Detection and Instance Segmentation of Neuroblastoma Cells using YOLO, UNet and Conditional Random Fields

Kristofer delas Peñas and Dominic Waithe

***Abstract*—The abstract goes here.**

***Index Terms*—IEEE, IEEEtran, journal, LaTeX, paper, template.**

I. Introduction

THIS demo file is intended to serve as a "starter file" for IEEE journal papers produced under LaTeX using IEEEtran.cls version 1.8b and later. I wish you the best of success.

II. Literature Review

A. Semantic Segmentation

B. Instance Segmentation

C. YOLO

The YOLO (You Only Look Once) network [1] is a state-of-the-art object detection system that gained popularity within the past few years. It employs a series of convolutions to detect and localize objects in real-time.

The implementation of YOLO starts with an initial 416 $\times$ 416 image input. The system performs 19 convolutions with an overall downsampling factor of 32, resulting to an output feature map of 13 $\times$ 13 dimension which is used to predict the bounding boxes of detected objects with the help of anchor boxes.

With version 2 of YOLO, [1] demonstrated a high mean average precision (mAP) on the VOC dataset, higher than existing architectures like Faster-RCNN.

A more recent YOLO iteration, version 3 [2], uses a deeper network and performs object detection on 3 stages of different resolutions. With these changes, an even higher mAP was achieved, however, at the expense of heavier computational resource requirement and longer detection time.

D. UNet

UNet[3] is a fully convolutional neural network architecture that aims to semantically label individual pixels in the image. The network consists of two paths - a contracting path and an expansive path. The contracting path is the typical convolutional network with a series of convolutional layers with rectified linear unit (ReLU) activations and pooling layers to downsample.

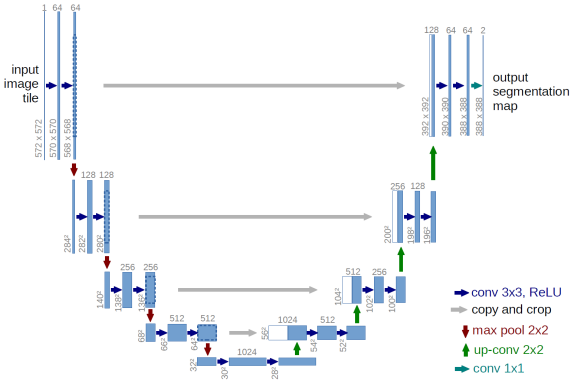


Fig. 1: **U-net architecture** [3] (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

The expansive path is an upsampling route, taking the output of the contracting path. This path performs a series of *up-convolutions*, a combination of upsampling and a 2 $\times$ 2 convolution.

The key feature in UNet is the shortcut connections between the two paths for each resolution scale as shown in Figure 1. At each scale, concatenating the output from the contracting path with the upsampled output from the previous scale in the expansive path, ensures that the finer details lost through downsampling can be recovered and be used to fine tune the segmentation.

UNet was first applied on microscopy images but since then have been applied to different imaging modalities like MRIs and CT scans.

E. Conditional Random Fields

Many learning problems can be described as a graphical model. In artificial intelligence, one common way to model these problems is a probabilistic approach wherein probability distributions Ψ are assigned over the random variables Y.

K. delas Peñas is with the Doctoral Training Centre, University of Oxford, United Kingdom, and the Department of Computer Science, University of the Philippines Diliman, Philippines e-mail: kristofer.delaspenas@wolfson.ox.ac.uk.

D. Waithe is with the MRC Weatherall Institute of Molecular Medicine, University of Oxford, United Kingdom.

Formally, the problem can be modelled as the product of all Ψ_i distributions

$$p(Y) = \frac{1}{Z}\prod_{1}^{A}\Psi_a(y_a) \tag{1}$$

for factors $F = \{\Psi_a\}$ that have $\Psi_a \geq 0$.

Markov networks and conditional random fields (CRF) are formulated similarly in this manner. The main difference is that the CRF learns the conditional probability $p(y|x)$ while Markov networks ultimately obtain the joint probability $p(y, x)$. With the joint probability $p(y, x)$, models like the Markov networks can describe the hypothesis space through the generation of all possible features x for all labels y. However, joint probability $p(y, x)$ involves prior knowledge on or estimate for $p(x)$, and the dependence (or independence) of the random variables. For general classification tasks, however, modelling of $p(x)$ is not needed as the concern is only on assigning labels to features, which is exactly what the conditional probability $p(y|x)$ gives.

Exact inference on conditional random fields is computationally expensive and usually impossible. If exact inference is possible, performing naively the sum of products of potentials (**message passing**) for all variables, can take a long time, especially for dense graphs. Current implementations of CRFs perform inference by approximation, with the speedup by employing dynamic programming. One inference algorithm is the mean field inference described in Algorithm 1.

Each iteration of the mean field inference described in Algorithm 1 performs a message passing step, compatibility transform, and local update and normalization. Each step, except the message passing, runs in linear time. Message passing is the computational bottleneck. For a naive solution, it requires summing up over all variables and runs in quadratic time. For this very reason, the conditional random fields gained the reputation of being nooriously slow and impractical for a lot of machine learning tasks, especially those involving dense graph representations like image segmentation.

Addressing the slow inference in CRFs, [4] showed that message passing can be approximated and expressed as a convolution with a Gaussian kernel as follows:

$$\widetilde{Q}_i^{(m)}(l) \leftarrow \sum_{j\neq i} k^{(m)}(f_i, f_j)Q_j(l) = [G_(m)\otimes Q(l)](f_i) - Q_i(l) \tag{2}$$

This approximation can be extended to higher dimensions, and with the permutohedral lattice data structure, efficient message passing can be done in $O(Nd)$ time where N is the number of variables and d the number of dimensions. Furthermore, [5] and [6] demonstrated how the mean field inference in CRFs can be written as recurrent neural network with learnable weights. This formulation allows the CRFs to be seamlessly integrated as part of a convolutional neural network model for tasks such as image segmentation.

With CRFs implemented as RNNs, several research has been done applying CNN-CRFs for general image segmentation task. [4] and [7] proposed systems using CRFs integrated in CNNs for semantic image segmentation. Both systems train the CRF by minimizing the Gibbs energy and in doing so, finding the Maximum A Posteriori labelling of the image pixels. The CRF for semantic image segmentation is formulated with the following energy function for assignment of the pixels to semantic classes,

$$E(X = x) = \sum_i U(x_i) + \sum_{i<j} P(x_i, x_j) \tag{3}$$

where U is the unary potential and P the pairwise potential. In [4], responses from the TextonBoost filter bank, color, histogram of oriented gradients (HOG) and pixel location features were used for the unary potential. On the other hand, [7] used the segmentation prediction from ResNet101 for the unary. Both systems used gaussian kernels for the pairwise potentials, given as:

$$k(f_i^I, f_j^I) = w^{(1)}G_{appearance} + w^{(2)}G_{smoothness} \tag{4}$$

$$G_{appearance} = exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right) \tag{5}$$

$$G_{smoothness} = exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right) \tag{6}$$

The two terms in the pairwise potential encourages neighboring pairs of pixels that look similar to be assigned the same semantic label.

Extending from this, [8] and [9] introduced some modification to this energy function to be able to perform pixelwise instance segmentation. The systems described for this task works on an initial instance segmentation from a detection algorithm, where each detection has a corresponding prediction score. This instance segmentation is further refined by adding information from semantic segmentation and the gaussian pairwise potentials. To do this, They have broken down the unary potential to accommodate two distinct terms Ψ_{box} and Ψ_{global} as follows:

$$U(x_i) = -ln[w_1\Psi_{box}(x_i) + w_2\Psi_{global}(x_i)] \tag{7}$$

The Ψ_{box} term encourages the pixel to be assigned to the instance corresponding to the detection. This is proportional to the probability of the semantic class assigned to the pixel and the detection score. The Ψ_{global} term accounts for pixels that might have been misclassified to another instance label but belongs to the same semantic label. This addresses the problem in the initial instance segmentation that does not fully cover the entire extent of the individual instances.

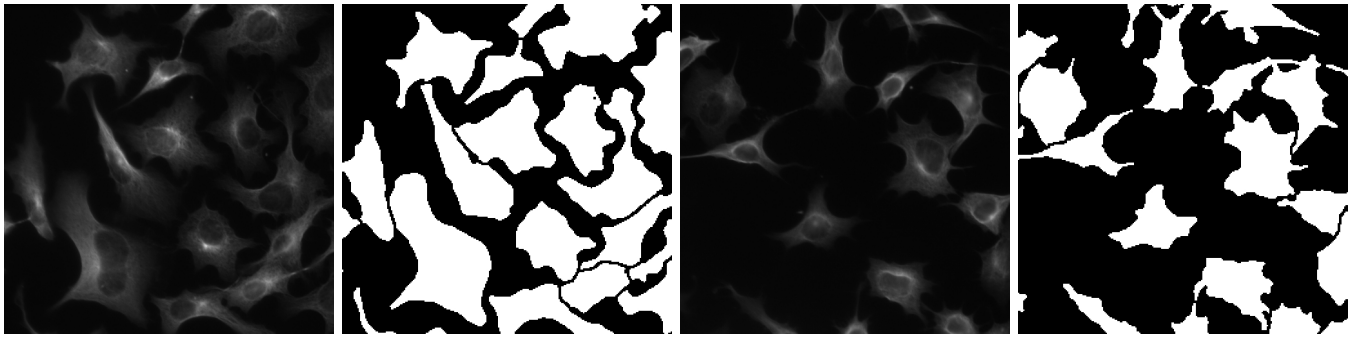
III. Experiment Setup

A. Dataset

The dataset is composed of wide field fluorescent images of cultured neuroblastoma cells labelled with phalloidn FITC and DAPI nuclear stain.

Algorithm 1 Mean Field Inference

Initialize Q
while not converged **do**
  $\widetilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(f_i, f_j) Q_j(l)$ for all m ▷ Message Passing
  $\hat{Q}_i(x_i) \leftarrow \sum_{l \in L} \mu^{(m)}(x_i, l) \sum_w^{(m)} \widetilde{Q}_i^{(m)}(l)$ ▷ Compatibility Transform
  $Q_i(x_i) \leftarrow exp{-}\psi_u(x_i) - \hat{Q}_i(x_i)$
  normalize $Q_i(x_i)$ ▷ Softmax
end while



(a) (b) (c) (d)

Fig. 2: **The Dataset**. Shown here are sample images of neuroblastoma cells (a,c) and their corresponding segmentation masks (b,d).

B. Model Architecture

1) Detection: For detection, we employ the YOLO version 2 network because as reviewed previously, it is effective in detecting objects in real-time. Specifically, we use its 'tiny' implementation. The tiny implementation of YOLO version 2 differs from its full counterpart in the number of filters in each convolutional layer. In each layer, the tiny implementation contains only half of the number of filters in the full implementation. With this reduction in the number of filters comes an improvement in the detection time of Tiny YOLO version 2. In a microscopy setting, especially dealing with fluorescence-labelled samples with the detection of cells done in an integrated manner like in [10], real-time processing is desirable. Moreover, the small memory footprint of Tiny YOLO version 2 makes it an even more suitable detection algorithm to consider. We start the training with the convolutional weights pre-trained on Imagenet and fine-tune the network for neuroblastoma detection.

2) Semantic Segmentation: We compare two models for semantic segmentation. We first train a UNet model with 5 resolution levels. Next, we build a hybrid model combining the detection from the Tiny YOLO network and conditional random fields. We use the initial bounding boxes from the neuroblastoma detection and try to refine the segmentation to find the cell boundaries using the pairwise Gaussian kernels described in [4] and [7].

3) Instance Segmentation:

C. Training Parameters

1) Detection: For training the tiny implementation of YOLO version 2 network for neuroblastoma detection, we employ 64 batch size, 8 subdivisions, 416 × 416 × 3 input image dimensions, 0.9 momentum, 0.0005 decay, 0.001 learning rate, and 120000 epochs. Additional image augmentation by vertical flips was done, as described in [10], to introduce robustness in detection.

2) Semantic Segmentation: For training the UNet, we set the learning rate to 0.001, batch size to 16, momentum to 0.9, decay to 0.0005, and epochs to 8000. The binary cross entropy with logits loss was used as the objective function.

For the YOLO-CRF hybrid model, we used a 0.00001 learning rate, 0.9 momentum, 4 batch size, 1000 epochs, and binary cross entropy with logits loss as objective function. The Gaussian CRF is configured with a filter size of 3, blur of 8, θ_α of 1/80, θ_β of 1/13, and θ_γ of 1/3, and a trainable bias term.

3) Instance Segmentation: For training the YOLO-UNet-CRF hybrid model, the CRF is configured with a filter size of 11, blur of 4, θ_α of 1/80, θ_β of 1/13, and θ_γ of 1/3, and a trainable bias term. The training, set for 10000 epochs, was done with a low learning rate of 1e-12 to account for the small batch size of 1.

D. Evaluation Metrics

We assessed the trained models for detection and segmentation with mean average precision. For the instance segmentation, we used the panoptic quality metric proposed in

$$PQ = \frac{\sum_{(p,g) \in TP} IoU(p,g)}{|TP|} \times \frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \tag{8}$$

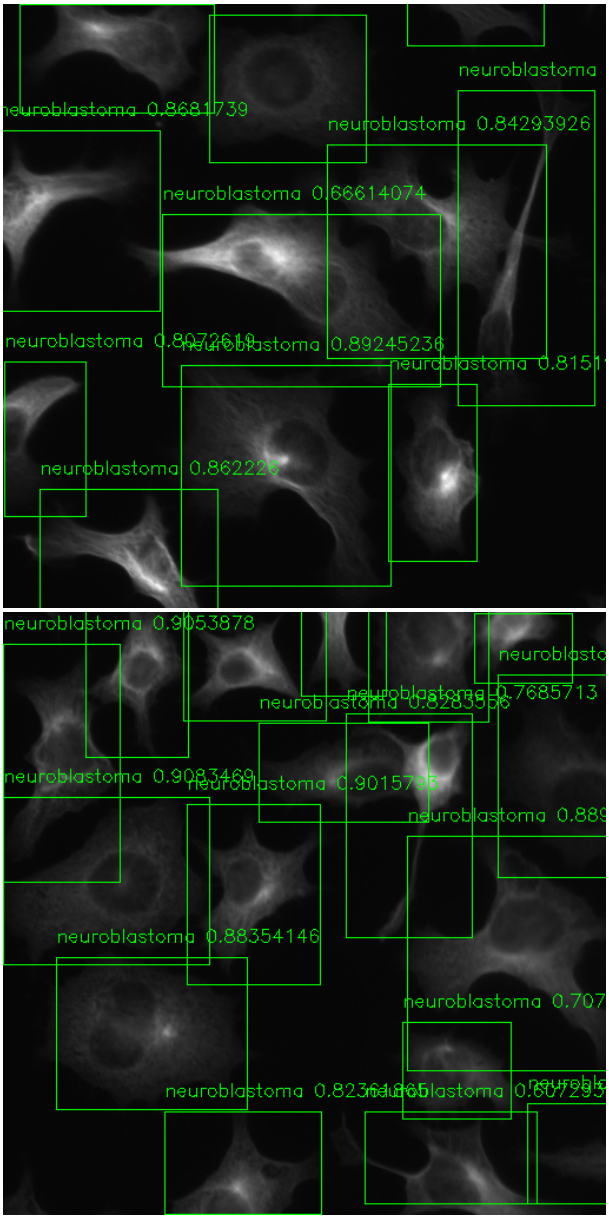


Fig. 3: **Neuroblastoma Detection**. Shown are some of the detected neuroblastoma cells using the Tiny YOLO network with the corresponding detection scores

IV. Results and Discussion

A. Neuroblastoma Detection

Detecting the neuroblastoma cells using the Tiny YOLO network proved to be an easy task. Figure 3 shows some of the boxed neuroblastoma cells in the test images. Even with the highly irregular and dissimilar shape of the neuroblastoma cells, the trained detection network obtained an mAP of (value here).

B. Neuroblastoma Semantic Segmentation

While the Tiny YOLO network trained to detect neuroblastoma proved to be effective and efficient in its task, the segmentation it gives is a coarse one - a bounding box. For problems requiring pixel-level labelling of the cells, the output of YOLO is not enough. Here, we tried to combine the good cell localization of YOLO and the smoothing effect of conditional random fields to resolve pixelwise the cells and the background.

C. Neuroblastoma Instance Segmentation

V. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[2] ——, "Yolov3: An incremental improvement," *arXiv*, 2018.

[3] O. Ronneberger, P.Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[4] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 109–117. [Online]. Available: http://papers.nips.cc/paper/4296-efficient-inference-in-fully-connected-crfs-with-gaussian-edge-potentials.pdf

[5] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr, "Conditional random fields as recurrent neural networks," in *International Conference on Computer Vision (ICCV)*, 2015.

[6] A. Arnab, S. Jayasumana, S. Zheng, and P. H. S. Torr, "Higher order conditional random fields in deep neural networks," in *European Conference on Computer Vision (ECCV)*, 2016.

[7] M. T. T. Teichmann and R. Cipolla, "Convolutional crfs for semantic segmentation," *CoRR*, vol. abs/1805.04777, 2018.

[8] A. Arnab and P. H. S. Torr, "Pixelwise instance segmentation with a dynamically instantiated network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 879–888, 2017.

[9] Q. Li, A. Arnab, and P. H. Torr, "Weakly- and semi-supervised panoptic segmentation," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[10] D. Waithe, J. M. Brown, K. Reglinski, I. Diez-Sevilla, D. Roberts, and C. Eggeling, "Object detection networks and augmented reality for cellular detection in fluorescence microscopy acquisition and analysis." *bioRxiv*, 2019. [Online]. Available: https://www.biorxiv.org/content/early/2019/02/08/544833