A Technical Report:
Prior Information Guided Regularized Deep Learning for Cell Nucleus Detection

Mohammad Tofighi, Tiantong Guo, Vishal Monga
Pennsylvania State University

I. COST FUNCTION: SP-CNN

Given luminance information of a raw input image \( x \), using Canny edge detection generating its raw edge image \( \hat{x} \). We construct \( y \in [0, 1] \) by processing the binary image of ground-truth nuclei center locations, which has 1 at the nucleus center and 0 elsewhere. This is accomplished by convolving the said ground-truth binary image with a zero mean Gaussian (\( \sigma = 2 \)) filter of size \( 7 \times 7 \). The network is training to generate \( \hat{y} \) that expecting to be as similar as \( y \) while incorporating the shape prior information during training. We denote the shape prior set as:

\[
S = \{S_i | i = 1, 2, \ldots, n\}.
\]

(1)

Note that the boundaries are labeled as 1 and the non-boundary regions are labeled as 0. The network is modeled by parameters set \( \Theta = \{W, b\} \), where the weights \( W \) and bias \( b \) are denoting the weights and bias of all layers collectively. Let the SP-CNN cost function be \( L = L^{\text{Loss}} + L^{\text{SP}} \), then we optimize the network parameters in the following manner:

\[
\Theta = \arg \min_{\Theta} L^{\text{Loss}} + L^{\text{SP}} = \arg \min_{\Theta} \|f(x; \Theta) - y\|_2^2 - \lambda \sum_{i=1}^{n} \| (g_p(\hat{y}) \circ \hat{x}) \ast S_i \|_2^2,
\]

(2)

where \( f(x) \) represents the CNN that generates the detection maps \( \hat{y} \). Note that \( \hat{y} := f(x; \Theta) \). The \( g_p(\cdot) \) denotes the max pooling operation on \( \hat{y} \) with window size \( p \), \( \circ \) denotes element-wise multiplication and \( \ast \) denotes 2-D convolution. There are two parts in the cost function: the detection fidelity cost term and shape priors cost term.

II. DETECTION FIDELITY COST TERM

The detection fidelity cost term reflects how good the detection map \( \hat{y} \) is fitting to the ground-truth \( y \) numerically. During the training, \( \ell_2 \) norm is used to capture the deviations:

\[
L^{\text{Loss}} = \|f(x; \Theta) - y\|_2^2,
\]

(3)

where \( f(x; \Theta) \) is the output of the CNN also denoted as \( \hat{y} \). To update \( \Theta \) according to this term, standard back-propagation algorithm is used. For detection fidelity cost term the back-propagation is performed by:

- At iteration step \( t \), weights are updated by the following equation:

\[
\Theta^{t+1} = \Theta^t - \eta \frac{\partial L^{\text{Loss}}}{\partial \Theta^t},
\]

(4)

where, \( \eta \) represents the learning rate for the stochastic gradient descent method and \( \Theta^t \) represents the values of weights at previous iteration.

- Since, our network parameter \( \Theta \) consists of weights from \( D \) convolutional layers, following gradients are to be computed:

\[
\frac{\partial L^{\text{Loss}}}{\partial W^d}, \frac{\partial L^{\text{Loss}}}{\partial b^d} \quad d = 1, \ldots, D
\]

- For simplicity, we focus on filters and assume that output image \( \hat{y} \) is of dimension \( N \times N \).
• The equation for computing the gradient of the weights at layer \( l \) is given by:

\[
\frac{\partial L}{\partial W_l} = -(y - \hat{y}) \cdot \frac{\partial \hat{y}}{\partial W_l}.
\]  

(5)

• This is the standard back-propagation for CNN, \( \frac{\partial \hat{y}}{\partial W_l} \) is obtained by [1].

### III. Shape Priors Cost Term

The shape prior cost term generates a measurement of how good the detection \( \hat{y} \) is fitting inside the cell shapes. If the detection \( \hat{y} \) has more labels predicted inside the cell boundary, Eq. (6) will produce a higher value, since more shape priors will find the similar shape boundaries in its surroundings of the correct detections. The cost term for shape priors is:

\[
L_{SP} = -\lambda \sum_{i=1}^{n} \| (g_p(\hat{y}) \odot \hat{x}) \ast S_i \|_2.
\]

(6)

Each part of the shape priors cost term is explained as:

• \( \hat{y} := f(x) \) - the output detection maps of the CNN,

• \( g_p(\cdot) \) - the max pooling with stride of 1, ‘SAME’ padding, and window size of \( p \),

• \( \hat{x} \) - the raw edge information generated by Canny edge filters,

• \( S_i \) - the \( i^{th} \) shape prior in the shape priors set, \( i = 1, ..., n \).

To carry the shape priors cost term into the \( \Theta \), we need to update Eq. (4) accordingly. Examining closely of the Eq. (6), we can re-write it as:

\[
L_{SP} = -\gamma \sum_{i=1}^{n} \| (g_p(f(x: \Theta)) \odot \hat{x}) \ast S_i \|_2^2.
\]

(7)

Now we can update Eq. (4) accordingly to include the gradient of shape prior term:

• Updated Eq. (4) will be:

\[
\Theta^{t+1} = \Theta^t - \eta \frac{\partial L_{Loss}}{\partial \Theta^t} - \eta \frac{\partial L_{SP}}{\partial \Theta^t}.
\]

(8)

• Since our network parameter \( \Theta \) consists of weights from \( D \) convolutional layers, following gradients are to be computed:

\[
\frac{\partial L_{SP}}{\partial W_{l}}, \frac{\partial L_{SP}}{\partial b_{l}} \quad l = 1, ..., D
\]

(9)

• The equations for computing the gradients of weights at layer \( l \) are given by:

\[
\frac{\partial L_{SP}}{\partial W_{l}^{m',n'}} = \sum_{i=0}^{N-k_1-1} \sum_{j=0}^{N-k_2-1} \frac{\partial L_{SP}}{\partial x_{l}^{i,j}} \frac{\partial x_{l}^{i,j}}{\partial W_{l}^{m',n'}}
\]

\[
= \sum_{i=0}^{N-k_1-1} \sum_{j=0}^{N-k_2-1} \delta_{l}^{i,j} \frac{\partial x_{l}^{i,j}}{\partial W_{l}^{m',n'}}
\]

(10)

(11)

where \( W \) is of dimension \( k_1 \times k_2 \) has \( m \) by \( n \) as the iterators, \( x_{l}^{i,j} \) is the convolved input vector at layer \( l \) plus the bias represented:

\[
x_{l}^{i,j} = \sum_{m} \sum_{n} W_{l}^{m+n} \cdot o_{l-1}^{i+m,j+n} + b_{l},
\]

(12)

and \( o_{l}^{i,j} \) is the output vector at layer \( l \) given by:

\[
o_{l}^{i,j} = \max(x_{l}^{i,j}, 0).
\]

(13)

• For \( l = D \) and \( x_D = \hat{y} \):

\[
\delta_{D}^{i,j} = \frac{\partial L_{SP}}{\partial x_{D}^{i,j}} = \sum_{i=1}^{n} g_p^{-1}(x_{D}^{i,j} \odot \hat{x}) \ast \text{rot}_{180=\{S^{m,n}\}},
\]

(14)
where \( g_p^{-1}(\cdot) \) assigns the weights to where it comes from - the winning unit because other units in the previous layers' pooling blocks did not contribute to it hence all the other assigned values of zero.

For the total loss \( L \), the equations for computing the gradients of the biases for both terms, \( L^{\text{Loss}} \) and \( L^{\text{SP}} \), at layer \( l \) with respect to a scalar entry \( b_{al} \) are given by:

\[
\frac{\partial L}{\partial b_{al}} = -<\hat{y} - y, \frac{\partial \hat{y}}{\partial b_{al}^p}>_F - \lambda \sum_{i=1}^{N} <(g_p(\hat{y}) \odot \hat{x}), g^{-1}(\frac{\partial \hat{y}}{\partial b_{al}^p}) \odot \hat{x} \ast S_i>_F .
\] (15)

The math notations are refereed to [2] and [3].

A visual illustration of nuclei detection results is presented in Fig. 1 for an example test image from UW Dataset [4]. Figure 1.a) is an example image, Fig. 1.b) is the TSP-CNN output, and Fig. 1.c) represents the detected nuclei locations and the ground-truth golden standard regions.

![Example image](image1)

**Fig. 1**: Example detection result for SP-CNN on UW Dataset.

**REFERENCES**

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *arXiv preprint arXiv:1603.07285*, 2016.

[3] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[4] K. Sirinukunwattana, S. E. A. Raza, Y. W. Tsang, D. R. J. Snead, I. A. Cree, and N. M. Rajpoot, “Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1196–1206, May 2016.