Review of “Microscopy cell counting and detection with fully convolutional regression networks”

MadDeep: Abubakar Abid and Alex Barraon

Summary

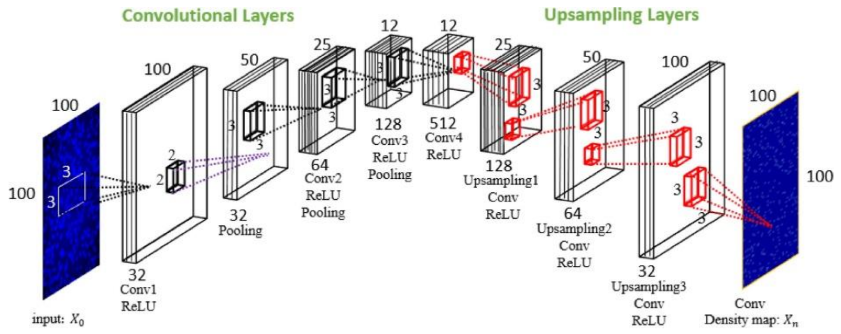
In this paper, Xie et al. tackle the problem of counting cells in microscopy images, a common task in pathology and microbiology. The challenge of counting cells in images arises because they are often very tightly clumped, with some cells overlapping and obscuring others. This makes the problem difficult to solve by typical detection and segmentation techniques (Arteta, 2015). Instead Xie et al. take the approach of estimating a density map from the microscopy image. The density map is an estimate of how much “cell” is in a particular area, which can then be easily integrated across various areas to estimate the number of cells in that region. The authors make 3 further contributions beyond this approach: (1) they examine two different CNN architectures for estimating the density map and explore the tradeoffs between the two (2) they demonstrate that synthetic data, which is easy to generate and label, can be used to train the model effectively, even for counting natural images (3) They propose a way to invert representations of density map to images, allowing the visualization of information flow through the CNN architecture.

Context and Related Literature

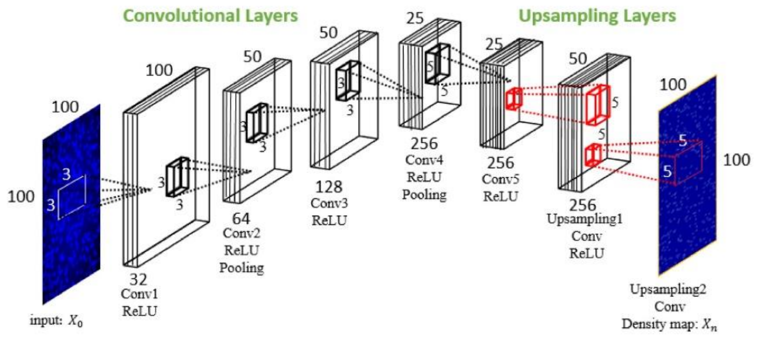
Using density maps to count objects without segmentation and detection has been proposed by several other researchers before Xie et al. Both (Fiaschi, 2012) and (Arteta, 2014) use similar approaches, though neither are deep-learning based methods. The former uses a random regression forests to generate density maps, while the latter uses an innovative cost function that updates as users interactively annotate images of cell. The closest article to the current paper, and seemly the only prior deep-learning based approach to generating density maps, is proposed by (Yuanpu, 2015), who train traditional CNNs on images labeled with global counts to create density maps. Their approach is limited to images of fixed sizes, as opposed to Xie et al., whose method is scalable to images of any size.

Methodology

Xie et al. use fully-convolutional neural networks, with regression outputs (instead of the classifier outputs that we have mostly seen in class) to generate density maps from inputs. The authors explore two very similar architectures, which they dub FCRN-A and FCRN-B. Both architectures consist solely convolutional layers (with ReLU activations), 2x2 pooling layers, as well as up-sampling convolutional layers (with ReLU activations). In each architecture, convolution and pooling is used to identify cells in different parts of the images, and then this information is up-sampled to generate a density map of the same resolution as the original image. The difference between the two architectures is the kernel size and the number of layers: the kernel size of all of the convolution layers in FCRN-A is 3x3, while FCRN-B includes 5x5 up-sampling layers. This leads to FCRN-B having about 3 times as many parameters as FCRN-A, even though it has one fewer down-sampling and one fewer up-sampling layer. The cost function used in the regression is the mean-square between the predicted and annotated density maps. Both architectures are compared in the figure below:



(a) Fully Convolutional Regression Network - A (FCRN-A)



(b) Fully Convolutional Regression Network - B (FCRN-B)

Because these density maps are difficult to generate for natural images by hand, Xie et al. explore several possibilities of increasing the data. For example, they use simple data augmentation techniques to rotate and flip the images, as well as sample 100x100 images from larger, 256x256 images. But the most significant step they take to solve this problem is the generation of synthetic images. Unlike real images, synthetic images are cheap and easy to annotate. The authors use the synthetic dataset created by (Lempitsky, 2010), which consists of 200 cell images, to train and validate their model before testing the same model (with and without fine-tuning) on real data.

Experiments

The authors first report their results on the synthetic dataset. The authors trained their model on different sizes of training sets, with a 5-fold cross-validation. They found that performance depended on training size, with the best performance being with training size $N = 64$. The mean absolute error in the number of cells was 2.9 for FCRN-A and 3.2 for FCRN-B, out of a total of 176 cells on average per image.

For the real data, the authors reported results only from FCRN-A (even though they claimed they were getting better results from FCRN-B in some cases). The typical error rates were about 1-2%. On the one image that they tried fine-tuning, the error rate reduced from 2% to 1%.

Strangely, there are any benchmarks in the literature with real images – only synthetic images, so it is difficult to know how substantive of an improvement the authors' work was. On the synthetic dataset, the authors' best model had an average error rate of 2.9. The best results in literature (Fiaschi, 2012) reported an error rate of 3.2, so the authors claimed an incremental improvement of about 9%.

Possible Extensions and Future Work

While the authors found an innovative application of regression convolutional neural networks to estimating density maps and show impressive results based on training on synthetic images alone, the results here did not seem to substantially improve state of the art. There could be many reasons for this: 1) **the inherent difficulty in assigning a count to the number of cells** – it may be that it is difficult, even for humans, to tell whether there is one cell or a clump of cells in certain cases. Establishing a baseline for how much variation there is among human actors can tell us whether there is so much variation in "ground truth" that incremental improvements like these even matter 2) the **choice of architecture** – we believe that the authors do not sufficiently motivate their choice of architecture. The only motivation is the comparison to the VGG-net (Simonyan, 2015), which seems particularly strange, since that is a very different kind of architecture, with many more layers and classification instead of regression. Either there should be extensive hyper-parameter searching, or there should be more reasoning provided for the 2 architectures used. 3) **Fine-tuning**: we thought it very interesting that fine-tuning was able to reduce the error by about half on the real images. Not enough information was provided about how this fine-tuning was carried out. It would be interesting if a manually annotating a handful of cells in a large image could significantly improve the estimation of cell count in that image. Such a procedure could be easily integrated into the workflow of an automated counting system.

References

Arteta C, Lempitsky V, Noble JA, Zisserman A. 2014. Interactive object counting. In: Proceedings of the European Conference on Computer Vision (ECCV); Zurich, Switzerland. p. 504–518.

Arteta C, Lempitsky V, Noble JA, Zisserman A. 2015. Detecting overlapping instances in microscopy images using extremal region trees. Med Image Anal. Available from: http://dxdoiorg/101016/jmedia201503002

Fiaschi L, Nair R, Koethe U, Hamprecht FA. 2012. Learning to count with regression forest and structured labels. In: 21st International Conference on Pattern Recognition (ICPR); Tsukuba, Japan. p. 2685–2688.

Lempitsky V, Zisserman A. 2010. Learning to count objects in images. In: Advances in Neural Information Processing Systems (NIPS); Hyatt Regency, Vancouver, Canada. p. 1324–1332.

Simonyan K, Zisserman A. 2015. Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (ICLR); San Diego, CA, USA.

Yuanpu X, Fuyong X, Xiangfei K, Hai S, Lin Y. 2015. Beyond classification: structured regression for robust cell detection using convolutional neural network. In: Medical Image Computing and Computer-Assisted Intervention (MICCAI). Munich: Springer; p. 358–365.