Microscopy Cell Counting with Fully Convolutional Regression Networks

Weidi Xie, J. Alison Noble, Andrew Zisserman

Department of Engineering Science, University of Oxford,UK

Abstract. This paper concerns automated cell counting in microscopy images. The approach we take is to use Convolutional Neural Networks (CNNs) to regress a cell spatial density across the image. This is applicable to situations where traditional single-cell segmentation based methods do not work well due to cell clumping or overlap.
We make the following contributions: **(i)** we develop and compare architectures for two Fully Convolutional Regression Networks (FCRNs) for this task; **(ii)** since the networks are fully convolutional, they can predict a density map for an input image of arbitrary size, and we exploit this to improve efficiency at training time by training end-to-end on image patches; and **(iii)** we show that FCRNs trained entirely on synthetic data are able to give excellent predictions on real microscopy images without fine-tuning, and that the performance can be further improved by fine-tuning on the real images.
We set a new state-of-the-art performance for cell counting on standard synthetic image benchmarks and, as a side benefit, show the potential of the FCRNs for providing cell detections for overlapping cells.

1 Introduction

Counting objects in crowded images or videos is an extremely tedious and time-consuming task encountered in many real-world applications, including biology, remote sensing, surveillance, etc. In this paper, we focus on cell counting in microscopy, but the developed methodology could also be used in other counting applications. Numerous procedures in biology and medicine require cell counting, for instance: a patient's health can be inferred from the number of red blood cells and white blood cells; in clinical pathology, cell counts from images can be used for investigating hypotheses about developmental or pathological processes; and cell concentration is important in molecular biology, where it can be used to adjust the amount of chemicals to be applied in an experiment.

Automatic cell counting can be approached from two directions, one is detection-based counting [1, 6], which requires prior detection or segmentation; the other is based on density estimation without the need for prior object detection or segmentation [2, 5, 10]. Recent work shows that the latter approach has so far been faster and more accurate than detection-based approaches.

Following [10], we first cast the cell counting problem as a supervised learning problem that tries to learn a mapping between an image $I(x)$ and a density map

$D(x)$, denoted as $F : I(x) \rightarrow D(x)$ $(I \in R^{m \times n}, D \in R^{m \times n})$ for a $m \times n$ pixel image, see Fig. 1. The density map $D(x)$ is a function over pixels in the image, and integrating this map over an image region gives an estimate of the number of cells in that region.

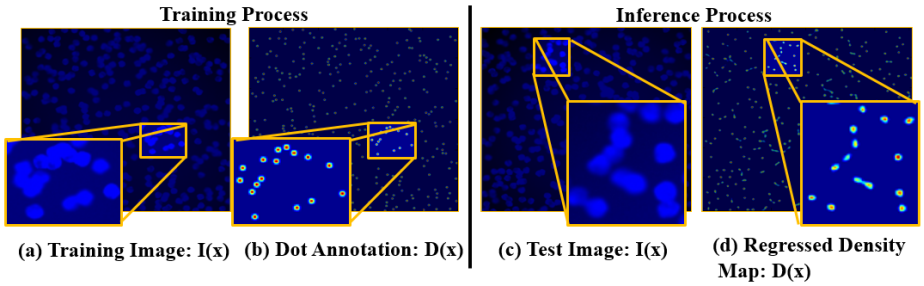

Fig. 1: *Problem Scenario:* **Left:** Training Process. **Right:** Inference Process. **(a):** Training image from a synthetic dataset [9]. **(b):** Dot annotations that create a Gaussian at the center of each cell with $\sigma = 2$. **(c):** Image from the test set. **(d):** Estimated Density Map, the number of cells in a specific region is calculated by integrating the density map over that region.

Recently, Convolutional Neural Networks (CNNs) [7, 8] have re-emerged as a mainstream tool in the computer vision community. They are also starting to become popular in biomedical image analysis and have achieved state-of-the-art performance in several areas, such as mitosis detection [4], neuronal membranes segmentation [3], and analysis of developing C. elegans embryos [12]. However, they have not yet been applied to solving the target problem here of regression in microscopy image cell counting.

In this paper we develop a Fully Convolutional Regression Networks (FCRNs) approach for regression of a density map. In section 2, we describe and compare two alternative architectures for the FCRNs, and discuss how the networks are trained efficiently using end-to-end patch training. In section 3, we present results on a synthetic dataset, and also show that a network trained only on synthetic data can be used to accurately regress counting density maps for different kinds of real microscopy images. Finally, we show that the performance of FCRNs can be further improved by fine-tuning parameters with real microscopy images. Overall, experimental results show that FCRNs can provide state-of-the-art cell counting, as well as the potential for cell detection of overlapping cells.

1.1 Related Work

Counting by density estimation: Cell counting in crowded microscopy images with density estimation avoids the difficult detection and segmentation of individual cells. It is a good alternative for tasks where only the number of cells is required. Over the recent years, several works have investigated this approach. In [10], the problem was cast as density estimation with a supervised learning algorithm, $D(x) = c^T \phi(x)$, where $D(x)$ represents the ground-truth density map, and $\phi(x)$ represents the local features. The parameters c are learned by

minimizing the error between the true and predicted density with quadratic programming over all possible sub-windows. In [5], a regression forest is used to exploit the patch-based idea to learn structured labels, then for a new input image, the density map is estimated by averaging over structured, patch-based predictions. In [2], an algorithm is proposed that allows fast interactive counting by simply solving ridge regression.

Fully Convolutional Networks: Recently, [11] developed a fully convolutional network for semantic labelling. By reinterpreting the fully connected layers of a classification net as convolutional and fine-tuning upsampling filters, it can take an input of arbitrary size and produce a correspondingly-sized output both for end-to-end training and for inference.

Inspired by these previous works, we propose Fully Convolutional Regression Networks (FCRNs) that allow end-to-end training for regression of images of arbitrary size.

2 Counting with Fully Convolutional Regression Networks

The problem scenario is shown in Fig. 1. The ground truth is provided as dot annotations, where each dot corresponds to one cell. For training, the dot annotations are each represented by a Gaussian, and a density surface $D(x)$ is formed by the superposition of these Gaussians. The task is to regress this density surface from the corresponding cell image $I(x)$. This is achieved by training a CNN using the mean square error between the output heat map and the target density surface as the loss function for regression. At inference time, given an input cell image $I(x)$, the CNN then predicts the density heat map $D(x)$.

The popular CNN architecture for classification contains convolution-ReLU-pooling [7]. Here, ReLU refers to rectified linear units. Pooling usually refers to max pooling and results in a shrinkage of the feature maps. However, in order to produce density maps that have equal size to the input, we follow the idea suggested in [11] and reinterpret the fully connected layers as convolutional layers. The first several layers of our network contains regular convolution-ReLU-pooling, then we undo the spatial reduction by performing upsampling-ReLU-convolution, map the feature maps of dense representation back to the original resolution (Fig. 2). During upsampling, we first use bilinear interpolation, followed by convolution kernels that can be learnt during end-to-end training. We present two networks, namely FCRN-A, FCRN-B.

Inspired by the very deep VGG-net [14], in both regression networks, we only use small kernels of size 3×3 or 5×5 pixels. The number of feature maps in the higher layers is increased to compensate for the loss of spatial information caused by max pooling. In FCRN-A, all of the kernels are of size 3×3 pixels, and three max-poolings are used to aggregate spatial information leading to an effective receptive field of size 38×38 pixels (i.e. the input footprint corresponding to each pixel in the output). FCRN-A provides an efficient way to increase the receptive

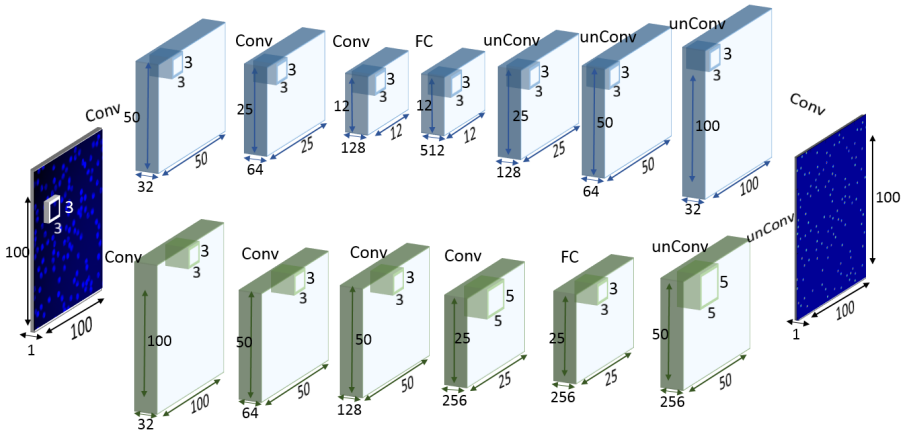

Fig. 2: *Network Structures:* FCRN-A is shown in **blue** & FCRN-B is shown in **green**. In both architectures, we first map the input image to feature maps with dense representation, and then recover the spatial span by bilinear upsampling.
FC – Fully Connected Layer (Implemented as convolution);
Conv – Convolutional Layer + ReLU (+ Max Pooling);
unConv – Upsampling + ReLU + Convolution;

field, while contains only about 1.3 million trainable parameters. In contrast, max pooling is used after every two convolutional layers to avoid too much spatial information loss in FCRN-B. In this case, the number of feature maps is increased after every max pooling up to 256, with this number of feature maps then retained for the remaining layers. Comparing with FCRN-A, in FCRN-B we use 5×5 filters in some layers leading to the effective receptive field of size 32×32 pixels. In total, FCRN-B contains about 3.6 million trainable parameters, which is about three times as many as those in FCRN-A.

2.1 Implementation details

The implementations are based on MatConvNet [15]. Back-propagation and stochastic gradient descent are used for optimization. During training, we cut large images into patches, for instance, we randomly sample 500 small patches of size 100×100 from 500×500 images. The amount of data for training has been increased dramatically in this way. Each patch is normalized by subtracting its own mean value and then dividing by the standard deviation. The parameters of the convolution kernels are initialized with an orthogonal basis [13]. Then the parameters w are updated by: $\triangle w_{t+1} = \beta \triangle w_t + (1-\beta)(\alpha \frac{\partial l}{\partial w})$, where α is the learning rate, and β is the momentum parameter. We initialize the learning rate as 0.01 and decrease it by a factor of 10 every 5 epochs. The momentum is set to 0.9, weight decay is 0.0005, and no dropout is used in either network. Besides good initialization of parameters, the Gaussian-annotated ground truth (Fig. 1b) must be scaled, for example multiplying the ground truth annotation by 100. Without this operation, the peak value for a Gaussian with $\sigma = 2$ is only about 0.07. Most of the pixels in the ground truth belong to background and

are labeled as zero. Therefore, the networks tend to be more focusing on fitting the background zero rather than Gaussian shapes.

After pretraining with patches, we fine-tune the parameters with whole images to smooth the estimated density map, since the 100×100 image patches sometimes may only contain part of a cell on the boundary. We train our networks on a computer with 8 Intel Xeon 3.5GHz CPUs. It took less than 8 hours to converge. Training the same architecture on a GPU would be a lot faster.

3 Experimental validation

In this section, we first determine how FCRN-A and B compare with previous work using synthetic data. Then we apply the network trained only on synthetic data to a variety of real microscopy images without fine-tuning. Finally, we compare the performance before and after fine-tuning on real microscopy images.

3.1 Dataset and evaluation protocol

Synthetic data: We generated 200 fluorescence microscopy cell images [9], each synthetic image has an average of 174±64 cells. The number of training images was between 8 and 64. After testing on 100 images, we report the mean absolute errors and standard deviations for FCRN-A and FCRN-B.

Real data: We evaluated FCRN-A and FCRN-B on two data sets; (1) retinal pigment epithelial (RPE) cell images. The quantitative anatomy of RPE can be important for physiology and pathophysiology of the visual process, especially in evaluating the effects of aging; and (2) Images of precursor T-Cell lymphoblastic lymphoma. Lymphoma is the most common blood cancer, usually occurs when cells of the immune system grow and multiply uncontrollably.

3.2 Synthetic Data

Network Comparison: Each image is mapped to a density map first, integrating over the map for a specific region gives the count of that region (Fig. 3). The performance of the two networks is compared in Table 1 as a function of the number of training images.

As shown in Table 1, FCRN-A performs slightly better than FCRN-B. The *size* of the receptive field turns out to be more important than being able to provide more detailed information *over* the receptive field, probably because the real difficulty in cell counting lies in regression for large cell clumps, and a larger receptive field is required to span these. For both networks, the performance is observed to improve by using more training images from $N = 8$ to $N = 32$, and only little changes if N is increased to 64.

The error cases mainly come from two sources: *firstly* from the boundary effect due to bilinear up-sampling, cells on the boundary of images tend to produce wrong predictions; *secondly*, is from very large cell clumps where four or more cells overlap. In the latter case, larger clumps can be more variable in shape

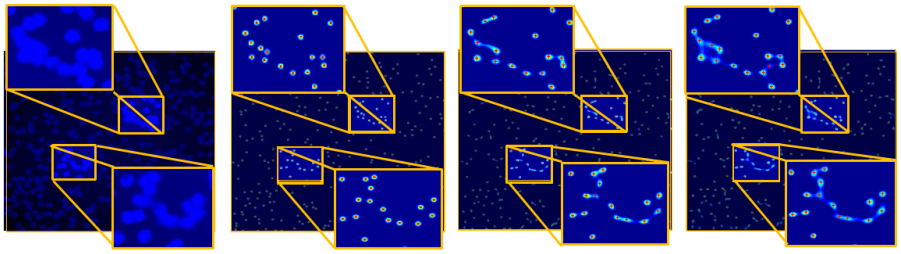

Fig. 3: *Inference Process:* **(a):** Input. **(b):** Ground-truth dot annotation. **(c):** Density map from FCRN-A. **(d):** Density map from FCRN-B. The density map is calculated first, then integration over the density map gives the cell count. For visualization, red crosses are obtained by taking local maxima detection.

Method	174±64 cells			
	N=8	N=16	N=32	N=64
Img-level ridge-reg [10]	8.8±1.5	6.4±0.7	5.9±0.5	N/A
Dens. estim. (MESA) [10]	4.9±0.7	3.8±0.2	3.5±0.2	N/A
Dens. estim. (RF) [5]	**3.4±0.1**	N/A	3.2±0.1	N/A
Dens. estim. (Interactive) [2]	4.5±0.6	3.8±0.3	3.5±0.1	N/A
Dens. estim. (Proposed FCRN-A)	3.9±0.5	**3.4±0.2**	**2.9±0.2**	**2.9±0.2**
Dens. estim. (Proposed FCRN-B)	4.1±0.5	3.7±0.3	3.3±0.2	3.2±0.2

Table 1: *Mean absolute error and standard deviations for cell counting on the synthetic cell dataset* [9]. The columns correspond to the number of training images. Standard deviations corresponds to five different draws of training and validation image sets.

than individual cells and so are harder to regress; further, regression for large cell clumps requires the network to have an even larger receptive field that can cover important parts of the entire clumps, like curved edges in specific directions. Our networks are relatively shallow and only have receptive field of size 38×38 pixels and 32×32 pixels. For elongated cell clumps, their curved edges can usually be covered, and correct predictions can be made. However, a roughly rounded cell clump with four or more cells is bigger than our largest receptive field, and this will lead to an incorrect prediction.

Comparison with state-of-the-art: Table 1 shows a comparison with previous methods on the synthetic cell dataset. FCRN-A shows about 9.4% improvement over the previous best method of [5] when $N = 32$.

3.3 Real Data

We test both regression networks on real datasets. However, limited by the space, we only show figures for results from FCRN-A in Fig. 4 (without fine-tuning) and Fig. 5 (before and after fine-tuning). During fine-tuning, two images of size 2500×

2500 pixels, distinct from the test image, are used for fine-tuning in a patch-based manner, the same annotations following Fig. 1b were performed manually by one individual, each image contains over 7000 cells. It can be seen that the performance of FCRN-A on real images improves by fine-tuning, reducing the error of 34 out of 1502 (before fine-tuning) to 18 out of 1502 (after fine-tuning). When testing FCRN-B on these two datasets, for RPE cells: Ground-truth / Estimated count = 705 / 698, and for Precursor T-Cell LBL cells: Ground-truth / Estimated count = 1502 / 1472 (Without fine-tuning). Surprisingly, FCRN-B achieves slightly better performance on real data. Our conjecture is that real data contains smaller cell clumps than synthetic data, therefore, the shape of cell clumps will not vary a lot. The network is then able to give a good prediction even with a small receptive field.

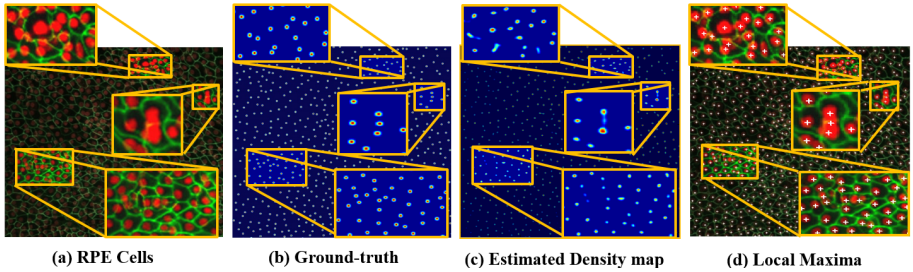

Fig. 4: *Test result on RPE:* **(a):** Retinal Pigment Epithelial Cells. **(b):** Ground-truth. **(c):** Estimated density map from FCRN-A. **(d):** Output by taking local maxima (White crosses). Ground-truth / Estimated count = 705 / 696. The data is from: http://sitn.hms.harvard.edu/waves/2014/a-stem-cell-milestone-2/

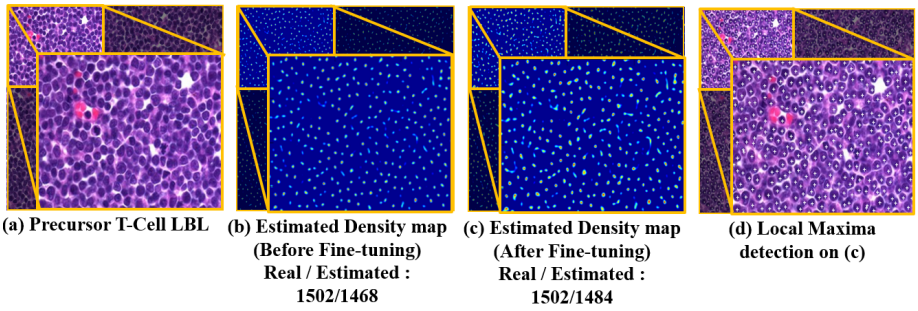

Fig. 5: *Test result on Precursor T-Cell LBL:* **(a):** Precursor T-Cell LBL. **(b):** Estimated density map from FCRN-A. **(c):** Estimated density map from fine-tuned FCRN-A. **(d):** Output by taking local maxima on (c) (White crosses).

4 Summary

We have proposed two Fully Convolutional Regression Networks for solving regression problems, focusing on cell counting. The approach is able to perform fast inference and accurate cell counting for real microscopy images. As a side benefit, the result shows the potential for cell detection – see the local maxima

of the predicted cell density in Fig. 4d and Fig. 5d.

Acknowledgement. Financial support for Weidi Xie was provided by a Google studentship and the China Oxford Scholarship Funds.

References

1. C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman, Learning to detect cells using non-overlapping extremal regions, in Proc. MICCAI, 2012, pp. 348–356.
2. ———, Interactive object counting, in Proc. ECCV, 2014, pp. 504–518.
3. D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, Deep neural networks segment neuronal membranes in electron microscopy images, in NIPS, 2012, pp. 2843–2851.
4. D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, Mitosis detection in breast cancer histology images with deep neural networks, in Proc. MICCAI, 2013, pp. 411–418.
5. L. Fiaschi, R. Nair, U. Koethe, and F. A. Hamprecht, Learning to count with regression forest and structured labels, in Proc. ICPR, IEEE, 2012, pp. 2685–2688.
6. R. Girshick, J. Donahue, T. Darrell, and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in Proc. CVPR, IEEE, 2014, pp. 580–587.
7. A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, in NIPS, 2012, pp. 1097–1105.
8. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE, 86 (1998), pp. 2278–2324.
9. A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja, Computational framework for simulating fluorescence microscope images with cell populations, Medical Imaging, IEEE Transactions on, 26 (2007), pp. 1010–1016.
10. V. Lempitsky and A. Zisserman, Learning to count objects in images, in NIPS, 2010, pp. 1324–1332.
11. J. Long, E. Shelhamer, and T. Darrell, Fully convolutional networks for semantic segmentation, in Proc. CVPR, IEEE, 2015, pp. 3431–3440.
12. F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. E. Barbano, Toward automatic phenotyping of developing embryos from videos, Image Processing, IEEE Transactions on, 14 (2005), pp. 1360–1371.
13. A. M. Saxe, J. L. McClelland, and S. Ganguli, Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, Proc. ICLR, (2014).
14. K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, Proc. ICLR, (2015).
15. A. Vedaldi and K. Lenc, Matconvnet-convolutional neural networks for matlab, arXiv preprint arXiv:1412.4564, (2014).