Feature Augmented Deep Neural Networks for Segmentation of Cells

Sajith Kecheril Sadanandan[1,2(✉)], Petter Ranefall[1,2], and Carolina Wählby[1,2]

[1] Department of Information Technology, Uppsala University, Uppsala, Sweden
sajith.ks@it.uu.se
[2] SciLifeLab, Uppsala, Sweden

Abstract. In this work, we use a fully convolutional neural network for microscopy cell image segmentation. Rather than designing the network from scratch, we modify an existing network to suit our dataset. We show that improved cell segmentation can be obtained by augmenting the raw images with specialized feature maps such as eigen value of Hessian and wavelet filtered images, for training our network. We also show modality transfer learning, by training a network on phase contrast images and testing on fluorescent images. Finally we show that our network is able to segment irregularly shaped cells. We evaluate the performance of our methods on three datasets consisting of phase contrast, fluorescent and bright-field images.

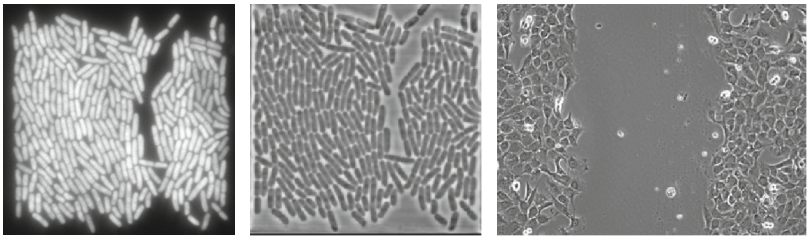
Keywords: Deep neural network · Feature augmentation · Cell segmentation · Convolutional neural network · Unstained cells

1 Introduction

Observation of biological samples over prolonged periods of time is commonly used to study phenotypical changes due to variations in environmental conditions or genetic modifications. High-throughput high-content screening is used to analyse many biological processes simultaneously [1]. It is tedious for human observers to monitor changes at the cellular level over long time. Automated image analysis algorithms are widely used to simplify and quantify the analysis process [2].

For automated image analysis at the cellular level, a commonly used approach is to segment the cellular regions and track the cell segments over time [3]. The cell segmentation is a crucial step in this process, which affects the quality of the cell tracking results. In this work, we aim to segment cells in time-lapse microscopy image sequences. Cell segmentation is a challenging process, especially when the cells are unstained. Deep Neural Networks (DNN) using Fully Convolutional Neural Networks (FCNN) have shown excellent results in semantic segmentation [4]. FCNNs were also used in segmenting unstained cells in microscopy images [5,6]. The network structures of these high performing FCNNs, as opposed to traditional DNNs [7,8], suggest that designing the proper

G. Hua and H. Jégou (Eds.): ECCV 2016 Workshops, Part I, LNCS 9913, pp. 231–243, 2016.
DOI: 10.1007/978-3-319-46604-0_17



(a) EFD (b) EPD (c) MBD

Fig. 1. (a) Input *E. coli* fluorescent dataset (EFD) (b) *E. coli* phase contrast dataset (EPD) and (c) mouse mammary cells bright-field dataset (MBD).

deep network is non-trivial. Often, a network that gives good results on a particular dataset may not give good results on another dataset. Fusing features from different layers of deep networks [9] or combining deep features with hand-crafted features [10] were used in video action recognition tasks, where the authors used 'late fusion', i.e., they combined features at later layers of the deep network for classification. A combination of Gabor filters and Convolutional Neural Networks was used for face detection [11], where the author performed an 'early fusion', by combining hand-crafted features in the first layer of the neural network. Recently, DNNs with a reduced number of parameters [12,13] were also successfully used for classification. In this work, we augment features at the first layer by combining hand-crafted features with raw images, and train an FCNN with a reduced number of parameters. We use wavelet filtering and eigen value based enhancement for feature augmentation. For one of our datasets we create ground-truth semi-automatically, using an existing method [14]. In Sect. 2, we describe how to modify a deep network with augmented features for two different datasets of *E. coli* cells and mouse cells. In Sect. 3, we perform a quantitative evaluation of our results with an existing method for the *E. coli* dataset and a qualitative evaluation of our results on the mouse cells dataset. We also show how the feature augmentation improves segmentation of the cells.

2 Materials and Methods

2.1 Input Images

The input images comprise two time-lapse datasets- (1) a prokaryotic cell dataset, consisting of *E. coli* and (2) a eukaryotic cell dataset, consisting of mouse mammary gland cells. The *E. coli* cell images were acquired using a phase contrast microscope and we call this dataset *E. coli* phase dataset (EPD) and the mouse cells were acquired using a bright-field microscope and we call it mouse bright-field dataset (MBD), hereafter. The EPD consists of 500 images of size 1024×1360 pixels in vertical and horizontal directions respectively, while the bright-field dataset consists of 411 images of size 1040×1392 pixels. These

two datasets are used for both training and testing of our FCNNs. In addition, we use a single image of the *E. coli* sample, imaged under a fluorescence microscope and we name this dataset *E. coli* fluorescent dataset (EFD), for additional testing of a modality that is different from the one used for training the FCNN. Sample images from the image datasets are shown in Fig. 1.

2.2 Image Preprocessing

The EPD contains regions outside the cell colony area. We therefore crop the images to a size of 860×860 pixels to set the field of view to cell regions alone. The images in the MBD, are not cropped as the raw images contain cells in the full field of view. We create three different types of feature maps depending on the dataset, such as eigen value based contrast enhancement [15], wavelet coefficient filtering [16] and truncated Singular Value Decomposition (SVD) [17]. The idea behind the feature augmentation is to highlight certain regions in the input images and make the network learn features from those regions resulting in improved segmentation of cellular regions. The choice of these features depends on the imaging modality and the type of cells imaged.

For the EPD, we use the eigen value based contrast enhancement and truncated SVD. The eigen value based contrast enhancement was successfully used in [3] to segment *E.coli* cells. This method improves the contrast between the regions of touching cells. The contrast enhancement using this approach helps the network to learn features for better cell segmentation than the network that does not use additional contrast enhancements, by accurately segmenting the touching cells. We use truncated SVD as denoising method and wanted to see if denoising has any impact on the segmentation results, especially for the EFD, which is more noisy than either the EPD or the MBD. The eigen value based contrast enhancement is done by extracting the minimum curvature of the intensity landscape in the input image. For this, first we find the Hessian matrix, H, which is created by finding the second derivative at each pixel position in the x and y directions of the image. The maximum and minimum curvatures (principal curvatures) at a position are the eigen values of the Hessian matrix. The eigen values are found by the following equation [15].

$$k_{1,2} = \frac{trace(H) \pm \sqrt{trace(H)^2 - 4 \times det(H)}}{2} \tag{1}$$

Here, k_1 and k_2 are the principal curvatures with $k_1 < k_2$. To perform contrast enhancement, we create an image with k_1, the lowest eigen value, from all spatial locations of the input image. To find the truncated SVD, we find singular values sorted in decreasing order, for the full raw image, and add the values till they sum upto 99 % of the sum of all singular values. All the singular values after 99 % are set to zero and an image is reconstructed from the new set of singular values. For the EPD, we use four different input combinations for the FCNN such as- (1) raw images, (2) eigen images, (3) combined raw and eigen images, and (4) truncated SVD images as summarized in Table 1.

Table 1. The inputs used for training. The EFD is not used for training

Dataset	Raw	Eigen	Raw + Eigen	SVD	Raw + Wavelet
EPD	yes	yes	yes	yes	-
MBD	yes	-	-	-	yes
EFD	-	-	-	-	-

For the MBD, we first find the wavelet transform using Daubechies 4 wavelet [16] (db4) to four decomposition levels and set the approximation coefficients to zero and then reconstruct the image using these modified coefficients. Two deep networks with raw images and a stack of combined raw images with wavelet filtered images as input are created for the MBD. We use open-source code available at [18] to find wavelet features. Table 1 shows the inputs used for training both the EPD and the MBD. All these input images were preprocessed by normalizing to the range $[0, 1]$ and subtracting the median value.

2.3 Semi-automatic Ground Truth Generation

Training data generation is one of the crucial steps for any FCNN application. We observed that the quality of the training set was equally important as the quantity. In this work, we employed two different strategies for training set generation. For the EPD, we generated the training set semi-automatically. We used the open-source code available at [14] to segment *E. coli* cells. Once the segmentation was finished, we manually removed the false positives to improve the quality of our training set. We observed that even a single false postive could adversely affect the results when testing the FCNN. This could be due to two reasons- (1) when false positives are present, the network learns parameters to detect foreground regions that actually have the features of the background, resulting in poor performance during testing. (2) since we perform data augmentation there is a high chance that the false positives are present in multiple training samples. We selected 30 images, equally spaced at regular intervals in the EPD consisting of 500 images, for training our FCNN. After selecting representative images, we set the input image size of the FCNN to 540×540 pixels through cropping. From every representative raw input image, we cropped 5 patches to cover the entire image region. The regions we cropped were such that the patches cover the top-left corner, bottom-left corner, top-right corner, bottom-right corner and the centre of the images. We created data augmentation using these representative images. The data augmentation step consisted of spatial transformations such as flipping, rotation and elastic deformations [5]. We did two experiments- (1) with the original U-Net and (2) with a modified U-Net. For the experiment with the original U-Net, we created a training set of 20000 images, while for the experiment with the modified U-Net, we created a training set of 600 images, followed by feature augmentation. Finally, we created weight maps to give additional weights to the foreground regions for weighted

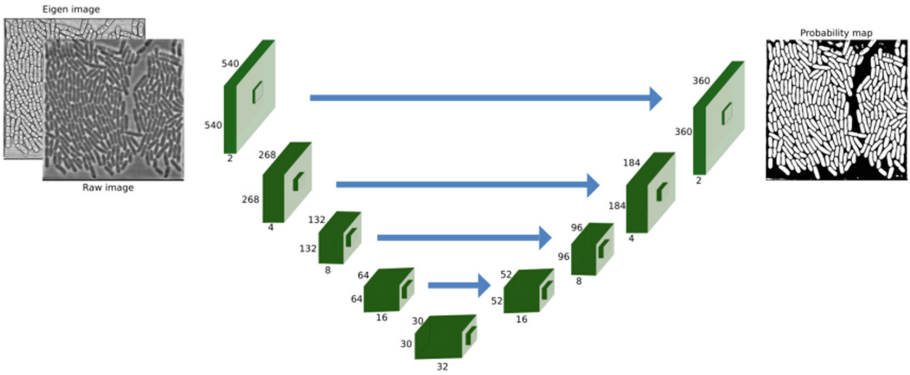


Fig. 2. Architecture of the network. Each block represents two convolutions with a kernel of size 3×3 without padding. An input image with its augmented feature map and the corresponding output probability map is also shown. The feature map is contrast enhanced for better visualization.

softmax [5]. Additional weights force the network to learn parameters in such a way that it can separate touching cells. We set the border weight value to 10 for the feature augmented networks and 3 for the single input image networks and standard deviation to 3 for all the networks to create the weighted labels.

For the MBD, there is no previously existing method that gives a satisfactory cell segmentation, to the best of our knowledge. Therefore semi-automated ground truth generation cannot be used in this case. We manually marked the boundary of each cell in two images. There were approximately 300 cells per image. Then, by data augmentation, 600 training samples were generated from these images, followed by feature augmentation. The rest of the processing pipeline is similar to the one used for the EPD.

2.4 Deep Neural Network Architecture and Training

The FCNN we use in this work, is inspired by the recent FCNN known as U-Net [5]. Initially we trained the original U-Net [5] architecture using 20000 training images, created through data augmentation, for 200000 training iterations. During testing, we found that the network was under-performing and the final segmentation result was poor for our dataset. This may be because of the large number of parameters in the network model and also that our dataset may be lying in a lower dimensional feature space. The details regarding these results are given in Sect. 3.1. Next, we modified the original network structure and combined it with traditional image processing techniques to improve performance on our dataset. We modified the original architecture by reducing the feature map size, i.e. number of featuremaps, to 1/32 of the original size. For example, the original U-Net has a feature map size of 64 in the first layer while our modified network has two feature maps in the first layer. The architecture of the network, a raw image, a feature map, and an output probability map is shown in Fig. 2.

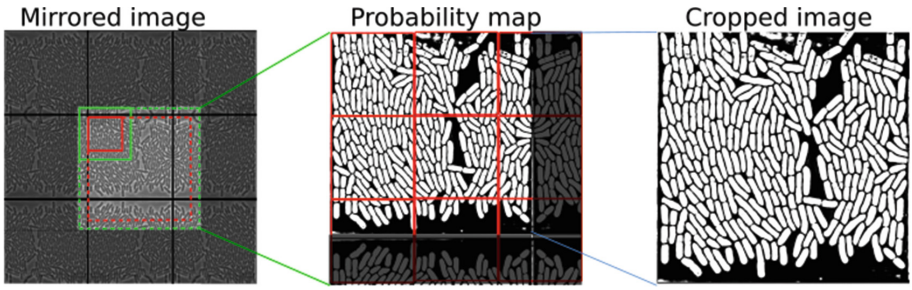
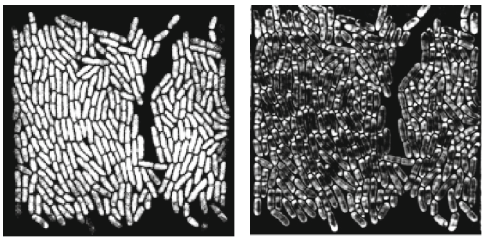


Fig. 3. Overlap-tiling to find the probability map of the entire image. The raw image or the stack of raw images with augmented feature maps is mirrored on all sides to take care of the boundary problem. Then patch-wise probability maps are created to cover the entire image and finally the probability map is cropped to the size of the raw input image.

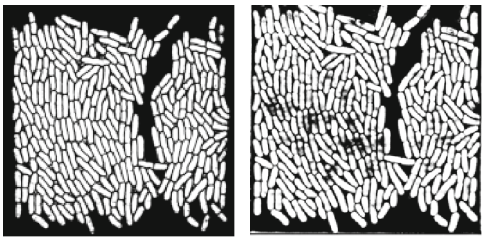
Here, we reduced the training set size to 600 images. We used the following hyperparameters for training: the number of training iterations was set to 20000 with base learning rate to 0.01 and momentum to 0.9 and reduced the learning rate to 1/10 of the current value after every 5000 iterations. We used the open-source framework Caffe [19], with an additional weighted softmax loss layer and a crop layer, to implement the FCNN. We trained all different networks on a workstation with Intel(R) Core(TM) i7-5930K CPU running at 3.50 GHz and a Nvidia Titan X GPU.

2.5 Segmentation and Postprocessing

For the EPD, testing was done on the whole time-lapse sequence comprising of 500 images, to quantify its usability for cell tracking applications. The final network had an input size of 540×540 or $540 \times 540 \times 2$, depending on network architecture, and the output probability map size was 356×356. So to create a probability map for the entire image, we first created an image that was 9 times the size of the original image by mirroring on all eight neighborhood positions of the image followed by feature map generation. After the mirroring step, we traversed through the whole image, in an overlap-tile strategy of input, in such a way that the output probability maps did not overlap, similarly as in [5]. The overlap-tiling along with cropping of the probability map is shown in Fig. 3. For the EPD, we empirically found a threshold of 0.2 for binarization and removed objects that were smaller than 200 pixels to eliminate false positive pixels. Finally, we filled holes in the output mask to get a final segmentation mask. For the MBD, a similar procedure was followed to create a probability map. The probability map for the MBD was not as sharp as the one for the EPD. We therefore applied watershed segmentation to the probability map to find the final segmentation mask. We used the openly available CellProfiler software [20] for watershed segmentation. In CellProfiler, we set the parameters minimum and



(a) EFD, Orig. U-Net (b) EPD, Orig. U-Net



(c) EFD, Mod. U-Net (d) EPD, Mod. U-Net

Fig. 4. Probability maps from the original U-Net and the modified U-Net when raw images were used as input. (a) and (b) show the probability maps for the EFD and the EPD, respectively, for the original U-Net. (c) and (d) show the corresponding probability maps for the modified U-Net with architecture shown in Fig. 2.

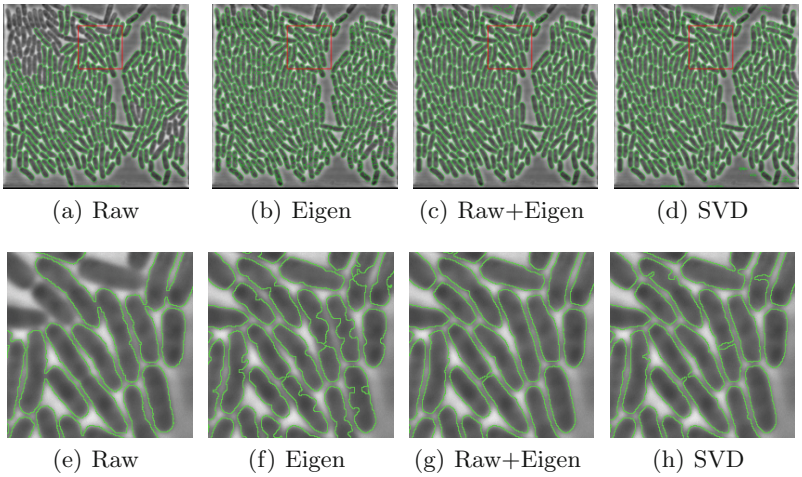
maximum diameter of objects to 30 and 100 respectively, for the segmentation, and kept other parameters at their default values.

2.6 Modality Transfer Learning

Finally, we investigated if a trained network can be transfered to a new imaging modality, in this case moving from phase contrast to fluorescence microscopy. For the EFD, we re-used the network trained on the EPD. We inverted the input image, so that the cells appeared as dark objects and the background was bright. After inverting the image, we created feature augmentation and fed to the corresponding network trained on the EPD. This way we were able to re-use the same network for testing on a different modality, that was not used for training.

3 Results and Discussions

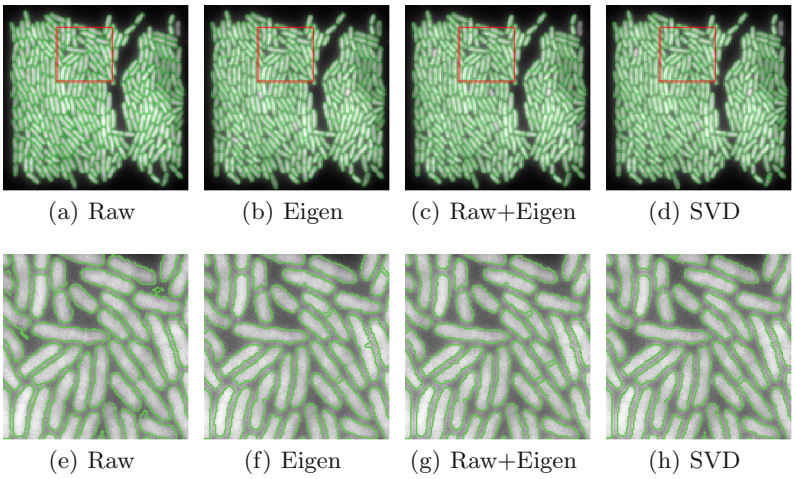
In this section, we show the results obtained from the modified network for our image datasets.



(a) Raw (b) Eigen (c) Raw+Eigen (d) SVD

(e) Raw (f) Eigen (g) Raw+Eigen (h) SVD

Fig. 5. (a–d) Segmentation result overlayed on the EPD (e–h) zoomed-in regions of corresponding segmentation results on highlighted (red) regions from (a–d). (Color figure online)



(a) Raw (b) Eigen (c) Raw+Eigen (d) SVD

(e) Raw (f) Eigen (g) Raw+Eigen (h) SVD

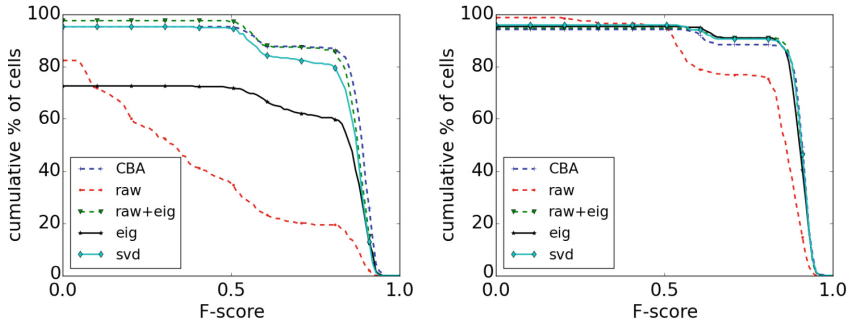
Fig. 6. (a–d) Segmentation result overlayed on the EFD (e–h) zoomed-in regions of corresponding segmentation results on highlighted (red) regions. (Color figure online)

3.1 Deep Neural Network

A comparison of results from the original U-Net and our modified network for raw input images is shown in Fig. 4. Visual inspection of these results shows that the modified network performs better than the original one for our dataset. This might be attributed to the small number of parameters for the network and low feature dimensionality of our dataset.

Table 2. The average F-score ± standard deviation for the previously published CBA method and the here proposed methods using raw images, eigen images, raw images with eigen images and truncated SVD images for the EPD and the EFD

Dataset	CBA	Raw	Eigen	Raw + Eigen	SVD
EPD	0.81 ± 0.27	0.37 ± 0.24	0.60 ± 0.21	$\mathbf{0.82 \pm 0.29}$	0.78 ± 0.28
EFD	0.83 ± 0.25	0.78 ± 0.29	0.84 ± 0.26	0.84 ± 0.25	$\mathbf{0.85 \pm 0.25}$

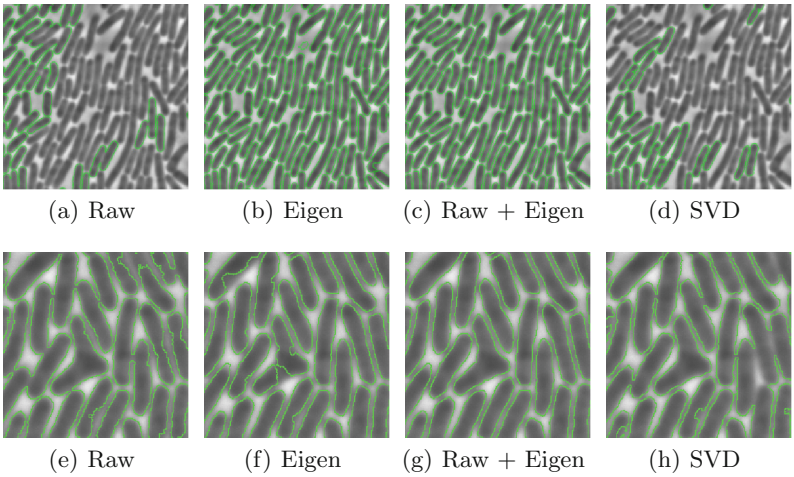


(a) EPD (b) EFD

Fig. 7. Percentage of cells above particular F-score value v/s F-score for the EPD and the EFD.

3.2 Segmentation Evaluation

The segmentation results for the EPD and the EFD for different networks are shown in Figs. 5 and 6 respectively. Zoomed-in regions of highlighted areas (in red) are also shown in these figures. For the EPD, a performance improvement for the proposed method can be seen from the results of our feature augmented networks, giving comparatively better results than single input networks as shown in Fig. 5. For the EFD, qualitative evaluation of the results shows that similar performance is obtained for all networks as shown in Fig. 6. Next we did a quantitative evaluation of the segmentation result on the EPD and the EFD. We compared all the segmentation results with the corresponding ground truth images. The results of the comparison are shown in Fig. 7(a) for the EPD and Fig. 7(b) for EFD. The average F-score value per cell, together with standard deviations for the EPD (288 cells) and the EFD (308 cells) with respect to the different methods are shown in Table 2. The evaluation on the EFD shows that our proposed method, using feature augmentation, is comparable to the state-of-the-art method, refered to as CBA [3]. The evaluation on the EPD shows that the feature augmentation based deep network using the raw image + the eigen image is better than the deep networks using either the raw image or the eigen image alone. Furthermore we found that our proposed method is better than the previous method for detecting irregularly shaped cells. The segmentation results for two images with irregularly shaped cells are shown in Fig. 8. The results show



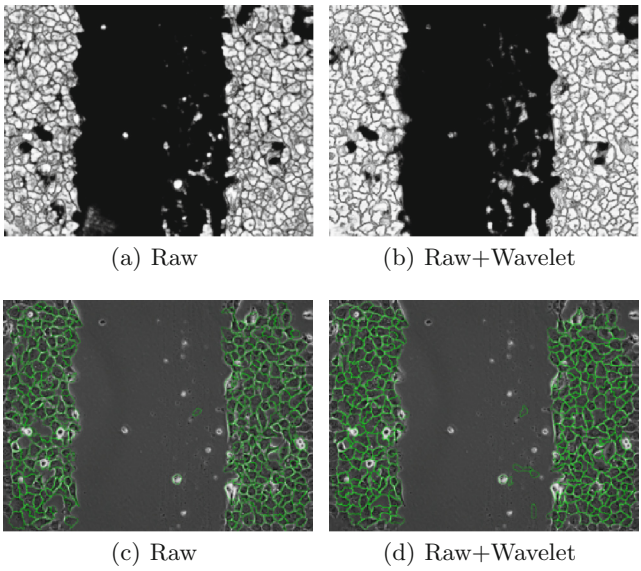
(a) Raw (b) Eigen (c) Raw + Eigen (d) SVD

(e) Raw (f) Eigen (g) Raw + Eigen (h) SVD

Fig. 8. (a–d) show segmentation results, on a selected region, using our networks when an unusually long cell is present in the input image. Similarly (e–h) show results when an irregular shaped cell is present in the input image.

that our method can detect *E. coli* cells of any length or other abnormalities. We observed that the networks with raw image and truncated SVD image as input had a low perfomance because the regions between the cells had a high value in the output probability map and were detected as foreground regions after thresholding. It is worth noting that the truncated SVD gave better results than the raw image. This might be due to the reduction in noise while doing SVD truncation. We observed that 99 % of the sum of all singular values gave the best performance, when the sum was varied from 80 % to 99 % of the total sum of all singular values. Since the truncated SVD acts as a denoising filter, it might be possible to use other denoising filters to achieve similar performance. Qualitative analysis on the MBD showed that the feature augmented network, using wavelet filtering, gave visibly improved results as compared to the network using only the raw images. The probability maps are shown in Fig. 9(a) and (b). Comparing the two results we can see that the regions inside the cells are brighter, indicating high probability of being cells, while the boundaries are dark. We did watershed segmentation on the probability maps to get a final segmentation mask. The final segmentation mask overlayed on raw input image is shown in Fig. 9(c) and (d). The results from the MBD also showed that improved performance can be obtained using feature augmentation.

We compared the execution speed of the CBA method with our deep neural network approach. A direct comparison of the CBA method with the proposed method is not possible since CBA is a CPU based algorithm while the proposed method is GPU based. For the comparison, we used the faster version of CBA [14]. We found that the CBA method took 1.86 s to segment the EPD of size 860×860 on a laptop with quad core Intel(R) Core(TM) i7 CPU running at



(a) Raw (b) Raw+Wavelet

(c) Raw (d) Raw+Wavelet

Fig. 9. Segmentation result of the MBD. Probability maps of results using the raw image alone and using the raw image along with feature augmentation using wavelet filtering is shown in (a) and (b) and corresponding segmentation results using the watershed algorithm on the probability maps are shown in (c) and (d). Training and testing was done on separate images.

2.7 GHz with 8 Gb RAM on Ubuntu 14.04. The proposed approach took 0.91 s to segment the same image on a workstation with six core Intel(R) Core(TM) i7 CPU running at 3.50 GHz with 32 Gb RAM and a Nvidia Titan X GPU on Ubuntu 14.04.

4 Conclusion and Future Work

In this work, we have modified an existing FCNN and augment the input layer with hand-crafted features to improve the performance. We used an existing method to generate the ground truth semi-automatically, for training our FCNNs. We showed that modality transfer learning is possible by training the FCNN on one imaging modality, such as phase contrast microscopy images and test on a different modality, such as fluorescence microscopy images. We also showed that our proposed feature augmentation technique improved the segmentation of cells on three different datasets. The previous state-of-the-art method (CBA) fails in finding cells that do not have an elliptical shape, while the proposed FCNN does not have such restrictions. It should be noted that these irregularly shaped cells were not part of the training set, and we believe that the success is due to the ability of the FCNN to identify the local structures that enables better segmentation of individual cells. In the future, we plan to

use other features for improved segmentation accuracy for unstained cultured cells and use the segmentation results for cell tracking applications.

Acknowledgements. This work was supported by the Swedish research council under Grant 2012-4968 (to CW) and the Swedish strategic research program eSSENCE. Image data was kindly provided by Johan Elf at the Department of Cell and Molecular Biology, Computational and Systems Biology, Uppsala University, Sweden and Theresa Vincent at the Department of Physiology and Pharmacology, Karolinska Institutet, Sweden.

References

1. Ishii, N., Nakahigashi, K., Baba, T., et al.: Multiple high-throughput analyses monitor the response of E. coli to perturbations. Science **316**(5824), 593–597 (2007)
2. Lin, S.C., Yip, H., Phandthong, R., Davis, B., Talbot, P.: Evaluation of dynamic cell processes and behavior using video bioinformatics tools. In: Bhanu, B., Talbot, P. (eds.) Video Bioinformatics. CB, vol. 22, pp. 167–186. Springer, Heidelberg (2015). doi:10.1007/978-3-319-23724-4_9
3. Sadanandan, S.K., Baltekin, Ö., Magnusson, K.E.G., et al.: Segmentation and track-analysis in time-lapse imaging of bacteria. IEEE J. Sel. Topics Signal Process. **10**(1), 174–184 (2016)
4. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015)
5. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Heidelberg (2015). doi:10.1007/978-3-319-24574-4_28
6. Chen, H., Qi, X.J., Cheng, J.Z., et al.: Deep contextual networks for neuronal structure segmentation. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 1167–1173 (2016)
7. Lecun, Y., Bottou, L., Bengio, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105. Curran Associates, Inc. (2012)
9. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. arXiv:1604.06573v1 (2016)
10. Wang, P., Li, Z., Hou, Y., et al.: Combining convnets with hand-crafted features for action recognition based on an HMM-SVM classifier. arXiv:1602.00749v1 (2016)
11. Kwolek, B.: Face detection using convolutional neural networks and gabor filters. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3696, pp. 551–556. Springer, Heidelberg (2005). doi:10.1007/11550822_86
12. Liu, B., Wang, M., Foroosh, H., et al.: Sparse convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 806–814, June 2015
13. Iandola, F.N., Moskewicz, M.W., Ashraf, K., et al.: Squeezenet: alexnet-level accuracy with 50x fewer parameters and <1mb model size. arXiv:1602.07360v3 (2016)

14. Sadanandan, S.K.: CBA segmentation. https://bitbucket.org/sajithks/fastcba/. Accessed 21 May 2016
15. Woodford, C., Philips, C.: Numerical Methods with Worked Examples, Matlab edn. Springer, New York (2012)
16. Daubechies, I.: Ten Lectures on Wavelets. SIAM, vol. 61. Springer, New York (1992)
17. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. J. Soc. Ind. Appl. Math. Ser. B: Numer. Anal. **2**(2), 205–224 (1965)
18. Wasilewski, F.: Pywavelets. http://www.pybytes.com/pywavelets/. Accessed 21 May 2016
19. Jia, Y., Shelhamer, E., Donahue, J., et al.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
20. Carpenter, A.E., Jones, T.R., Lamprecht, M.R., et al.: Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. Genome Biol. **7**, R100 (2006)