Cell Counting by Regression Using Convolutional Neural Network

Yao Xue[1], Nilanjan Ray[1(✉)], Judith Hugh[2], and Gilbert Bigras[2]

[1] Department of Computing Science, University of Alberta, Edmonton, Canada
nray1@ualberta.ca
[2] Department of Laboratory Medicine, University of Alberta, Edmonton, Canada

Abstract. The ability to accurately quantitate specific populations of cells is important for precision diagnostics in laboratory medicine. For example, the quantization of positive tumor cells can be used clinically to determine the need for chemotherapy in a cancer patient. In this paper, we describe a supervised learning framework with Convolutional Neural Network (CNN) and cast the cell counting task as a regression problem, where the global cell count is taken as the annotation to supervise training, instead of following the classification or detection framework. To further decrease the prediction error of counting, we tune several cutting-edge CNN architectures (e.g. Deep Residual Network) into the regression model. As the final output, not only the cell count is estimated for an image, but also its spatial density map is provided. The proposed method is evaluated with three state-of-the-art approaches on three cell image datasets and obtain superior performance.

Keywords: Cell counting · Convolution neural network · Deep residual net · Detection · Classification

1 Introduction

1.1 Problem Definition

Automatic cell counting is to obtain the number of certain types of cells in a medical image like microscopic images. It is of great interest to a wide range of medical scenarios [2,4,23]. An example is the diagnosis and treatment of breast cancer, which is one of the most common female diseases leading to death worldwide. The number of proliferating (e.g. Ki67 positive) tumor cells is an important index associated with the severity of disease clinically. One available method of quantization involves counting the nuclei of proliferating cells using traditional image analysis techniques on a microscopic image. However, it has been proven to be challenging because of inability to distinguish tumor cells from surrounding normal tissue like vessels, fat and fibrous tissue [11], especially in reality the resolution of input medical image could be very high, at the same time the target cells could easily be extremely dense. Consequently, it is quite difficult to manually count target cells one by one. This is the principal motivation of automatic cell counting.

G. Hua and H. Jégou (Eds.): ECCV 2016 Workshops, Part I, LNCS 9913, pp. 274–290, 2016.
DOI: 10.1007/978-3-319-46604-0_20

1.2 Background

From the perspective of computer vision community, automatic cell counting is a branch task of the object counting problem. Many methods have chosen to fulfill object counting task following the detection pipeline [17,20,21,27,30,32]. In this case, an object detection framework is designed to localize each object (e.g. cell, head or vehicle) one by one, after that a counter naturally takes all the detected objects and produces the final count. Following the success of deep learning applied in computer vision like detection, segmentation and localization [13,14,33], most recent counting-by-detection works choose learning based approach, where each training object has annotation information, sometimes dot annotation indicating the centroid of the object [8,18,31], sometimes bounding-box annotation around the object [30]. However, it is well known that the problem of detecting and localizing individual object instance is far from being solved, especially in real world application of cell counting, where the cell density can be extremely high [7,15,29]. For example, the number of cells can easily reach or exceed thousands per image, and the cells also show huge variations in terms of type, size, shape and appearance etc. Another related work is Fully Convolutional Neural Network (FCNN) [26], which has remarkable result in semantic segmentation and spatial density prediction (in some way, object counting can be seen as an integration over spatial density prediction). To build the end-to-end and pixel-to-pixel FCNN, its training phase requires the pixel-wise annotation, which is strongly supervised information and gives much benefit to FCNN, consequently work like [9] is proposed to compensate.

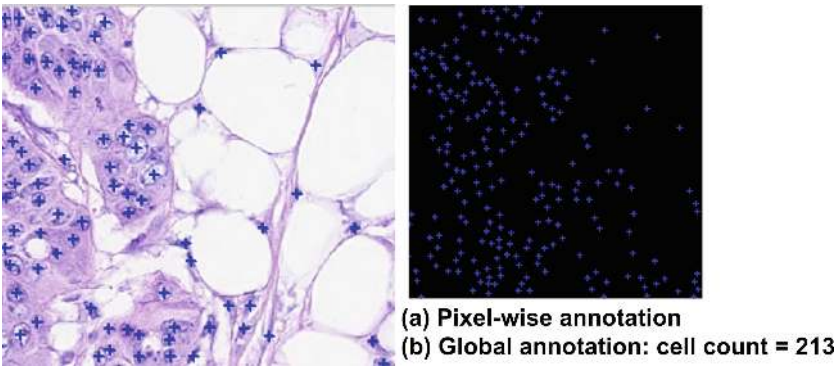

Fig. 1. Counting-by-detection framework, most of which requires the pixel-wise annotation, is not a good choice in cell counting task, where cells could be extremely dense.

Figure 1 shows a cell image example with its two level of annotations: (a) pixel-wise annotation, where the nuclei of each cell is dot-annotated; (b) global count, where the total number of cell in the image is provided. Most counting-by-detection framework takes the strongly supervised pixel-wise annotation as

input during training, and then generate global count (i.e. the total number of target cells) for test image. However, the automatic cell counting task is to predict a global count only, thus it is a limitation and also unnecessary to design an object counting framework relying on the more expensive annotation data, which will make system unpractical in the real-world application of cell counting.

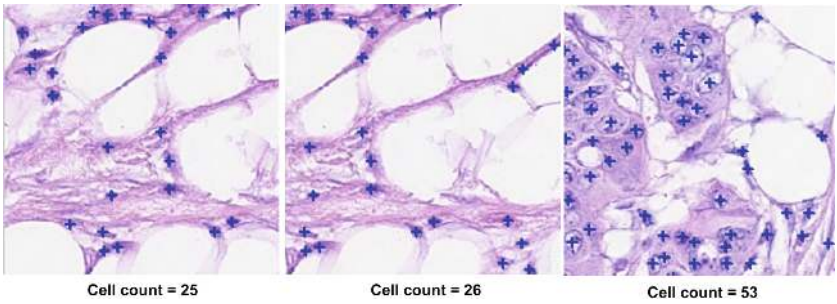

Fig. 2. Cell counting is better to be modeled as a regression problem rather than a classification problem.

Here we expand the scope of our related work study from purely cell counting to all the object counting problem. Object (including cell, people, vehicle, etc.) counting task [3,7,8,15,18,19,24,29,35] benefits a lot from the superior performance of Convolutional Neural Network (CNN) in recent years. Several work [3,7,15] exploit CNN to enable their counting system work in dense and crowded environment. While [24,29] focus on extracting deep features from pre-trained or fine-tuned CNN models. However a multi-class classification CNN is usually trained and used. That means the counting problem is cast as a classification problem, where the cell count is treated as class ID, and images with the same number of objects are seen as belonging to the same class. During its test phase, each test image is predicted with an integer class label (for example: 25, 26 or 53 in Fig. 2), which indicates the number of objects in the image. However as we know in object classification task, training images of different categories (for example: cat, bicycle and airplane) are independent to each other, and softmax loss function is used in classification CNN architectures [13,14]. Actually, a classification-orientated CNN model treats cell counts 25 and 53 as far apart as counts 25 and 26. However from the nature of cell counting, the distance between different cell counts is very important. It is beneficial to treat the object counting task as a regression task [29], where counts 25 and 26 should be closer and that could be correctly reflected in the regression setting.

Taking all the discussion above into consideration, the advantages of cell counting approach proposed in this paper can be summarized into the following aspects:

1. To capture the relationship between RGB cell image and its overall cell count, we cast the cell counting task as developing an end-to-end regression

framework, which is more suitable for counting task compared to counting by detection. Additionally, instead of being applied for classification purpose, a convolutional neural network architecture with Euclidean loss function is used for regression.
2. As the final output, the proposed approach not only estimate the global number of certain cells in an image but also produce the spatial density prediction, which is able to describe the local cell density of an image sub-region. In many clinical imaging systems [10,28,34], researchers have confirmed that the topographic map that illustrate the cell density distribution is a valuable tool correlated with the disease diagnose and treatment.
3. We utilize several mainstream CNN architectures (including the Deep Residual Network [13], AlexNet [14]) into our regression model. To the best of our knowledge, this is the first piece of work to expand the deep residual network from classification, detection, segmentation to object counting.

2 The Proposed Framework

2.1 System Overview

In this section, we give a general overview on the proposed approach, details of every part are provided in the following sections. In this paper, we propose a supervised learning framework for cell counting task shown in Fig. 3.

In the training phase, a Convolutional Neural Network (CNN) is utilized to build a regression model between image patch and its cell count number. We employ several kinds of CNN architectures and use Euclidean loss function during training, to enable the regression model fit for the cell counting task. To prepare the training data, we generate a large number of square patches from every training image. Along with each training patch, there is a patch count number, which indicates the number of target cells present in the patch. After that, patch rotation is performed on the collected training patches for the purpose of making the system more robust to rotational variance and data augmentation.

In the test phase, one test image is cropped into a number of overlapping test patches with the same size as the training patches in the sliding-window manner. Each of these test patches is passed into the CNN-based regression model, and then the estimated cell count of the input test patch is output from the last layer of the CNN model. After predicting cell counts for all the patches, we perform a 2-D Linear Interpolation over the estimated cell count and its corresponding x-y coordinates to build a heatmap, which provides a spatial density prediction as shown in Fig. 3. Lastly, integrating these interpolated counts on pixel locations provides us the final count on the test image. The whole procedure is illustrated in Fig. 3.

2.2 Data Preparation and Processing

In the real application of automatic cell counting, the resolution of input medical image could be very high, at the same time the target cells could be very dense.

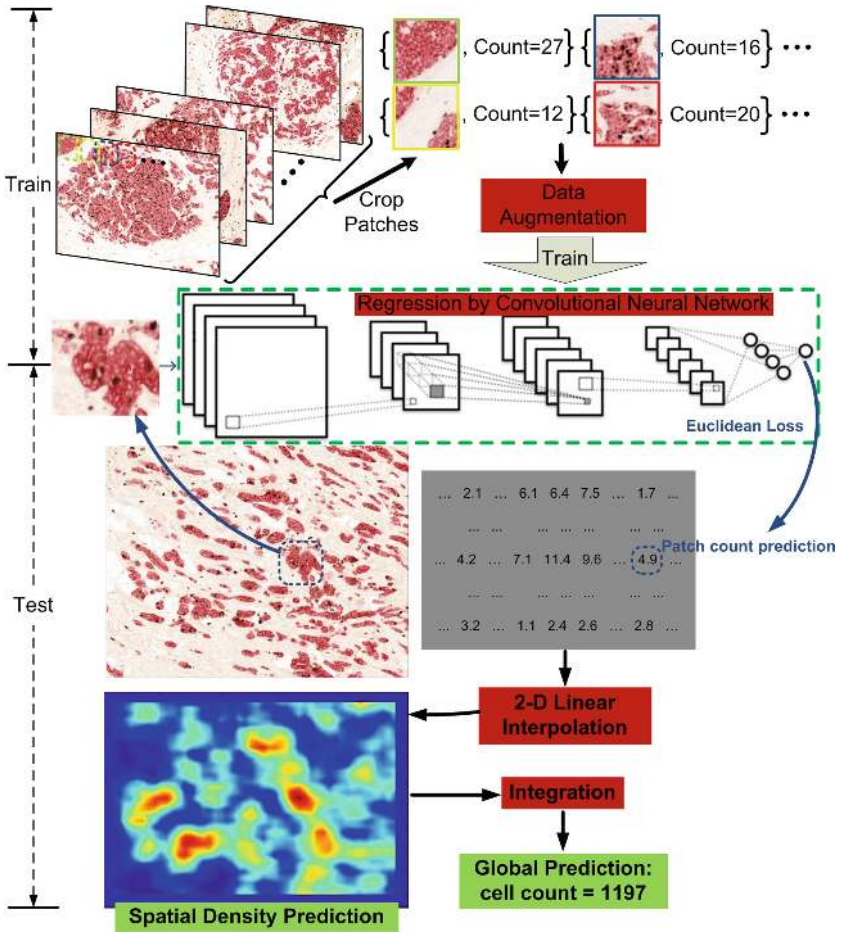

Fig. 3. System overview

Consequently, it is quite difficult to manually count target cells one by one. This is the original motivation of automatic cell counting. Considering the nature of these medical images, which need automatic cell counting, data preparation and processing is naturally necessary. In this work, we crop image into consistent patches and then perform training and prediction over these patches, in order to (1) make the approach more robust to scale variance, (2) avoid resizing original microscope image, which could cause information loss, (3) prepare more training data to prevent the CNN based regression model from overfitting during training.

The proposed method operates by first partitioning image to smaller patches. Patches are generated in a sliding window manner: from the top-left corner of a large W-by-H image with a certain patch size and stride size. Usually, stride size is set smaller than half of path size to ensure that adjacent patches have overlapping region. To construct training data, every training patch is

accompanied by a patch count, which is an integer indicating how many cells exist in the patch. Then for data augmentation, a training patch is rotated from 0 degree to 360 degree with a certain rotation step, for example 30 degrees.

2.3 Convolutional Neural Network Regression Model

2.3.1 Classification vs. Regression for Counting

As we know, in a CNN-based classification model, the network outputs a vector whose size is the same size as the number of classes. The i-element in the vector describes the confidence score that the input image belongs to the i-th class. During test phase, the index with the highest confidence score is selected as the final classification result. Softmax loss is widely used for classification problem.

However for counting problems, it is not proper to take cell count number as class index. The reason why regression is a better choice than classification for counting task has been explained in detail in introduction part. In our counting-by-regression model, the difference between ground-truth value and the estimated value can be better preserved during calculating the error. This information is quite helpful for optimizing the CNN weights more accurately in the back-propagation phase. The layer of our regression model outputs a single number, indicating the number of cells that our model predicts. In our model, we employ two kinds of CNN architectures, the first one is AlexNet [14] which consists of 5 convolution layers + 3 fully connected layers; the other is the deep residual network (ResNet) [13] which we will explain in the next section. In both of these architectures, the loss function is defined as the Euclidean loss, which measures the sum of squares of differences between the ground truth and prediction. We train the AlexNet model from scratch with Softmax and Euclidean loss layer respectively, the performance improvement of regression over classification is experimentally explained in detail at Sect. 3.4.

2.3.2 Deep Residual Network for Regression

Since the 2012 ImageNet competition, convolutional neural networks have become popular in large scale image recognition tasks, several milestone networks (including AlexNet [14] VGGNet [22] and GoogLeNet [6], etc.) have been proposed. Recently, the introduction of residual connections into deep convolutional networks has yielded state-of-the-art performance in the 2015 ILSVRC challenge. This raises the question of whether there is any benefit in introducing deep residual network (ResNet) [13] in to the cell counting task. In the following section, we are going to explain the network architecture and its components used in this paper.

Convolutional layer. It consists of a set of learnable filters. During the forward pass, we slide each filter along the width and height of the input volume and compute dot products between its weights and the activation map from previous layer. Intuitively, the filters will be trained to be active to some type of visual feature such as an edge of some orientation or a blotch of some color on the first layer.

Pooling layer. It works by down-sampling the convolutional features using the max operation (*max-pooling*) or average operation (*average-pooling*). Pooling layer is usually inserted between successive convolutional layers, in order to reduce the amount of network parameters and also to control overfitting.

Batch Normalization layer. In the ResNet architecture, authors [13] deploy Batch Normalization (BN) layer [25] right after each convolution and before activation. As we know, normalization is often used as a pre-processing step to make the data consistent. When the input flows through a deep network, the weights and parameters adjust the values of the input, sometimes making the data too big or too small again. Batch Normalization layer allows us to normalize the data in each mini-batch across the network rather than just performing normalization once in the beginning, thus this problem is largely avoided. [25] has demonstrated that batch normalization helps to boost the learning speed and also increase the overall accuracy.

Fully-Connected layer. As the name implies, each neuron in a Fully-Connected (FC) layer has full connections to all neurons in the previous layer. After gathering all the responses from previous layers into each of its neuron, fully connected layer is responsible for computing a class-specific confidence vectors, where its each neuron outputs a score for a certain class. For example, the ResNet ends with a 1000-way fully-connected layer, on which the class with maximum score is selected as its final predicted label.

Overall Architecture. Different from other CNN architectures, ResNet consists of a number of Residual Blocks. Each residual block is a made up of Convolutional layer, Batch Normalization layer and a shortcut that connects the original input with the output as shown in Fig. 4 (a) and (b), where a Residual Block with Identity Shortcut (RB-IS) and a Residual Block with Projection Shortcut (RB-PS) is illustrated, respectively. The mathematical model of residual block can be summarized as:

$$y_l = F(X_l, \{W_l\}) + h(X_l) \qquad X_{l+1} = f(y_l) \tag{1}$$

$$h(X_l) = \begin{cases} X_l & identity\,mapping \\ W_p X_l & projection\,mapping \end{cases} \tag{2}$$

X_l and X_{l+1} are the input and output of the l-th residual block, $F(X_l, \{W_l\})$ stands for the residual function, and f is a activation function (e.g. ReLU). $h(X_l)$ represents the shortcut connection: identity mapping or projection mapping. If the dimension of X_l and X_{l+1} is the same, the identity shortcuts is used; otherwise a linear projection W_p is performed on the shortcut connections to match the dimension, that is projection mapping. The central idea [13] of ResNet is to learn the additive residual function F with respect to $h(X_l)$, with a key choice of using an identity mapping and/or projection mapping.

Figure 4 (a) and (b) show two types of Residual Blocks, which are used in different layers of ResNet model (c) according to whether the dimensions of

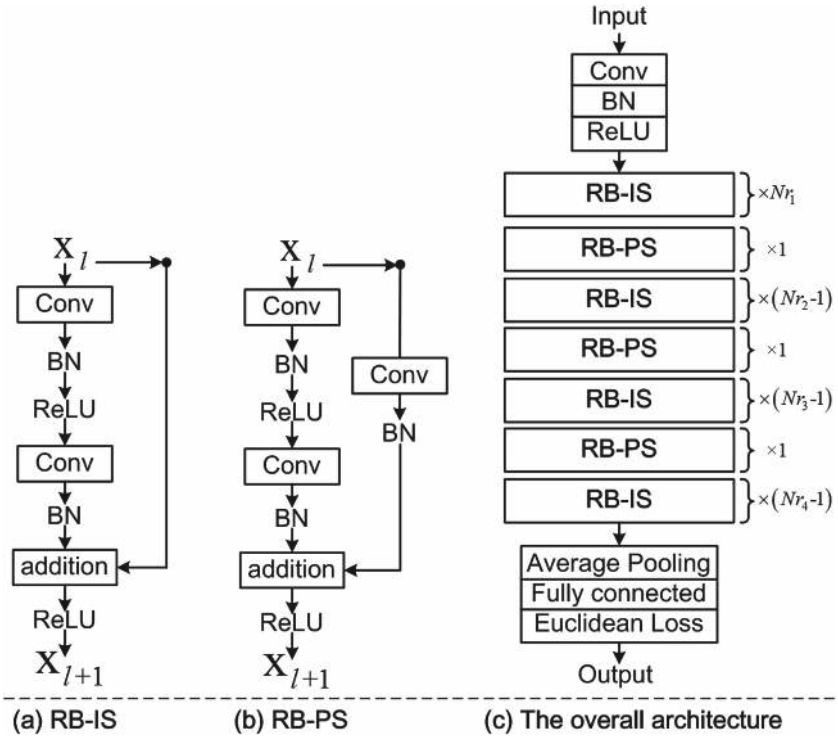

Fig. 4. (a) RB-IS stands for the Residual Block with Identity Shortcut; (b) RB-PS is the Residual Block with Projection Shortcut; (c) An illustration of the architecture that we used in this paper.

input and output are the same. Nr_1, Nr_2, Nr_3 and Nr_4 represent the number of residual blocks used in four sections of ResNet model. For example, $Nr_1 = 3$, $Nr_2 = 4$, $Nr_3 = 6$ and $Nr_4 = 3$ in ResNet-50. Additionally, it has been demonstrated that pre-trained network can be adjusted to be effective for other computer vision tasks. We modify the last fully-connected layer of ResNet from outputting a 1000-D vector to outputting 1 item indicating the predicted number of cells. Additionally, we replace the softmax loss with Euclidean loss. After that, we perform fine-tuning on the weights in fully-connected layer of the ResNet using cell datasets, the parameters of previous layers are preserved. Finally, we obtain three ResNet based regression models for cell counting.

3 Experiment and Performance

3.1 Datasets

First, we describe the three cell datasets, on which the proposed method and other comparison methods are evaluated. The first dataset [12] involves 100 H&E

stained histology images of colorectal adenocarcinomas. A total of 29,756 nuclei were marked at/around the center with over 22,000 labeled with the cell type. The second dataset [1] consists of 200 highly-realistic synthetic emulations of fluorescence microscopic images of bacterial cells. The third dataset comprise of 55 high-resolution RGB images, each of them is a microscopic image of proliferative tumor cells area with a resolution of 1920-by-2560 pixels. The tumor cell size is about 10 to 20 pixels in diameter or 10 μm in physical length.

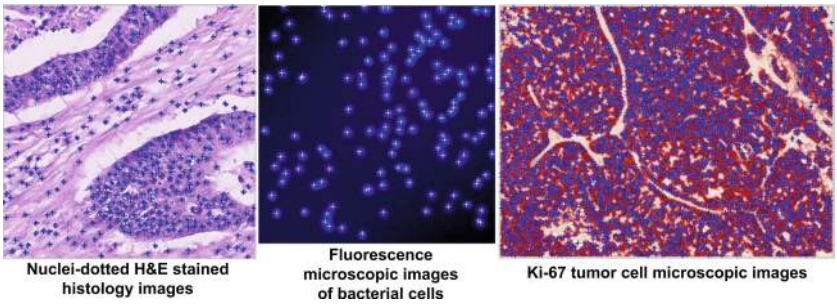

Fig. 5. Example images from the three evaluation datasets and their dotted annotation. (Color figure online)

Table 1. Details of three datasets: *Size* is the image size; *Ntr/Nte* is the number of images selected for training and testing; *AC* indicates the average number of cells; *MinC-MaxC* is the minimum and maximum numbers of cells in a dataset.

Cell dataset	Size	Ntr/Nte	AC	MinC-MaxC
Nuclei [12]	500×500	50/50	310.22	1–1189
Bacterial [1]	256×256	100/100	171.47	74–317
Ki67 cell	1920×2560	45/10	2045.85	70–4808

To build this Ki-67 cell image dataset, a 10X microscopic field representing the highest proliferative area was acquired using a Nikon Eclipse E600 microscope with 0.25 aperture and a QImaging Micropublisher 5.0 RTV camera equipped with a Sony ICX282 CCD, finally it gives us 24-bit color pictures with a resolution of 1920×2560 pixels. All of the three evaluation cell datasets have their dotted annotation available, which represents the location of cells as shown in Fig. 5. For the three datasets, we randomly select images for training and testing. Details of the three evaluation datasets are summarized in Table 1.

3.2 Implementation Details

The proposed method is implemented in Matlab, and we utilize Caffe [33], a fully open source implementation of Convolutional Neural Network, which affords clear access to Matlab/Python with support for GPU computation. As discussed in image partion part, each original RGB images is partitioned under certain rotation, stride size, and patch size. After taking experiments under different settings, we use stride size = 30 (pixels), patch size = 60×60 (pixels) and rotation step = 30 for the nuclei data; stride size = 20 (pixels), patch size = 40×40 (pixels) and rotation step = 30 for the bacterial data; stride size = 50 (pixels), patch size = 200×200 (pixels) and rotation step = 30 for the Ki-67 cell data. All the experiments are run on a machine with Intel Core i7-4790K CPU@4.00 GHz $\times$ 8 and GPU GeForce GTX TITAN Black/PCIe/SSE2.

3.3 Evaluation Metric and Counting Examples

In all the experiments, we use the Mean Relative Error (MRE) and Mean Absolute Error (MAE) as the metric for quantitative evaluation: where N is the total number of test images, t_i and p_i are the true and predicted numbers of cells in the i-th test image. MRE and MAE are defined as follows:

$$MRE = \frac{1}{N}\sum_{i=1}^{N}\frac{|t_i - p_i|}{t_i} \tag{3}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|t_i - p_i| \tag{4}$$

3.4 Counting Performance Using Different Models

First, we are going to investigate the performance difference between Classification (C) model and Regression (R) model for the cell counting task. The whole framework follows the pipeline shown in Fig. 3. As for the CNN architecture, we employ AlexNet (5conv + 3fc) and ResNet (50 layers) separately. And softmax loss function and Euclidean loss function are used respectively in the classification and regression model. We conduct this comparison experiment on all the three cell datasets and evaluate the performance in terms of Mean Relative Error and Mean Absolute Error (std also provided). Table 2 shows that on all the three datasets regression model shows lower prediction error by considerable margins than the classification model, which experimentally support the discussion in Sect. 2.3.1. It is also necessary to note that the AlexNet based regerssion model outperforms the ResNet based classification model. Table 3 shows the counting performances using ResNets with different number of layers. The 50/101/152-layer ResNet based regression models are used in this experiment. We can observe that ResNet-152 model shows the lowest prediction error, followed by ResNet-101 and ResNet-50 respectively.

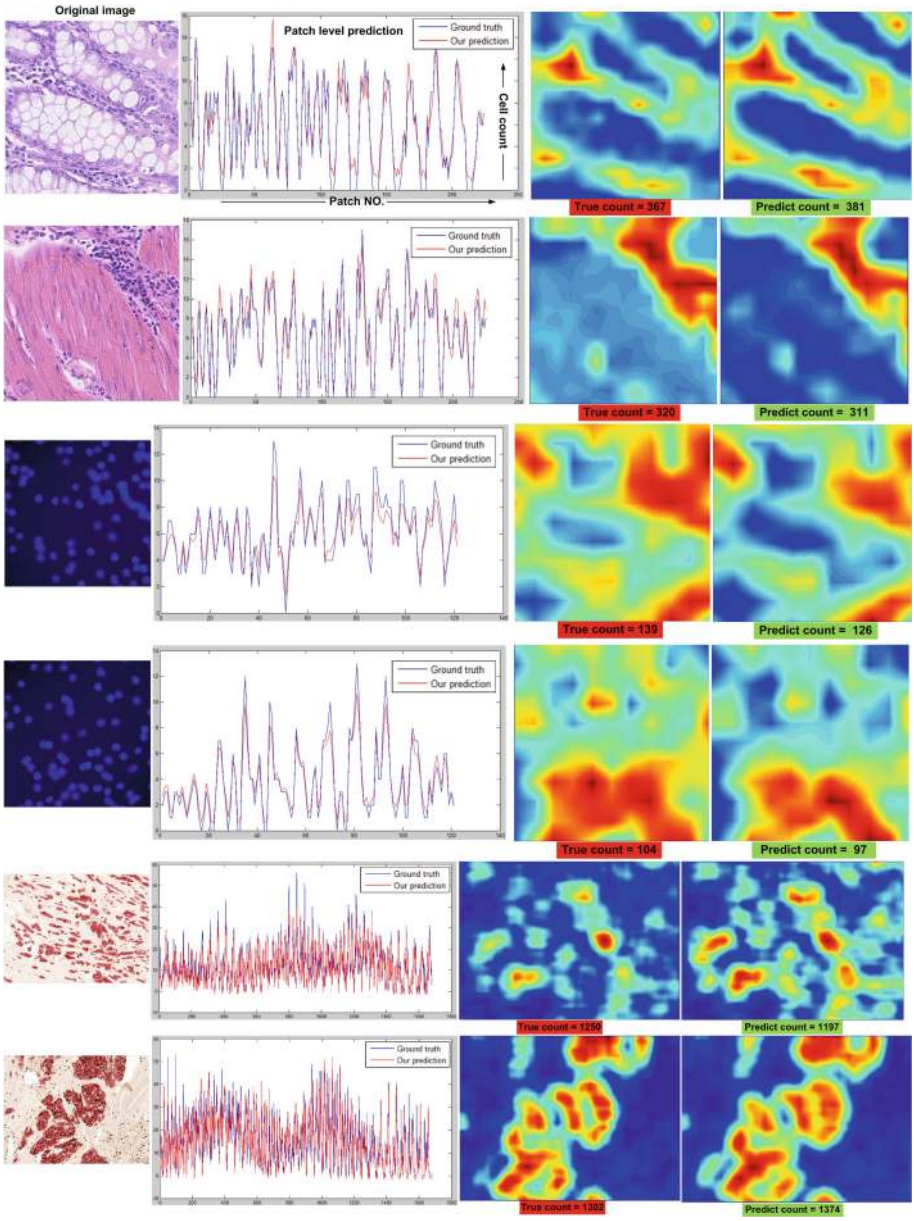

Fig. 6. Spatial density prediction and counting results on the cell datasets. Figure 6 provides several cell counting results on the three evaluation datasets. Original cell image is shown in on left side; the middle panel shows the patch level prediction, which is a middle result of our cell counting result; The right panel shows the spatial density prediction map (measured in number/square pixel) as well as the global counts of ground-truth and our prediction. From the patch level prediction curve, we can see that our estimated counts for each patch approximate the pattern of ground truth counts well.

Table 2. Counting performance in terms of MRE and MAE±std comparison between Classification (C) and Regression (R) model

MRE	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
AlexNet(C)	0.2175	0.0918	0.1226
AlexNet(R)	0.2019	0.0651	0.0959
ResNet(C)	0.2104	0.0772	0.1170
ResNet(R)	0.1925	0.0539	0.0775
MAE±std	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
AlexNet(C)	20.7636±13.9416	12.9667±4.5361	213.5302±65.7220
AlexNet(R)	18.5720±12.6055	9.2591±3.3142	151.2059±44.6032
ResNet(C)	19.8742±13.5217	11.8711±3.8247	169.5076±50.2124
ResNet(R)	17.1437±11.5073	8.2064±2.8515	128.7426±40.5621

Table 3. Counting performance in terms of MRE and MAE±std using different models.

MRE	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
ResNet-50(R)	0.1925	0.0539	0.0775
ResNet-101(R)	0.1845	0.0507	0.0697
ResNet-152(R)	0.1666	0.0450	0.0641
MAE±std	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
ResNet-50(R)	17.1437±11.5073	8.2064±2.8515	128.7426±40.5621
ResNet-101(R)	16.3164±10.8762	7.7542±2.4580	116.3076±39.0215
ResNet-152(R)	14.9275±10.4368	7.4741±2.2248	108.3014±40.4698

3.5 Comparison with State of the Art

We carry out experimental performance comparison between our method and three other state-of-the-art approaches (presented in [5,16,24]) on three evaluation datasets. The counting result from ResNet-152 regression model is used as our approach result during this comparison. Figure 7 provides the cell counts of ground-truth and four predictions of "Le.count" [16], "Le.detect" [5], "DeepFeat" [24] and "The proposed" on every test image. To quantify Fig. 7, Table 4 reports the performance in terms of Mean Relative Error (MRE) and Mean Absolute Error (MAE) over the three evaluation datasets.

The proposed method has achieved very competitive result on Nuclei dataset and Ki67 dataset, MRE = 16.66 % and 6.41 % respectively. The images from Nuclei and Ki67 datasets contain 310.22 and 2045.85 cells on average, our proposed method is able to predict with only 14.93 and 108.30 cells in terms of mean absolute error; while other three methods gives 33.89–71.80 and 189.35–259.67 error cells on average.

Table 4. Counting performance (MRE) and (MAE±std) comparison on the three evaluation datasets.

MRE	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
DeepFeat [24]	0.3581	0.1751	0.1249
Le.count [16]	0.2674	**0.0208**	0.1151
Le.detect [5]	0.3206	0.1083	0.1540
The proposed	**0.1666**	0.0450	**0.0641**
MAE±std	Nuclei-dataset	Bacterial-dataset	Ki67-dataset
DeepFeat [24]	71.8046 ± 51.4109	25.4792 ± 19.1504	189.3559 ± 53.6329
Le.count [16]	51.4479 ± 39.8087	**6.4061 ± 3.5657**	185.9391 ± 60.5042
Le.detect [5]	33.8995 ± 23.9252	18.1937 ± 13.4393	259.6736 ± 85.0594
The proposed	**14.9275 ± 10.4368**	7.4741 ± 2.2248	**108.3014 ± 40.4698**

On Bacterial dataset, the proposed method gives MRE = 4.50 %, but [16] makes 2.4 % further improvement over our result. The central idea of [16] is to estimate a density function whose integral over any image region gives the count of objects within that region. In its learning phase, each cell is dot-annotated and is assigned a real-valued Sift feature vector describing the local appearance. It means that for each cell, [16] needs its x-y coordinate on an image and then compute the Sift feature on the image sub-region around this cell. In comparison, the proposed method only takes the number of cells as annotation to an image patch during training. As one can imagine, for an image containing hundreds to thousands of cells, the complexity and time consuming of [16] will increase greatly. Furthermore, when it comes to the much more dense datasets (Nuclei and Ki67), the Sift descriptor based learning of [16] becomes less reliable.

It is also necessary to mention that the MRE values of all the four methods on Nuclei dataset are higher than those on other two datasets, because Nuclei dataset has several test images, which only contains a few cells e.g. 1, 4 or 8. For example, predicting the cell count from 1 (ground truth) to 2 or from 4 (ground truth) to 6 will greatly affect the final MRE value.

4 Conclusions

In this paper, we propose a novel regression based framwork for cell counting. As the output, spatial density map and global cell count are provided. The proposed method is able to handle dense cell microscopy image, where the cells also present huge variation in appearance. We have experimentally demonstrated that the proposed approach achieved superior performance compared with several recent related methods.

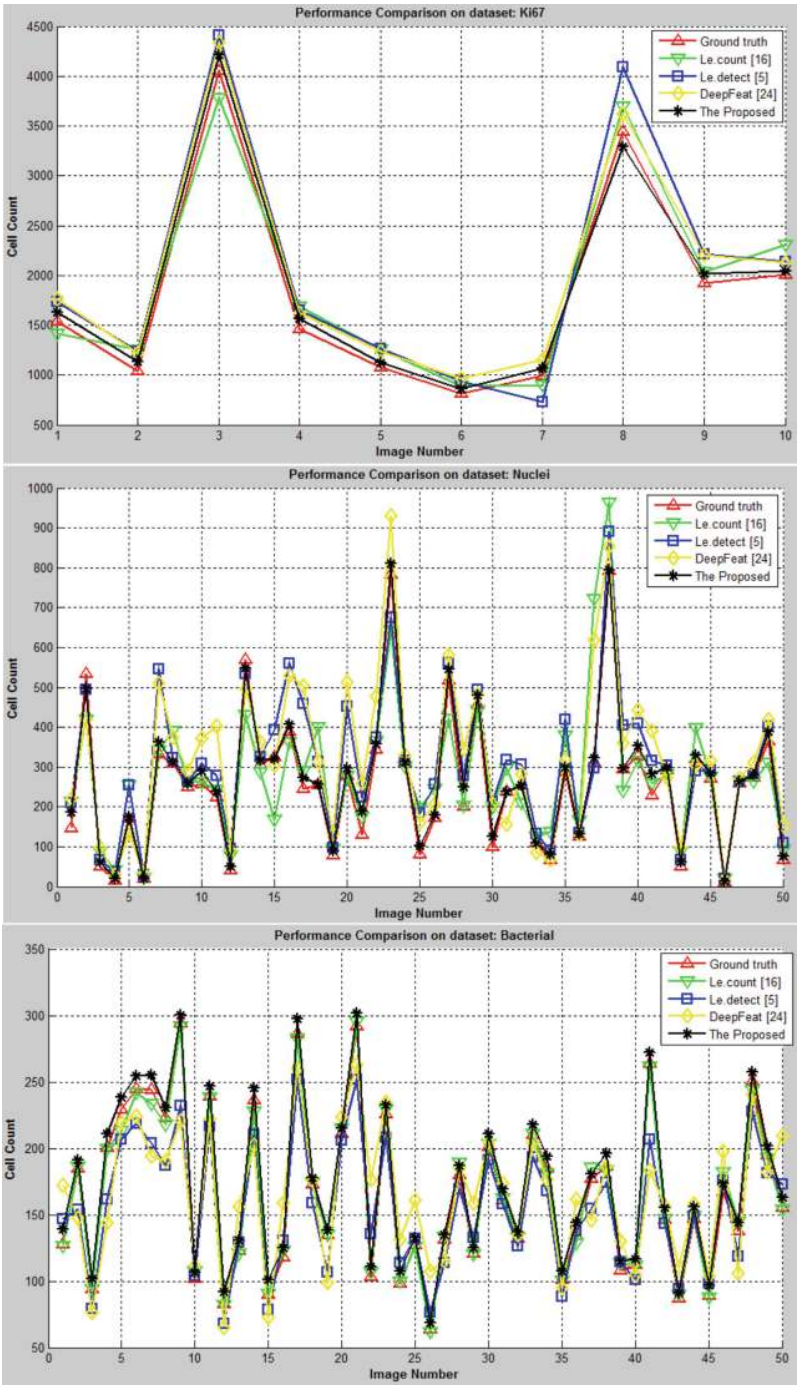

Fig. 7. The estimated count versus ground-truth of different approaches on the three evaluation datasets.

References

1. http://www.robots.ox.ac.uk/vgg/research/counting/
2. Goldhirsch, A., Gelber, R.D., Gnant, M., Piccart-Gebhart, M., Thrlimann, B., Coates, A.S., Winer, E.P., Senn, H.-J.: Tailoring therapies - improving the management of early breast cancer: St. gallen international expert consensus on the primary therapy of early breast cancer 2015. Ann Oncol first published online 4 May 2015. doi:10.1093/annonc/mdv221
3. Li, H., Zhang, C., Wang, X: Cross-scene crowd counting via deep convolutional neural network. In: Computer Vision and Pattern Recognition (CVPR) (2015)
4. Rimm, D.L., Camp, R.L., Chung, G.G.: Automated subcellular localization and quantification of protein expression in tissue microarrays. Nat. Med. **8**, 1323–1327 (2002)
5. Arteta, C., Lempitsky, V., Noble, J.A., Zisserman, A.: Learning to detect cells using non-overlapping extremal regions. In: Ayache, N., Delingette, H., Golland, P., Mori, K. (eds.) MICCAI 2012. LNCS, vol. 7510, pp. 348–356. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33415-3_43
6. Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., Szegedy, C., Liu, W.: Going deeper with convolutions. In: Computer Vision and Pattern Recognition (2014)
7. Yang, L., Liu, S., Cao, X., Wang, C., Zhang, H.: Deep people counting in extremely dense crowds. In: ACM International Conference on Multimedia (2015)
8. Cireşan, D.C., Giusti, A., Gambardella, L.M., Schmidhuber, J.: Mitosis detection in breast cancer histology images with deep neural networks. In: Mori, K., Sakuma, I., Sato, Y., Barillot, C., Navab, N. (eds.) MICCAI 2013. LNCS, vol. 8150, pp. 411–418. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40763-5_51
9. Pathak, D., Krahenbuhl, P., Darrell, T.: Constrained convolutional neural networks for weakly supervised segmentation. In: ICCV (2015)
10. Hart, N.S., Collin, S.P., Garza-Gisholt, E., Hemmi, J.M.: A comparison of spatial analysis methods for the construction of topographic maps of retinal cell density. PLoS One **9**(4), e93485 (2014)
11. Cantaloni, C., Eccher, C., Bazzanella, I., Aldovini, D., Bragantini, E., Morelli, L., Cuorvo, L.V., Ferro, A., Gasperetti, F., Berlanda, G., Dalla Palma, P., Fasanella, S., Leonardi, E.: Proliferative activity in human breast cancer: Ki-67 automated evaluation and the influence of different ki-67 equivalent antibodies. Diagn. Pathol. (2011)
12. Tsang, Y.W., Cree, I.A., Snead, D.R.J., Rajpoot, N.M., Sirinukunwattana, K., Raza, S.E.A.: Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. IEEE Trans. Med. Imaging (2016)
13. Ren, S., Sun, J., He, K., Zhang, X.: Deep residual learning for image recognition. In: CVPR (2015)
14. Sutskever, I., Krizhevsky, A., Hinton, G.E., Imagenet classification with deep convolutional neural networks. In: Neural Information Processing Systems, pp. 1097–1105 (2012)
15. Lebanoff, L., Idrees, H.: Counting in dense crowds using deep learning. In: CRCV (2015)
16. Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: Neural Information Processing Systems (NIPS) (2010)
17. Lin, Z., Davis, L.S.: Shape-based human detection and segmentation via hierarchical part-template matching. IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI) **32**, 604–618 (2010)

18. Liu, F., Yang, L.: A novel cell detection method using deep convolutional neural network and maximum-weight independent set. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 349–357. Springer, Heidelberg (2015). doi:10.1007/978-3-319-24574-4_42
19. Habibzadeh, M., Krzyżak, A., Fevens, T.: White blood cell differential counts using convolutional neural networks for low resolution images. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2013. LNCS (LNAI), vol. 7895, pp. 263–274. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38610-7_25
20. Sivic, J., Rodriguez, M., Laptev, I., Audibert, J.-Y.: Density-aware person detection and tracking in crowds. In: IEEE International Conference on Computer Vision (ICCV) (2011)
21. Kholi, P., Barinova, O., Lempitsky, V.: On detection of multiple object instances using hough transforms. IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI) **34**, 1773–1784 (2012)
22. Zisserman, A., Parkhi, O.M., Vedaldi, A.: Deep face recognition. In: BMVC (2015)
23. McShane, L.M., Gao, D., Hugh, J.C., Mastropasqua, M.G., Viale, G., Zabaglo, L.A., Penault-Llorca, F., Bartlett, J.M., Gown, A.M., Symmans, W.F., Piper, T., Mehl, E., Enos, R.A., Hayes, D.F., Dowsett, M., Nielsen, T.O., Polley, M.Y., Leung, S.C.: An international ki67 reproducibility study. J. Natl. Cancer Inst. **105**(24), 1897–1906 (2013)
24. Pujol, O., Seguí, S., Vitrià, J.: Learning to count with deep object features. In: Computer Vision and Pattern Recognition (CVPR) (2015)
25. Szegedy, C., Ioffe, S.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
26. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
27. Subburaman, V.B., Descamps, A., Carincotte, C.: Counting people in the crowd using a generic head detector. In: IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance (AVSS), pp. 470–475 (2012)
28. Messinger, J.D., Zhang, T., Bentley, M.J., Gutierrez, D.B., Ablonczy, Z., Smith, R.T., Sloan, K.R., Curcio, C.A., Ach, T., Huisingh, C., McGwin Jr., G.: Quantitative autofluorescence and cell density maps of the human retinal pigment epithelium. Invest. Ophthalmol. Vis. Sci. **55**(8), 4832–4841 (2014)
29. Tota, K., Idrees, H.: Counting in dense crowds using deep features. In: CRCV (2015)
30. Wang, M., Wang, X.: Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
31. Zisserman, A., Xie, W., Noble, J.A.: Microscopy cell counting with fully convolutional regression networks (2015)
32. Wu, B., Nevatia, R.: Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In: IEEE Computer Society Conference on IEEE International Conference on Computer Vision (ICCV) Vision and Pattern Recognition (CVPR) (2005)
33. Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T., Jia, Y., Shelhamer, E.: Caffe: convolutional architecture for fast feature embedding. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)

34. Zhang, X., Chen, Y.: Study of cell behaviors on anodized tio 2 nanotube arrays with coexisting multi-size diameters. Nano-Micro Lett. **8**, 61–69 (2015)
35. Xie, Y., Xing, F., Kong, X., Su, H., Yang, L.: Beyond classification: structured regression for robust cell detection using convolutional neural network. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 358–365. Springer, Heidelberg (2015). doi:10.1007/978-3-319-24574-4_43