The DCC cell dataset suffers from staining impurities, occlusion, and other sources of prediction error

native to the ADI and MBM. Additionally, the DCC cell data was complied using different microscopes and cell types. We hypothesized that performance would be the worst on this dataset, but increasing the number of training examples seemed to bolster predictions despite added variance in this CNN architecture.

Method	**MAE**	$\mathbf{N}_{train}$
Marsden et al. [30]	8.4*	100
U-Net + AdamW + CA + ASPP	2.9 ± 0.4	100

Table 6: The comparison results on DCC Cells dataset with state-of-the-art works.

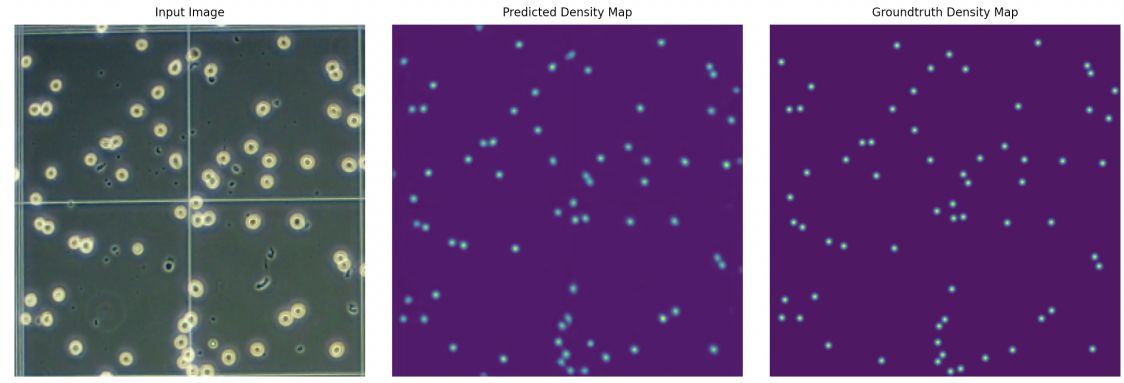

Figure 10: Example of a predicted density map (middle) based on the input image (left). The groundtruth density map is shown on the right. Predicted cell count: 66.145. True cell count: 65.0.

6. Conclusion

In conclusion, we reported on a U-Net based architecture to perform cell counting of 2-D microscopy images. By adding the ASPP module, we effectively increased the capacity of the network to capture various details and account for more spatial features. This, in turn, had a positive impact on the network's ability to localize cells in the density map, and count these cells once integrated. The MSE loss function, Adam with decoupled weight decay optimizer, and cosine annealing learning schedule all helped the network converge faster, and to a higher accuracy input-output mapping. After experimenting on four datasets, we can conclude that this method works well for cell counting. Since this network requires only require 50 - 100 annotated samples to train, it can easily be implemented in biomedical applications.

Future work should focus on encoder–decoder network architectures, attempting to swap out parts to increase the quality of predictions. Jointly-trained models based on adversarial learning could take advantage of data from a wide range of data sources in order to bolster predictions, and further improve models.

Acknowledgments

We would like to acknowledge and thank Dr. Olga Russakovsky and Danqi Liao for their project mentorship as well as the computer vision curriculum at Princeton University.

References

[1] Phelan, Mary C., and Gretchen Lawler. "Cell counting." Current Protocols in Cytometry 1 (1997): A-3A.

[2] Venkatalakshmi, B., and K. Thilagavathi. "Automatic red blood cell counting using hough transform." 2013 IEEE Conference on Information & Communication Technologies. IEEE, 2013.

[3] Coates, Alan S., et al. "Tailoring therapies—improving the management of early breast cancer: St Gallen International Expert Consensus on the Primary Therapy of Early Breast Cancer 2015." Annals of oncology 26.8 (2015): 1533-1546.

[4] Solnica-Krezel, Lilianna. "Conserved patterns of cell movements during vertebrate gastrulation." Current biology 15.6 (2005): R213-R228.

[5] Zhang, Su-Chun, et al. "In vitro differentiation of transplantable neural precursors from human embryonic stem cells." Nature biotechnology 19.12 (2001): 1129-1133.

[6] Thomson, James A., et al. "Embryonic stem cell lines derived from human blastocysts." science 282.5391 (1998): 1145-1147.

[7] Lagutin, Oleg V., et al. "Six3 repression of Wnt signaling in the anterior neuroectoderm is essential for vertebrate forebrain development." Genes & development 17.3 (2003): 368-379.

[8] Kothari, Sonal, Qaiser Chaudry, and May D. Wang. "Automated cell counting and cluster segmentation using concavity detection and ellipse fitting techniques." 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro. IEEE, 2009.

[9] Zhang, Chao, et al. "Segmentation of clustered nuclei based on curvature weighting." Proceedings of the 27th Conference on Image and Vision Computing New Zealand. 2012.

[10] Chan, Tony, and Luminita Vese. "An active contour model without edges." International Conference on Scale-Space Theories in Computer Vision. Springer, Berlin, Heidelberg, 1999.

[11] Xu, Chenyang, and Jerry L. Prince. "Snakes, shapes, and gradient vector flow." IEEE Transactions on image processing 7.3 (1998): 359-369.

[12] Fiaschi, Luca, et al. "Learning to count with regression forest and structured labels." Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012). IEEE, 2012.

[13] Lempitsky, Victor, and Andrew Zisserman. "Learning to count objects in images." Advances in neural information processing systems 23 (2010): 1324-1332.

[14] Arteta, Carlos, et al. "Detecting overlapping instances in microscopy images using extremal region trees." Medical image analysis 27 (2016): 3-16.

[15] Xie, Weidi, J. Alison Noble, and Andrew Zisserman. "Microscopy cell counting and detection with fully convolutional regression networks." Computer methods in biomechanics and biomedical engineering: Imaging & Visualization 6.3 (2018): 283-292.

[16] Zhu, Runkai, et al. "An extended type cell detection and counting method based on FCN." 2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE). IEEE, 2017.

[17] Pan, Xipeng, et al. "Cell detection in pathology and microscopy images with multi-scale fully convolutional neural networks." World Wide Web 21.6 (2018): 1721-1743.

[18] Walach, Elad, and Lior Wolf. "Learning to count with cnn boosting." European conference on computer vision. Springer, Cham, 2016.

[19] Paul Cohen, Joseph, et al. "Count-ception: Counting by fully convolutional redundant counting." Proceedings of the IEEE International conference on computer vision workshops. 2017.

[20] Rad, Reza Moradi, et al. "Blastomere cell counting and centroid localization in microscopic images of human embryo." 2018 IEEE 20th international workshop on multimedia signal processing (MMSP). IEEE, 2018.

[21] Akram, Saad Ullah, et al. "Cell segmentation proposal network for microscopy image analysis." Deep Learning and Data Labeling for Medical Applications. Springer, Cham, 2016. 21-29.

[22] Idrees, Haroon, et al. "Multi-source multi-scale counting in extremely dense crowd images." Proceedings of the IEEE conference on computer vision and pattern recognition. 2013.

[23] Zhang, Shanghang, et al. "Fcn-rlstm: Deep spatio-temporal neural networks for vehicle counting in city cameras." Proceedings of the IEEE international conference on computer vision. 2017.

[24] Bilen, Hakan, and Andrea Vedaldi. "Universal representations: The missing link between faces, text, planktons, and cat breeds." arXiv preprint arXiv:1701.07275 (2017).

[25] Rebuffi, Sylvestre-Alvise, Hakan Bilen, and Andrea Vedaldi. "Learning multiple visual domains with residual adapters." arXiv preprint arXiv:1705.08045 (2017).

[26] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[27] Chen, Liang-Chieh, et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation." Proceedings of the European conference on computer vision (ECCV). 2018.

[28] Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." arXiv preprint arXiv:1711.05101 (2017).

[29] P. Kainz, M. Urschler, S. Schulter, P. Wohlhart, and V. Lepetit, "You should use regression to detect cells," in International Conference on Medical Image Computing and Computer-Assisted Intervention, (Springer, 2015), 276–283.

[30] Marsden, Mark, et al. "People, penguins and petri dishes: Adapting object counting models to new visual domains and object types without forgetting." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[31] Lonsdale, John, et al. "The genotype-tissue expression (GTEx) project." Nature genetics 45.6 (2013): 580-585.

[32] Y. Xue, N. Ray, J. Hugh, and G. Bigras, "Cell counting by regression using convolutional neural network," in European Conference on Computer Vision, (Springer, 2016), 274–290.

[33] E. Lu, W. Xie, and A. Zisserman, "Class-agnostic counting," in Asian conference on computer vision, (Springer, 2018), 669–684.

[34] Jiang, Ni, and Feihong Yu. "Multi-column network for cell counting." OSA Continuum 3.7 (2020): 1834-1846.

[35] Jia, Chen, Abhyudai Singh, and Ramon Grima. "Cell size distribution of lineage data: analytic results and parameter inference." Iscience 24.3 (2021): 102220.