Manuscript submitted to
AIMS' Journals
Volume **X**, Number **0X**, XX **200X**

doi:10.3934/xx.xx.xx.xx

pp. **X–XX**

NONLOCAL REGULARIZED CNN FOR IMAGE SEGMENTATION

Fan JIA

Department of Mathematics, Hong Kong Baptist University
Hong Kong, China

Xue-cheng TAI

Department of Mathematics, Hong Kong Baptist University
Hong Kong, China

Jun LIU*

Laboratory of Mathematics and Complex Systems (Ministry of Education of China)
School of Mathematical Sciences, Beijing Normal University
Beijing, China

(Communicated by the associate editor name)

Abstract. Non-local dependency is a very important prior for many image segmentation tasks. Generally, convolutional operations are building blocks that process one local neighborhood at a time which means the convolutional neural networks(CNNs) usually do not explicitly make use of the non-local prior on image segmentation tasks. Though the pooling and dilated convolution techniques can enlarge the receptive field to use some nonlocal information during the feature extracting step, there is no nonlocal priori for feature classification step in the current CNNs' architectures. In this paper, we present a nonlocal total variation (TV) regularized softmax activation function method for semantic image segmentation tasks. The proposed method can be integrated into the architecture of CNNs. To handle the difficulty of back-propagation for CNNs due to the non-smoothness of nonlocal TV, we develop a primal-dual hybrid gradient method to realize the back-propagation of nonlocal TV in CNNs. Experimental evaluations of the non-local TV regularized softmax layer on a series of image segmentation datasets showcase its good performance. Many CNNs can benefit from our proposed method on image segmentation tasks.

1. **Introduction.** Image segmentation has long been a hot topic and attracted hundreds of thousands of researchers from lots of fields all over the world. Generally speaking, given an image, image segmentation aims to classify all pixels to several classes. In the past few decades, kinds of methods have been proposed for image segmentation. Depending on whether there are labels available or not, all the methods could be mainly classified into two types, unsupervised methods and supervised methods. Given no label prior, unsupervised methods such as thresholding method [24] , edge based method[7], region based method[1], partial differential equation(PDE) and variational based methods[12, 21, 20], graph partitioning method [29] and their variations first appear in the last century. These methods

2010 *Mathematics Subject Classification.* 68U10, 68T10, 65K10.
Key words and phrases. CNN, Nonlocal TV, Regularization, Duality, Image Segmentation.
* Corresponding author: Jun LIU.

usually make use of constraints according to some prior information such as image intensity, shape prior. Besides unsupervised methods, when there are annotated samples available, supervised methods come into being and learn valid information from training dataset. Discriminative features and context information will be extracted followed by a dense pixel-wise classification.

Total variation(TV) exhibits prominent performance in image restoration problems, which is first introduced in computer vision by Rudin, Osher and Fatemi[26]. It is one of the most popular regularization methods in image processing field due to its good performance in handling minimization problems. In the recent decades, TV has been explored by thousands of researchers and extended to a series of forms for dealing with sorts of other image processing tasks, such as anisotropic TV[6], weighted TV[31], fourth-order PDE model(two-step method)[18], higher-order TV[5], and non-local TV[8]. An effective framework has also been proposed in recent years. After introducing a novel region force term into Potts model,it achieves good performance in multi-phase image segmentation and semi-supervised data clustering[33] tasks. Their method can be easily applied to high dimensional data clustering tasks via graph total variation.

Convolutional Neural Networks (CNNs)[14, 15] have achieved distinguished performance in a series of tasks in the last decade. Especially in computer vision area, CNNs showcase their prominent abilities in learning discriminative features from various large scale datasets. Leading other methods by a large margin, CNNs achieve the first place in many kinds of tasks such as image classification, object detection and image segmentation. Semantic image segmentation is a dense classification task which aims to classify each pixel to a certain class. It not only segments a given image into several regions, but also tells you which label each pixel belongs to[4, 28].

Fully Convolutional Networks (FCNs)[17] was the first successful attempt for semantic image segmentation task via an end-to-end CNN framework. Noh et al.[22] proposed an extension of FCNs. They used a VGG[30] 16-layer network as the convolution network, followed by a series of up-pooling and deconvolutional layers. Utilizing the spatial dependency between neighbor pixels, Conditional Random Fields (CRFs) were employed as a post-processing after CNNs to refine segmentation results[16]. CNNs were also introduced to medical image processing fields due to its powerful ability. Inspired by FCN, U-net[25] using a symmetric structure while adding skip-connections like FCN to concatenate feature maps from different levels together. With abundant features from different levels, U-Net achieved very prominent performance and thus has since been applied to kinds of medical imaging tasks such as image translation. In recent years, variations of Unet come forth. Attention U-Net[23] employs attention gates to help CNN focus on extracting discriminative features from foreground. R2U-Net[2] introduces recurrent residual blocks to U-Net and achieves better results on several retina blood vessel segmentation datasets. Different variations of U-Net continuously improving the segmentation performance on kinds of medical image datasets.

However, convolution operators just can learn features from local context. Long range dependency is also important information in semantic image segmentation. Dilated convolution operators[35] could capture long range information. It has a larger receptive field with same computation and memory costs while also preserving resolution. But the dilated convolution would lose some position information in image segmentation. What is more, the receptive field of dilated convolution

is not continuous and dilated convolution does not work well when facing small objects. Capturing multi-scale feature information, well designed Hybrid Dilated Convolution(HDC) [32] could eliminate the effect of some disadvantages and improve segmentation performance. Although HDC can work well when there is big enough training corpus, when the training dataset is quite small, HDC can hardly showcase its performance. That is why dilated convolution operators seldom appear in CNNs for medical image segmentation.

Given a few training samples, we explore the potential of non-local operators and provide a novel way to capture long range information in CNNs. In summary, the contributions of this paper are as follows:

- We introduce graph total variation to softmax activation function, one can easily extend this model to other activation functions in CNNs. Some earlier works have tried to introduce local total variation to softmax activation function[10], but it is well known that non-local dependency is an important prior for the image segmentation problem.
- We introduce a primal-dual hybrid gradient method for our proposed regularized softmax activation function that enables end-to-end training.
- Experimental results show the good performance of the nonlocal total variation. Local total variation regularized softmax activation function could produce smoother objects, but it may lose some details such as corners. It is numerically verified that nonlocal total variation could eliminate isolated regions and preserve object details at the same time.

The paper is organized as follows. In Section 2, we give brief descriptions to related work. Our proposed method is illustrated in Section 3. In this section, we apply our proposed method to softmax layer and give the general formulas for forward propagation, backward propagation. Some implementation details are also illustrated here. The experimental results are described in Section 4, and the conclusions follow in Section 5

2. Related Work.

2.1. Multi-phase Image Segmentation.

Let $I(x)$ be an image which is defined on a domain $\Omega \in \mathbb{R}^2$, the multi-phase image segmentation task is to classify Ω into K partitions, where K is the number of classes. Let $\{\Omega_k\}_{k=1}^K$ be the partitions, we have $\Omega = \cup_{k=1}^K \Omega_k$ and $\Omega_{\hat{k}} \cap \Omega_k = \emptyset$ when $\hat{k} \neq k$. Potts model is a general variational based image segmentation model for multi-phase image segmentation. It consists of two terms, the data fidelity term and regularization term. Generally, it can be defined by the following minimization problem:

$$\min_{\Omega_k} \sum_{k=1}^{K} \int_{\Omega_k} f_k(x)dx + R(\{\Omega_k\}_{k=1}^K). \tag{1}$$

The second term in Eq. (1) is a jump penalty which is usually defined as follows:

$$R(\{\Omega_k\}_{k=1}^K) = \sum_{k=1}^{K} |\partial\Omega_k|_\alpha = \sum_{k=1}^{K} \int_{\partial\Omega_k} \alpha(x)ds, \tag{2}$$

where $\alpha(x)$ is an edge detector defined as $\alpha(x) = \frac{\beta}{1+\gamma|\nabla I_\sigma|^2}$. γ and β are manually set parameters controlling the property of edge detector. I_σ is the result of convoluting the image $I(x)$ with a Gaussian kernel g_σ. The jump penalty is a scaled sum of boundary total length when $\alpha(x)$ is a constant $\lambda \in \mathbb{R}$.

If we define an indicator function $\phi_k(x)(k = 1, 2, \cdots, K.)$ on the k-th sub-domain,

$$\phi_k(x) = \begin{cases} 1 & x \in \Omega_k \\ 0 & otherwise. \end{cases} \tag{3}$$

then we have

$$\sum_{k=1}^{K} \int_{\partial\Omega_k} \alpha(x) ds \quad = \sum_{k=1}^{K} \int_{\Omega} \alpha(x) |\nabla \phi_k(x)| dx, \tag{4}$$

where $\boldsymbol{\phi} = (\phi_1, \cdots, \phi_K)$. The segmentation condition becomes a relaxed one

$$\mathbb{S} = \{\boldsymbol{\phi}(x) : \sum_{k=1}^{K} \phi_k(x) = 1, 0 \leq \phi_k(x) \leq 1\}. \tag{5}$$

Corresponding to the binary segmentation constraint on ϕ_k in (1), one can get the following convex programming problem which is a dual of a min-cut problem:

$$\min_{\boldsymbol{\phi} \in \mathbb{S}} \sum_{k=1}^{K} \int_{\Omega} f_k(x) \phi_k(x) dx + \sum_{k=1}^{K} \int_{\Omega} \alpha(x) |\nabla \phi_k(x)| dx. \tag{6}$$

2.2. **Graph Model for Data Clustering.** Graph model is a useful tool if we want to utilize pairwise relations between pixels. A undirected weighted graph $G = (\mathbb{V}, \mathbb{E}, w)$ is constructed by vertex set $\mathbb{V}$, edge set $\mathbb{E}$ and a weight function $w : \mathbb{E} \to \mathbb{R}_+$ which is defined on the edges. In the image segmentation task, each image could be seen as a graph, each pixel in the image is a vertex. For $x_i, x_j \in \mathbb{V}$, $w_{ij} = w(x_i, x_j)$ measures the similarity between two vertexes.

Since an image often has at least dozens of thousands of pixels, the computation and memory cost will be extremely huge if we use complete graph. Therefore, we assume that each pixel is connected to only a portion of other pixels. Then we get a sparse affinity matrix W. There are several methods to measure the similarity between two pixels. Given a distance metric $dist(\cdot)$ which measures the distance of the feature vector of two pixels x_i and x_j, the radial basis function (RBF) [27] is defined as:

$$w(x_i, x_j) = \exp\left(\frac{-dist(x_i, x_j)^2}{2\epsilon}\right). \tag{7}$$

If we replace the constant 2ϵ with the product of local variances $\sigma(x_i)\sigma(x_j)$, here comes the Zelnik-Manor and Perona function (ZMP) [36]:

$$w(x_i, x_j) = \exp\left(\frac{-dist(x_i, x_j)^2}{\sigma(x_i)\sigma(x_j)}\right). \tag{8}$$

The cosine similar function is also widely used to measure the similarity between two non-zero vectors. It is defined as:

$$w(x_i, x_j) = cos(x_i, x_j) = \frac{<x_i, x_j>}{|x_i||x_j|}, \tag{9}$$

In the fully connected pairwise CRF model [13], the weight function is often defined by pairwise potentials as

$$w_p(x_i, x_j) = \mu(x_i, x_j) \sum_{m=1}^{M} \lambda_m \exp(-\tfrac{1}{2}(\boldsymbol{f}_i, \boldsymbol{f}_j)^T \Lambda_m (\boldsymbol{f}_i, \boldsymbol{f}_j)), \tag{10}$$

where $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$ are feature vectors for pixels x_i, x_j, λ_m is a coefficient to control the impact of each kernel, and μ represents the label compatibility function. Λ_m is a symmetric, positive-definite precision matrix.

Let $\mu(x_i, x_j) = 1$ if $x_i \neq x_j$ and $\mu(x_i, x_j) = 0$ otherwise, we use RGB color vectors I_i, I_j and spatial positions p_i, p_j as feature vectors. Given two different pixels, the weight function in Eq. (10) can be rewritten as:

$$w(x_i, x_j) = \lambda_1 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\alpha} - \frac{||I_i - I_j||^2}{2\sigma_\beta}\right) + \lambda_2 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\gamma}\right), \tag{11}$$

where $\sigma_\alpha, \sigma_\beta \sigma_\gamma$ are parameters controlling the scale of Gaussian kernels. The first term depends on both pixel colors and positions. Pixels with small differences in positions and colors are likely to have the same label. The second term only takes into account the spatial correlation, isolated points and regions will be removed.

2.3. **Graph Operators.** After introducing the weight functions in graph, some graph operators will be given in the following. One important operator is gradient operator. Given $u \in L^2(\mathbb{V})$ defined on the vertex set, the gradient operator

$$\nabla_w : L^2(\mathbb{V}) \to L^2(\mathbb{V} \times \mathbb{V}) \tag{12}$$

is defined by

$$(\nabla_w u)(x_i, x_j) = w(x_i, x_j)(u(x_j) - u(x_i)). \tag{13}$$

So $\nabla_w u$ is a function in $L^2(\mathbb{V} \times \mathbb{V})$. Since we assume each pixel is only connected to a small potion of other pixels, we get a sparse graph G and each x_i has at most d neighbors. Then $(\nabla_w u)(x_i, x_j)$ is a sparse vector

$$(\nabla_w u)(x_i, x_j) = (w(x_i, x_j)(u(x_j) - u(x_i)))_{x_j \in \mathcal{N}(x_i)} \tag{14}$$

with at most d non-zeros.

Correspondingly, the divergence operator

$$div_w : L^2(\mathbb{V} \times \mathbb{V}) \to L^2(\mathbb{V}) \tag{15}$$

is given by

$$(div_w v)(x_i) := \textstyle\sum_{x_j \in \mathcal{N}} w(x_i, x_j)(v(x_i, x_j) - v(x_j, x_i)), \tag{16}$$

where $v \in L^2(\mathbb{V} \times \mathbb{V})$,.

2.4. **Discrete Potts Model.** Given a graph $G = (\mathbb{V}, \mathbb{E}, w)$, we want to classify the vertexes $\mathbb{V}$ into K partitions, denoted by $\mathbb{V}_1, \cdots, \mathbb{V}_K$. Then the corresponding indicator function $\phi_k(x_i)$ for the k-th class is defined as:

$$\phi_k(x_i) = \begin{cases} 1 \ if \ x_i \in \mathbb{V}_k \\ 0 \ otherwise. \end{cases} \tag{17}$$

The discrete counterpart of the Potts model defined in Eq. (6) is given by :

$$\min_{\boldsymbol{\phi} \in \mathbb{S}} \sum_{k=1}^{K} \sum_{x_i \in \mathbb{V}} f_k(x_i)\phi_k(x_i) + \sum_{k=1}^{K} NLTV_\alpha(\phi_k), \tag{18}$$

where $f_k(\cdot)$ is a region force function and $NLTV_\alpha(\phi_k)$ is the α weighted non-local total variation. As the dual norm of ℓ_1-norm is ℓ_∞-norm, $NLTV_\alpha(\phi_k)$ has the following form:

$$\begin{aligned} NLTV_\alpha(\phi_k) &= \max_{||q_k||_\infty \leq \alpha} < \nabla_w \phi_k, q_k > \\ &= \max_{||-q_k||_\infty \leq \alpha} - < \phi_k, div_w q_k > \\ &= \max_{||q_k||_\infty \leq \alpha} \textstyle\sum_{x_i \in \mathbb{V}} \phi_k(x_i)(div_w q_k)(x_i), \end{aligned} \tag{19}$$

where $< \cdot, \cdot >$ is the standard inner product of two vectors, q_k is the dual variable of ϕ_k, ∇_w and div_w are non-local gradient and divergence operators defined in Eq. (14) and Eq. (16), respectively.

3. **Proposed Method.** Usually, a softmax layer is employed as the last layer of a neural network, converting an input feature vector into a probability distribution vector. The sum of elements in the probability vector is 1.

3.1. **Softmax for CNN Segmentation Task.** Given a vector $\boldsymbol{o} = (o_1, o_2, \cdots, o_K) \in \mathbb{R}^K$, the softmax activation function $\mathcal{S} : \mathbb{R}^K \to \mathbb{R}^K$ is given by:

$$\mathcal{S}(o_k) = \frac{e^{o_k}}{\sum_{\hat{k}=1}^{K} e^{o_{\hat{k}}}}, k = 1, \cdots, K. \tag{20}$$

Given an image with size $N = N_1 \times N_2$ and N_1, N_2 is the height and width. If we want to segment the image into K classes using CNN, here comes the following minimization problem:

$$\min_{\mathcal{A} \in \mathbb{S}} \left\{ \sum_{i=1}^{N} \sum_{k=1}^{K} -a_{ik} \cdot o_{ik} + a_{ik} \cdot log(a_{ik}) \right\}, \tag{21}$$

where $\mathcal{A} = (a_{ik}) \in \mathbb{R}^{N \times K}$ is the activation function, $\mathbb{S}$ is the soft segmentation condition defined in (5), and $\boldsymbol{o} = (o_{ik}) \in \mathbb{R}^{N \times K}$ is the feature map taken as input. Eq. (21) could be rewritten as:

$$\min_{\mathcal{A} \in \mathbb{S}} \left\{ \sum_{k=1}^{K} - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k > \right\}, \tag{22}$$

where $\mathcal{A}_k \in \mathbb{R}^N$ is the k-th column of $\mathcal{A}$, $\boldsymbol{o}_k \in \mathbb{R}^N$ is the k-th column of $\boldsymbol{o}$. Solving the minimization problem in Eq. (22), the minimizer is

$$\mathcal{A}^*_{ik} = \frac{\exp(\boldsymbol{o}_{ik})}{\sum_{\hat{k}=1}^{K} \exp(\boldsymbol{o}_{i\hat{k}})}, i = 1, 2, \cdots N, k = 1, \cdots, K. \tag{23}$$

This is just the standard softmax activation function and we denote it as:

$$\mathcal{A}^* = \mathcal{S}(\boldsymbol{o}). \tag{24}$$

3.2. **Proposed Non-local TV Regularized Softmax Function.** Now we replace the edge force item in Eq. (18) with the function $\sum_{k=1}^{K} - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k >$ which is defined in Eq. (22) and regularize the prediction result by non-local total variation. We set the edge detector $\alpha(x)$ as a constant parameter λ, the regularized Softmax function is defined as:

$$\min_{\mathcal{A} \in \mathbb{S}} \left\{ \sum_{k=1}^{K} - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k > + \lambda NLTV(\mathcal{A}_k) \right\}. \tag{25}$$

The variational formulation of non-local total variation is given by

$$NLTV(\mathcal{A}_k) = \max_{\eta_k \in \mathbb{R}^{N \times N},\ ||\eta_k||_\infty \leq 1} < \mathcal{A}_k, div_w \eta_k >, \tag{26}$$

where $\eta_k \in \mathbb{R}^{N \times N}$ is the dual variable of $\mathcal{A}_k$. The minimization problem Eq. (25) can be reformulated as a saddle-point problem:

$$\min_{\mathcal{A} \in \mathbb{S}} \max_{||\eta_k||_\infty \leq \lambda} \left\{ \sum_{k=1}^{K} - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k > + < \mathcal{A}_k, div_w \eta_k > \right\}. \tag{27}$$

The above minimization problem can be solved by primal-dual hybrid gradient method updating dual variables η_k and primal variables $\mathcal{A}_k$ alternatively. The iteration is given by

$$\begin{cases} \eta_k^t = \Pi_{||\eta_k||_\infty \leq \lambda}(\eta_k^{t-1} - \tau \nabla_w \mathcal{A}_k^{t-1}), \ k = 1, \cdots, K \\ \mathcal{A}_k^t = \mathcal{S}(\boldsymbol{o}_k - div_w \eta_k^t), \end{cases} \tag{28}$$

where t is the iteration number.

We also record the primal energy and dual energy during the iteration to monitor the convergence of the algorithm. The primal energy $E_P(\mathcal{A})$ is as follows:

$$E_P(\mathcal{A}) = \sum_{k=1}^K - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k > + \lambda NLTV(\mathcal{A}_k). \tag{29}$$

The dual energy $E_D(\eta)$ is as follows:

$$E_D(\boldsymbol{\eta}) = \sum_{k=1}^K - < \mathcal{A}_k, \boldsymbol{o}_k > + < \mathcal{A}_k, log \mathcal{A}_k > + < \mathcal{A}_k, div_w(\eta_k) >, \tag{30}$$

where $\mathcal{A}_k = \mathcal{S}(\boldsymbol{o}_k - div_w(\eta_k))$.

There are two stopping criteria, a maximum of 1500 iterations is reached or the relative absolute duality gap is smaller than a threshold e, i.e.:

$$\frac{|E_P - E_D|}{|E_P|} \leq e, \tag{31}$$

where $e = 10^{-5}$ in our experiments.

Algorithm 1 Primal-Dual Hybrid Gradient Decent Method

Require: the output of last layer $\boldsymbol{o}$, initialize $\boldsymbol{\eta}^0 = \mathbf{0}, \mathcal{A}^0 = \mathcal{S}(\boldsymbol{o})$.
Ensure: $\mathcal{A}$

```
function Non-local Regularized Softmax
τ = 0.03, λ = 3,
for t = 1, ..., T + 1 until convergence do
    calculate ∇_w A,
    for k = 1, ..., K do
        η_k^t = Π_{||η_k||_∞ ≤ λ}(η_k^{t-1} − τ ∇_w A_k^{t-1}),  k = 1, ··· , K
    end for
    calculate div_w η^t,
    for k = 1, ..., K do
        calculate A_k^t = S(o_k − div_w η_k^t),
    end for
end for return A
end function
```

We iteratively perform Eq. (28), when it converges, we get the optimum of Eq. (27), $\mathcal{A}^* = \lim_{t \to +\infty} \mathcal{A}^t$,$\eta^* = \lim_{t \to +\infty} \eta^t$. Then we have the regularized softmax

$$\mathcal{A}^* = \mathcal{S}(\boldsymbol{o} - div_w(\eta^*)), \eta^* = (\eta_1^*, \eta_2^*, \cdots, \eta_K^*). \tag{32}$$

Replacing softmax with regularized softmax, we have regularized $\mathcal{A}^*$ and

$$\mathcal{A}_{ik}^* = \frac{e^{o_{ik} - div_w \eta_{ik}^*}}{\sum_{\hat{k}=1}^K e^{o_{i\hat{k}} - div_w \eta_{i\hat{k}}^*}}. \tag{33}$$

In our numerical experiments, we set $\tau = 0.03$ and $\lambda = 3$. Generally, we initially select a large λ and a small step size τ to perform the algorithm. Since the parameters λ and τ are image dependent, we iteratively finetune the parameters and finally select a best set of them. It is summarized as Algorithm 1.

Noticing that the second term in Eq. (25) could be seen as a negative entropy term which can enforce $\mathcal{A}$ to be smooth. If we add a control parameter $\epsilon > 0$ to it, Eq. (25) becomes

$$\min_{\mathcal{A}\in\mathbb{S}} \left\{ \sum_{k=1}^{K} - < \mathcal{A}_k, \boldsymbol{o}_k > + \epsilon < \mathcal{A}_k, log\mathcal{A}_k > + \lambda NLTV(\mathcal{A}_k) \right\}. \tag{34}$$

The corresponding minimizer is

$$\mathcal{A}^* = \mathcal{S}\left(\frac{\boldsymbol{o} - div_w \eta^*}{\epsilon} \right), \eta^* = (\eta_1^*, \eta_2^*, \cdots, \eta_K^*). \tag{35}$$

We can see that when adding a control parameter ϵ, it is equivalent to rescaling the output of regularized softmax by a factor $\frac{1}{\epsilon}$. In all our experiments, we set $\epsilon = 0.5$.

3.3. General Convolutional Neural Network for Semantic Image Segmentation.

A general convolution neural network consists sets of convolution layers and activation layers. Given an input v, the convolution layer can be formulated as:

$$\mathcal{T}(v) = \mathcal{W}v + b, \tag{36}$$

where $\mathcal{W}$ is a linear operator such as convolution or deconvolution, b is a bias.

The activation function takes $\boldsymbol{o}$ as input and outputs v, it can be represented by

$$v = \mathcal{A}(\boldsymbol{o}), \tag{37}$$

where $\mathcal{A}$ can be ReLU, softmax, sigmoid, sampling and other activation functions.

Given an image as input, a general convolution neural network with L layers can be described by recursive connections as follows:

$$\begin{cases} v^0 = & v, \\ o^l = & \mathcal{T}_{\boldsymbol{\Theta}^{l-1}}(v^{l-1}), \\ v^l = & \mathcal{A}^l(\boldsymbol{o}^l), l = 1, \ldots, L, \end{cases} \tag{38}$$

where $\boldsymbol{\Theta}$ is the parameter set, and we have $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}^l = (\mathcal{W}^l, b^l) |\ l = 0, \ldots, L-1\}$.

Given a training dataset and a loss function $\mathcal{L}$, the CNN learns a parameter set $\boldsymbol{\Theta}$ by iteratively training such that a loss functional $\mathcal{L}(\mathcal{N}_{\boldsymbol{\Theta}}(X), Y)$ is minimized by $\boldsymbol{\Theta}$. The training dataset consists of M images $X = (v_1, v_2, \ldots, v_M) \in \mathbb{R}^{M\times N_1 N_2}$ and their ground truth segmentation $Y = stack(y_1, y_2, \ldots, y_M) \in \{0,1\}^{M\times K\times N_1 N_2}$ with $y_m \in \{0,1\}^{K\times N_1 N_2}$.

A widely used loss function in many tasks is cross entropy which is given by

$$\mathcal{L}(\mathcal{N}_{\boldsymbol{\Theta}}(X), Y) = -\frac{1}{M} \sum_{m=1}^{M} < y_m, \log \mathcal{N}_{\boldsymbol{\Theta}}(x_m) > . \tag{39}$$

The algorithm of learning is a gradient descent method:

$$(\boldsymbol{\Theta}^l)^{step} = (\boldsymbol{\Theta}^l)^{step-1} - \tau_{\boldsymbol{\Theta}} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}^l}\Big|_{\boldsymbol{\Theta}^l = (\boldsymbol{\Theta}^l)^{step-1}}, \tag{40}$$

where $step = 1, 2, \ldots$ is the training iteration number and $\tau_{\boldsymbol{\Theta}}$ is a hyper parameter controlling learning rate. $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}^l}$ can be calculated by back-propagation technique

using chain rule. Let $\Delta^l = \frac{\partial \mathcal{L}}{\partial \boldsymbol{o}^l}$, then the back-propagation scheme is in the following

$$\begin{cases} \Delta^l &= \frac{\partial v^l}{\partial \boldsymbol{o}^l} \cdot \frac{\partial \boldsymbol{o}^{l+1}}{\partial v^l} \cdot \frac{\partial \mathcal{L}}{\partial \boldsymbol{o}^{l+1}} \\ &= \frac{\partial \mathcal{A}^l}{\partial \boldsymbol{o}^l} \cdot \frac{\partial \mathcal{T}_{\boldsymbol{\Theta}^l}}{\partial v^l} \cdot \Delta^{l+1}, \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}^l} &= \frac{\partial \boldsymbol{o}^{l+1}}{\partial \boldsymbol{\Theta}^l} \cdot \frac{\partial \mathcal{L}}{\partial \boldsymbol{o}^{l+1}} = \frac{\partial \mathcal{T}_{\boldsymbol{\Theta}^l}}{\partial \boldsymbol{\Theta}^l} \cdot \Delta^{l+1}, \end{cases} \tag{41}$$

where $l = 0, 1, \ldots, L-1$.

3.4. **Back-propagation of Regularized Softmax.** During the forward propagation stage, we obtain a regularized $\boldsymbol{o}$ by performing Algorithm 1 when given the output of last layer $\boldsymbol{o}$, $\boldsymbol{\eta}^0 = \mathbf{0}, \mathcal{A} = \mathcal{S}(\boldsymbol{o})$ as the initial values. The gradient of loss $\mathcal{L}$ with respect to $\boldsymbol{o}$ should be computed in the back-propagation stage. The for loop in Algorithm 1 is performed $T+1$ steps during each forward propagation iteration, the gradients are computed in an inverse order.

Since η^t only contributes to computing $\mathcal{A}^t$ when $t = 1, \ldots, T+1$, the gradient of $\mathcal{L}$ with respect to η^t is given by

$$\frac{\partial \mathcal{L}}{\partial \eta^t} = \frac{\partial \mathcal{L}}{\partial \mathcal{A}^t} \cdot \frac{\partial \mathcal{A}^t}{\partial \eta^t}, \ t = 1, \ldots, T+1. \tag{42}$$

Eq. (28) could be reformulated as :

$$\begin{cases} \xi_k^t &= \xi_k^{t-1} - \tau \nabla \mathcal{S}(\boldsymbol{o}_k - div_w \eta_k^{t-1}), \\ \eta_k^t &= \Pi_{||\xi_k||_\infty \le \lambda}(\xi_k^t), \\ \mathcal{A}_k^t &= \mathcal{S}\left(\boldsymbol{o}_k - div_w \eta_k^t\right). \end{cases} \tag{43}$$

ξ^t contributes to computing both η^t and ξ^{t+1} when $t = 1, \ldots, T$. However, ξ^{T+1} contributes to compute η^{T+1} only. Then the gradient of $\mathcal{L}$ with respect to ξ^t is given by

$$\frac{\partial \mathcal{L}}{\partial \xi^t} = \begin{cases} \frac{\partial \mathcal{L}}{\partial \eta^t} \cdot \frac{\partial \eta^t}{\partial \xi^t}, & t = T+1 \\ \frac{\partial \mathcal{L}}{\partial \eta^t} \cdot \frac{\partial \eta^t}{\partial \xi^t} + \frac{\partial \mathcal{L}}{\partial \xi^{t+1}}, & t = 1, \ldots, T. \end{cases} \tag{44}$$

$\mathcal{A}^t$ is the input to compute ξ^{t+1} when $t = 0, \ldots, T$, then the gradient of $\mathcal{L}$ with respect to $\mathcal{A}^t$ is given by

$$\frac{\partial \mathcal{L}}{\partial \mathcal{A}^t} = \frac{\partial \mathcal{L}}{\partial \xi^{t+1}} \cdot \frac{\partial \xi^{t+1}}{\partial \mathcal{A}^t}, \ t = 0, \ldots, T. \tag{45}$$

$\boldsymbol{o}$ contributes to computing each $\mathcal{A}^t$ when $t = 0, \ldots, T+1$. $\mathcal{A}^0$ is initialized with $\mathcal{S}(\boldsymbol{o})$, finally the gradient of $\mathcal{L}$ with respect to $\boldsymbol{o}$ is given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{o}} = \frac{\partial \mathcal{L}}{\partial \mathcal{A}^0} \cdot \mathcal{S}'(\boldsymbol{o}) + \sum_{t=1}^{T+1} \frac{\partial \mathcal{L}}{\partial \mathcal{A}^t} \cdot \mathcal{S}'(\boldsymbol{o} - div_w(\eta^t)). \tag{46}$$

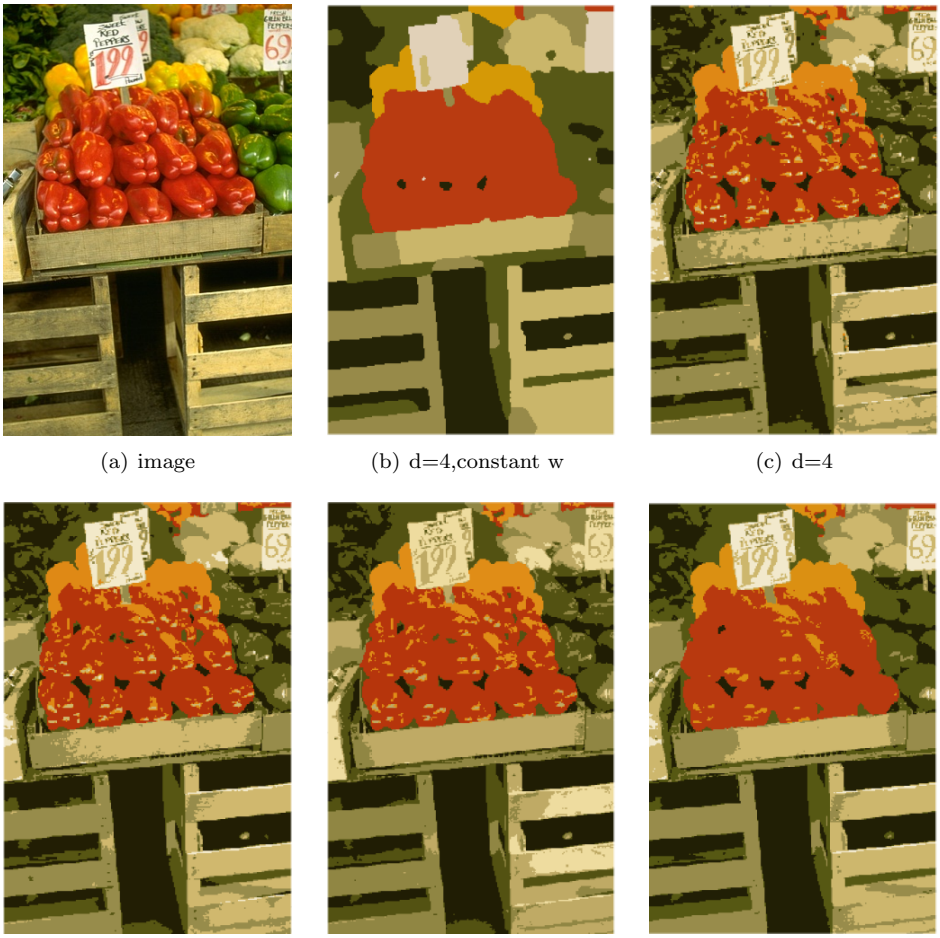
$\frac{\partial \mathcal{L}}{\partial \mathcal{A}^{t+1}}$ is given by the loss layer in the backward propagation stage, so we can successively get $\frac{\partial \mathcal{L}}{\partial \eta^{T+1}}, \frac{\partial \mathcal{L}}{\partial \xi^{T+1}}, \frac{\partial \mathcal{L}}{\partial \mathcal{A}^T}, \ldots, \frac{\partial \mathcal{L}}{\partial \eta^1}, \frac{\partial \mathcal{L}}{\partial \xi^1}, \frac{\partial \mathcal{L}}{\partial \mathcal{A}^0}$ by Eq. (42), Eq. (44) and Eq. (45).

At last, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{o}}$ is given by Eq. (46).

3.5. **Implementation Details.** Since the total variation in this paper is defined on graph, we treat each input image as a graph $G = (\mathbb{V}, \mathbb{E}, w)$ and each pixel is a vertex in $\mathbb{V}$. One essential problem is how to define a proper edge set $\mathbb{E}$ and weights of edges. Assuming that each pixel is connected to at most d neighbors and these neighbors are chosen according to distances between the feature vectors of pixels. Geometrical four nearest neighbors may not be among these d neighbors. When each pixel is connected to every other pixel, G is a fully connected graph. When each pixel is connected to only a few neighbor pixels, G is a sparse graph, then $\nabla_w \mathcal{A}_k$ and $div_w \eta_k$ are both sparse. We tried different d and found that a small d

could work well. We will show some experimental results of different d in Section 4.

When we introduce regularized softmax to CNN, we need to keep each $\mathcal{A}_k^t, \eta_k^t$ and some intermediate variables in graphics memory during forward propagation stage as they will be used to compute gradients in the backward propagation stage. Therefore, if d or t is too big, numerous computation and memory resources will be required. We use a small t and d in our experimental part, but there is still obvious regularization effect.



(a) image (b) d=4,constant w (c) d=4

(d) d=10 (e) d=20 (f) d=40

FIGURE 1. An example of segmentation results by applying the algorithm of [34] and our proposed method on an image from BSD500. When using 4 geometrical nearest neighbors, the weights are set to 1. The segmentation is quite smooth and missing details (Figure 1(b)). When we use Eq. (11) to compute W, the segmentation results are with more details and better accuracy.

The limited GPU memory can only store variables of no more than dozens of steps, we only perform Eq. (28) one or a few steps each iteration in the training stage which indeed brings regularization effect. What's more, even though the primal energy curve continuously decreases in hundreds of steps, the segmentation results change slightly after dozens of steps. It's a trade-off between accuracy, memory resources and efficiency. We set the initialization ξ^0 and η^0 to $\mathbf{0}$, respectively. Then the first iteration is

$$\begin{cases} \xi_k^1 & = -\tau \nabla \mathcal{S}(\boldsymbol{o}_k), \\ \eta_k^1 & = \Pi_{||\xi_k||_\infty \leq \lambda}(\xi_k^1), \\ \mathcal{A}_k^1 & = \mathcal{S}(\boldsymbol{o}_k - div_w \eta_k^1) \end{cases} \tag{47}$$

According to the back-propagation procedure described in Subsection 3.4, the gradient of $\mathcal{L}$ with respect to $\boldsymbol{o}$ could be computed easily.

4. **Experimental Results.** In our experiments, we rescale all the intensity of the images to [0,1]. First of all, we try different d and select a proper one by comparing the segmentation results from a toy example.

In [34] several images from BSD500 [19] were selected to test their algorithm. We use the same image for comparison. In their experiments, each pixel has 4 neighbors and the weights of edges are set to 1. In this experiments, we use Eq. (11) as our distance metric and select the nearest 4,10,20,40 neighbors to perform our algorithm, respectively.

The parameters in Eq. (11) are set as follows, $\lambda_1 = 1, \lambda_2 = 0.5, \sigma_\alpha = 40, \sigma_\beta = 13/255, \sigma_\gamma = 3$.

From Figure 1, we can see that, when using 4 geometrical neighbors with constant weights, the segmentation result is properly regularized and smoothed. There are not so many details. However, when using weights computed by Eq. (11), more details are preserved. When there is a few neighbors, the segmentation results appears to be a little noisy. There are many obvious isolated small regions on the vegetables and planks. The segmentation results appear to be smoother with increased number of neighbors. Nevertheless, a large number of neighbors need extra computation memory resources. In our experiments, we use $d = 20$ for WBC Dataset[37], $d = 10$ for CamVid Dataset[11].

We apply our proposed method to Unet, Attention Unet[23] and Segnet [3] using Caffe implementation. Unet, Unet with local regularization (RUnet)[10], Unet with non-local regularization (NLUnet), Attention Unet(AUnet), Attention Unet with local regularization(RAUnet), and Attention Unet with non-local regularization(NLAUnet) are tested on White Blood Cell Dataset [37]. Segnet, Segnet with local regularization(RSegnet)[10] and Segnet with non-local regularization(NLSegnet) are conducted on CamVid Dataset [11].

For each network, we use SGD solver with momentum of 0.9. We set the learning rates to 0.0001 for Unet and its variations, the weights of Unet, RUnet, AUnet and ARUnet is randomly initialized. The weights of NLUnet and NLAUnet are finetuned from Unet and AUnet, respectively. We set the learning rates to 0.001 for Segnet, RSegnet and NLSegnet. Like the author of Segnet, we also initialize the weights of Segnet and RSegnet from the VGG model trained on ImageNet [9]. The weights of NLSegnet is finetuned from Segnet.

In data preparation stage, we compute the affinity matrix for each image. Since the affinity matrix is sparse, we use two matrices to represent it. One is $W = (w_i)$, it keeps the edge weights computed by Eq. (11). The other is $Widx = (widx_i)$, it

keeps the indexes of nearest neighbors of the i-th pixel. We save the two matrices as local files so that we can load them during training and testing stages. Global pixel accuracy and mean intersection over union (mIoU) are two common metrics in image segmentation tasks, we also use them as our quantitative measures.

When evaluating a standard machine learning model, the prediction results are usually classified into four categories: true positives(TP), false positives(FP), true negatives(TN), and false negatives(FN). Global accuracy gives percent of pixels in all images which were correctly classified. The global accuracy is defined as

$$Accuracy = \frac{TP + FP}{TP + FP + TN + FN}. \tag{48}$$

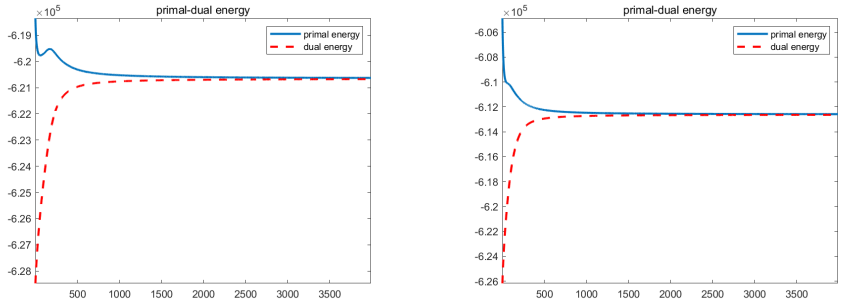
The Intersection over Union (IoU) metric, also called the Jaccard index, calculates the percent overlap between the ground truth mask and the prediction output. The IoU metric is defined by

$$IoU = \frac{TP}{TP + FP + FN}. \tag{49}$$

For multi-class segmentation tasks, the mean IoU(mIoU) is the mean value of the IoU of each class.

The RE score defined in the article[10] measures the regularization effect of segmentation result. Segmentation results with lower RE scores have smoother edges and less isolated regions.

4.1. **WBC Dataset.** There are two sub-datasets in White Blood Cell Image Dataset [37]. The image size in Dataset 1 is 120x120. It is too small for a CNN-based segmentation task. Dataset 2 consists of one hundred 300x300 color images. There is one white blood cell in the center of each image. Each image consists three classes, nucleus, cell sap and background. Comparing to Dataset 1, Dataset 2 is more suitable for a segmentation task thus selected in our experiments.



(a) Convergence of RSoftmax (b) Convergence of NLSoftmax

FIGURE 2. Given an input O, $\lambda = 3$ and $\tau = 0.03$,we perform algorithms for regularized softmax with local operator and non-local operator, respectively. Figure 2(a) is the convergence of softmax with local operate, the primal energy curve has a peak during the iteration. While in Figure 2(b), the energy curve drops rapidly at first and finally converges smoothly.

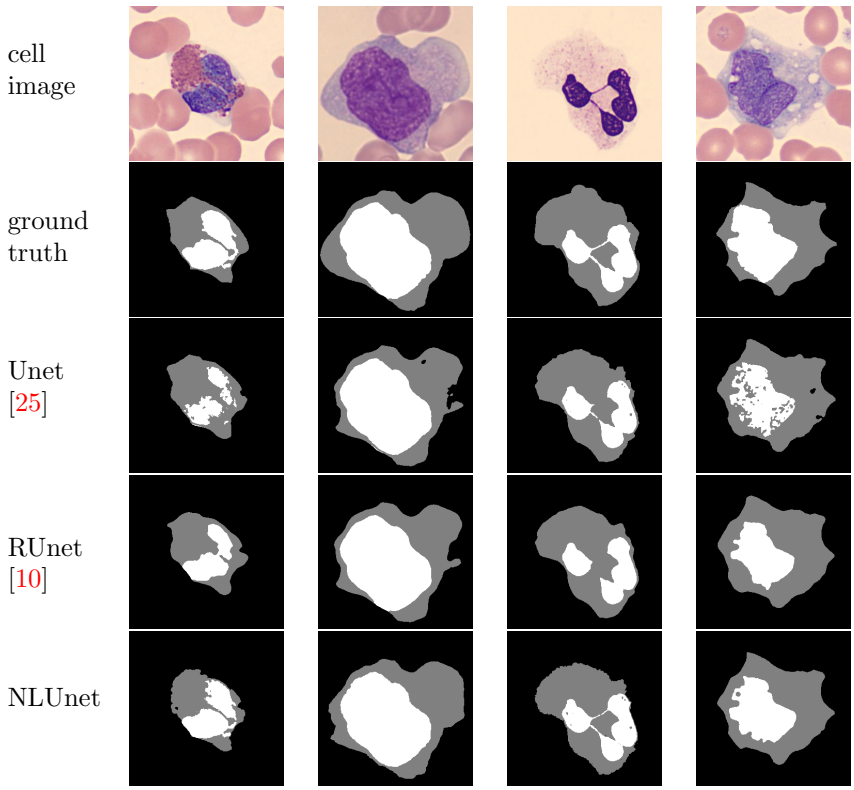


FIGURE 3. Segmentation results predicted by Unet, RUnet and NLUnet on images from testing dataset of White Blood Cell. From row 2 to row 5, The black regions are background, the gray regions are cell sap, the white regions are nucleus.

TABLE 1. Results of Unet, RUnet and NLUnet trained on WBC Dataset.

Method	Unet [25]	RUnet [10]	NLUnet
mIoU	89.79	90.15	**90.80**
Accuracy	97.04	97.13	**97.42**
RE	1.82	**1.30**	1.59

The training dataset contains 60 image randomly picked from WBC Datset2. The others are used for testing. We finetune Unet with non-local softmax(NLUnet) from Unet for 10k iterations. The CNN weights of Unet and RUnet are randomly initialized and they are trained for 20k iterations. Since the non-local softmax will take up some graphics memory for computing $\nabla_w \mathcal{A}$ and $div_w \boldsymbol{\eta}$, the mini-batch size is three.

Since the affinity matrix W measures the similarity between pixels, if the pixel color value is perturbed, W will become inaccurate and wrong pixels will be selected as nearest neighbors. In our experiments, all the image used in training and testing stages are clean image, no noise is added to them. From Table 1 we can see that

both mIoU and accuracy of NLUnet are improved compared to RUnet on testing dataset. The RE score of NLUnet is higher that RUnet, but less than Unet. This is because NLUnet could eliminate some isolated regions and produce smooth edges. Nevertheless, NLUnet can also preserve some details.

We show the convergence of RSoftmax and NLSoftmax in Figure 2(a) and Figure 2(b), respectively. The primal energy and dual energy of NLSoftmax are computed by Eq. (29) and Eq. (30), respectively. The primal energy and dual energy of RSoftmax are computed by the same equations after replacing the non-local operators ∇_w, div_w with local operators ∇, div. In Figure 2, the y-axis represents the energy value and the x-axis represents iteration number. Since we use a very small step size $\tau = 0.03$, the energy values of primal and dual functions converges with 1000 iterations. We can also use a larger step size to make them converges faster.

(a) Enlarged View (b) Ground Truth (c) Unet [25]

(d) RUnet [10] (e) NLUnet

FIGURE 4. An enlarged view of segmentation results from Figure 3.

In Figure 3, we can see that NLUnet provides more details comparing to RUnet. In Figure 3 column 1, the segmentation result of Unet misses some nucleus RUnet provides better segmentation results. The nucleus regions are closer to ground truth, but still some details are missed. NLUnet achieves the best segmentation result. There are less isolated regions and the edges are smoother comparing to Unet. Meanwhile, details are well preserved. Figure 4 is an enlarged view, we can see the segmentation details clearly. In Figure 3 column 2, we can see that a part of cell sap(grey region) is missing on the right hand side in both Unet and RUnet, while the segmentation result of NLUnet is relatively complete. In Figure

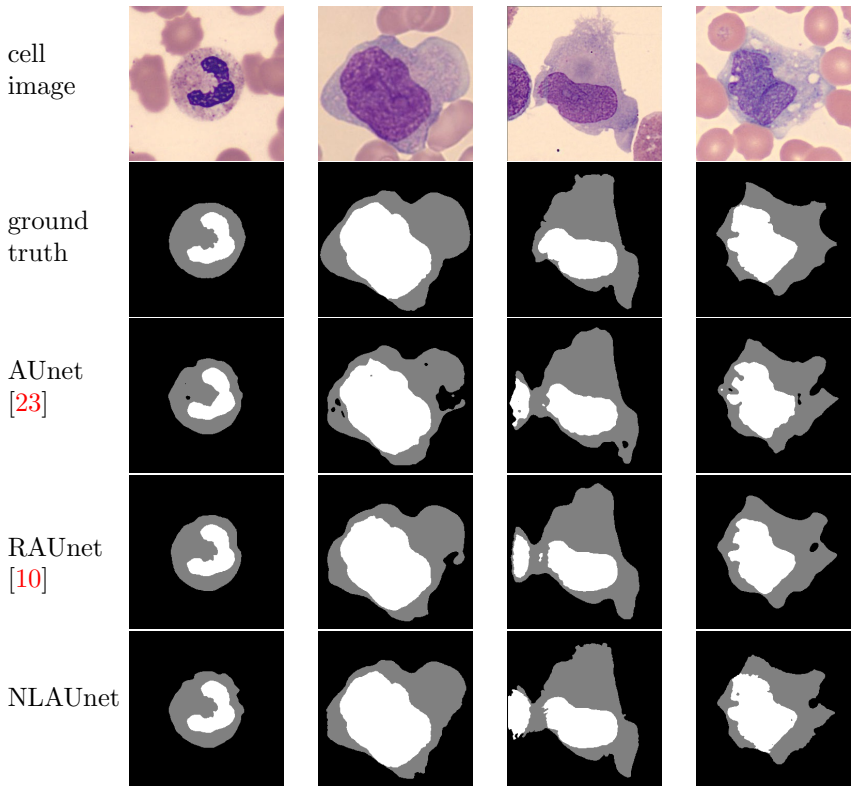


FIGURE 5. Segmentation results predicted by AUnet, RAUnet and NLAUnet on images from testing dataset of White Blood Cell. From row 2 to row 5, The black regions are background, the gray regions are cell sap, the white regions are nucleus.

3 column 3, there are two thin lines connecting different parts of nucleus(while region), Unet misses one of them and RUnet misses both of them. Surprisingly, NLUnet successfully preserves those details. If we take a closer look at the curves of nucleus and cell sap, we can see that the result of Unet is quite rough, RUnet gives much smoother edges. The edges provided by NLUnet are smoother than those of Unet, and more closer to ground truth comparing to RUnet. In Figure 3 column 4, we can see that the segmentation result of Unet is fragmented. RUnet gives a smooth segmentation result, but the nucleus is smaller comparing to ground truth due to its regularization effect. While NLUnet give relatively good result and the segmentation is closer to ground truth.

Since some variations of Unet appear in recent years, here we also use Attention Unet(AUnet)[23] to further evaluate the performance of our method. Comparing with original Unet, the Attention Unet introduces attention gates to help the network focus its attention on foreground. We simply add attention gates to Unet as the author do and use our own Caffe implementation. Excepting the attention gates, the other configurations are the same with Unet.

From Table 2 we can see that the attention gates help improve the performance of Unet. Nevertheless, the Attention Unet with local regularized softmax activation

TABLE 2. Results of AUnet, RAUnet and NLAUnet trained on WBC Dataset.

Method	AUnet [23]	RAUnet [10]	NLAUnet
mIoU	90.75	91.01	91.69
Accuracy	97.35	97.40	97.57
RE	1.43	1.41	1.43

function(RAUnet) and the Attention Unet with non-local regularized softmax activation function(NLAUnet) further improve the mIoU and accuracy. Our proposed method achieves the best result.

In Figure 5, we can see that the segmentations of nucleus(white regions) are very close to ground truth. Comparing with Unet in Figure 3, AUnet gives more complete nucleus. Inside the cell sap, there are some bubbles which looks very close to background. This may distract the attention gate such that some cell sap pixels are classified as background wrongly.

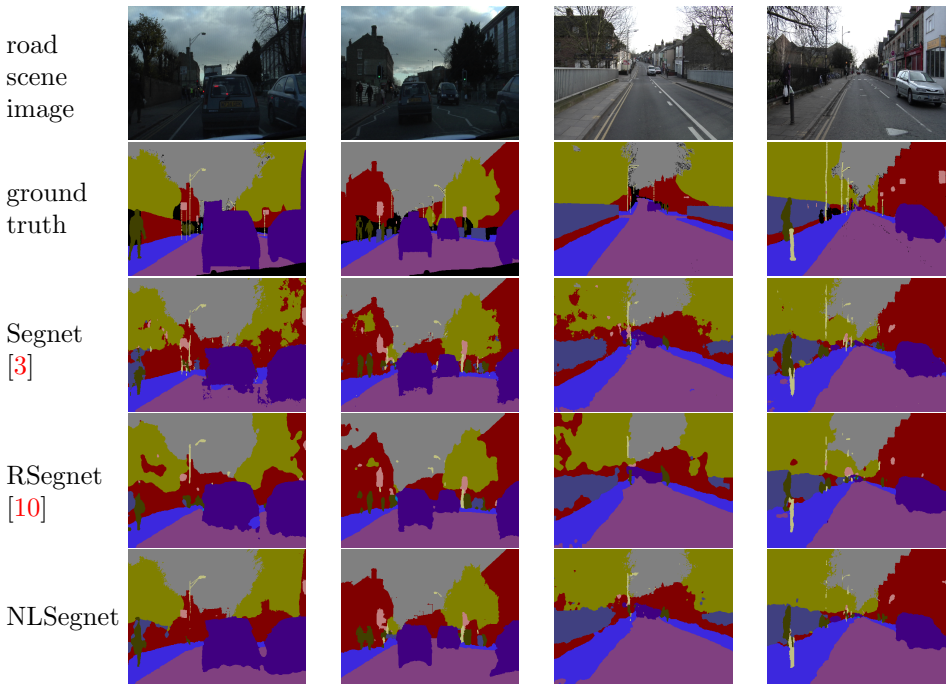


FIGURE 6. Segmentation results predicted by Segnet, RSegnet and NLSegnet trained on CamVid Dataset.

4.2. **CamVid Dataset.** CamVid Dataset [11] consists of a sequence of road scene images with size 360x480 collected by driving a car in the city of Cambridge. There are 367 images in the training dataset and 233 images in the testing dataset. This dataset contains 11 classes and pixels are ignored both in training stage and testing

stage if they don't belong to these 11 classes. The authors of Segnet choose this data as their benchmark dataset.

(a) Enlarged View

(b) Ground Truth

(c) Segnet [3]

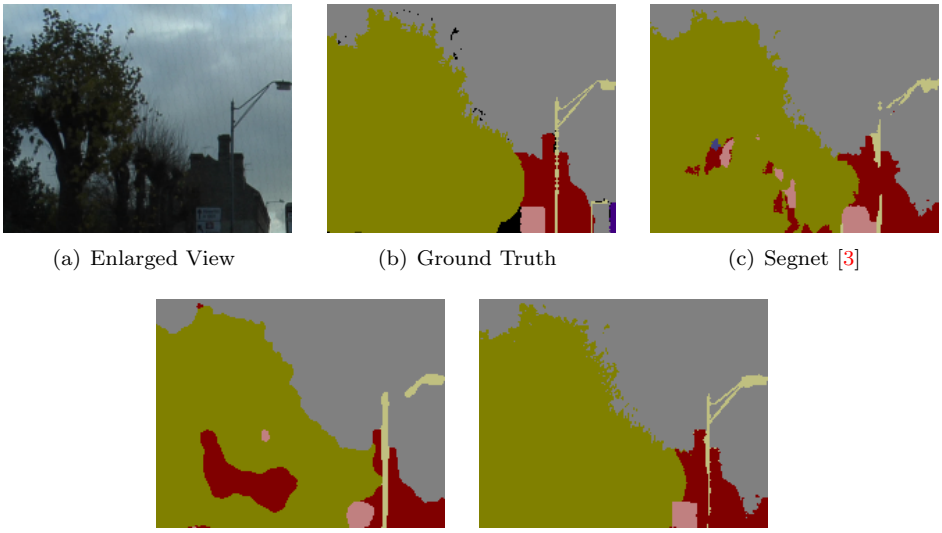
(d) RSegnet [10]

(e) NLSegnet

FIGURE 7. An enlarged view of segmentation results from Figure 6 column 2.

We apply our non-local regularized softmax layer to Segnet, other configurations remain the same. The initial weights of Segnet are finetuned from the VGG model trained on ImageNet, its mini-batch size is four. The CNN weights of NLSegnet is initialized from Segnet and finetuned for 3k iteration with learning rates fixed to 0.01. The mini-batch size of NLSegnet is 1.

From Table 3 we can see that both mIoU and accuracy of NLSegnet are improved compared to RSegnet on testing dataset. The RE score of NLSegnet is higher that



(a) Enlarged View (b) Ground Truth (c) Segnet [3]

(d) RSegnet [10] (e) NLSegnet

FIGURE 8. An enlarged view of segmentation results from Figure 6 column 1.

TABLE 3. Results of Segnet, RSegnet, NLSegnet trained on CamVid Dataset.

Method	Segnet [3]	RSegnet[10]	NLSegnet
mIoU	57.35	57.79	**59.84**
Accuracy	87.74	88.01	**88.59**
RE	4.10	**2.43**	3.40

RSegnet, but less than Segnet. The result is very similar to that of WBC dataset. But it is important to note that the mIoU is significantly improved from 57.79 to 59.84 by NLSegnet. Since mIoU measures the mean intersection over union of overall classes, the main gain in mIoU comes from classes which have small proportion pixels, such as pole and traffic sign. As non-local softmax could preserve more details, this can greatly benefit these minor classes.

In Figure 6, we can find that NLSegnet preserves many details such as tree branch, pole and roof top. In Figure 6 column, many isolated points and regions are removed in RSegnet and NLSegnet, there is a signal sign which is in pink color on the left hand side. The signal sign has a square shape which is well preserved by NLSegnet. Distinct details could be found in the enlarged view in Figure 8. However, the signal sign is distorted and becomes irregular in the segmentation results in Segnet and RSegnet. In Figure 6 column 2, the roof top on the left hand side is well preserved by NLSegnet, the segmentation result is nearly the same with ground truth. More details could be found in Figure 7. The segmentation result of Segnet is very coarse, whereas RSegnet gives smooth edges but some details are missed.

5. **Conclusions and Future Work.** Even though regularized softmax with local operators could eliminate scattered points, tiny regions and give smooth edges, some details are often missed. Inspired by regularized softmax with local operator, we successfully apply non-local operator to regularized softmax. After observing the experimental results on WBC Datset and CamVid Dataset, our proposed method obviously helps improve the performance of Unet, Attention Unet and Segnet. The proposed method not only inherits the regularization property from regularized softmax, but also showcases its prominent performance by preserving many more details. Since our method is a variation of softmax activation function, it is applicable to all networks with softmax. Especially, it can showcase its performance on small datasets with simple network structures. Now the parameters in computing the pairwise potential Eq. (11) is manually tuned. In the future, we will find a way to generate the affinity matrix W online efficiently and make the parameters in Eq. (11) trainable.

6. **Acknowledgment.** The work of Liu was supported by the National Key Research and Development Program of China (No. 2017YFA0604903) and the National Natural Science Foundation of China (No. 11871035). The work of Tai was supported by Hong Kong Baptist University through grants RG(R)-RC/17-18/02-MATH, HKBU 12300819 and NSF/RGC grant NHKBU214-19.

REFERENCES

[1] Rolf Adams and Leanne Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16** (1994), 641–647.

[2] Md Zahangir Alom, Mahmudul Hasan, Chris Yakopcic, Tarek M Taha, and Vijayan K Asari. Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation. arXiv:1802.06955.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv:1511.00561.

[4] Lauren Barghout and Lawrence Lee. Perceptual information processing system, March 25 2004. US Patent App. 10/618,543.

[5] Martin Benning, Christoph Brune, Martin Burger, and Jahn Müller. Higher-order tv methodsenhancement via bregman iteration. *Journal of Scientific Computing*, **54** (2013), 269–310.

[6] Harald Birkholz. A unifying approach to isotropic and anisotropic total variation denoising models. *Journal of Computational and Applied Mathematics*, **235** (2011), 2502–2514.

[7] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8** (1986), 679–698.

[8] Guy Gilboa and Stanley Osher. Nonlocal operators with applications to image processing. *Multiscale Modeling & Simulation*, **7** (2008), 1005–1028.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, (2015), 1026–1034.

[10] Fan Jia, Jun Liu, and Xue-cheng Tai. A regularized convolutional neural network for semantic image segmentation. arXiv:1907.05287.

[11] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? arXiv:1610.01983.

[12] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, **1**, (1988) 321–331.

[13] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. in *Advances in Neural Information Processing Systems*, JMLR.org, (2011), 109–117.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. in *Advances in Neural Information Processing Systems*, JMLR.org, (2012), 1097–1105.

[15] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, **1** (1989), 541–551.

[16] Guosheng Lin, Chunhua Shen, Anton Van Den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. in *Proceedings of the IEEE Conference on Computer Cision and Pattern Recognition*, IEEE, (2016), 3194–3203.

[17] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE (2015), 3431–3440.

[18] Marius Lysaker, Arvid Lundervold, and Xue-Cheng Tai. Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. *IEEE Transactions on Image Processing*, **12**, (2003), 1579–1590.

[19] David R Martin, Charless C Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26** (2004), 530–549.

[20] Karol Mikula, Alessandro Sarti, and Fiorella Sgallari. Co-volume level set method in subjective surface based medical image segmentation. in *Handbook of Biomedical Image Analysis*, Springer, (2005), 583–626.

[21] David Mumford and Jayant Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, **42** (1989), 577–685.

[22] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. in *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, (2015), 1520–1528.

[23] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. arXiv:1804.03999.

[24] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, **9** (1979), 62–66.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, (2015), 234–241.

[26] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, **60** (1992), 259–268.

[27] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Support vector machine applications in computational biology*. MIT press, 2004.

[28] Linda Shapiro and George C Stockman. *Computer Vision*. Prentice Hall, 2001.

[29] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*,**22** (2000), 888-908.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

[31] Markus Unger, Thomas Mauthner, Thomas Pock, and Horst Bischof. Tracking as segmentation of spatial-temporal volumes by anisotropic weighted tv. in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Springer (2009), 193–206.

[32] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, (2018), 1451–1460.

[33] Ke Wei, Ke Yin, Xue-Cheng Tai, and Tony F Chan. New region force for variational models in image segmentation and high dimensional data clustering. arXiv:1704.08218.

[34] Ke Yin and Xue-Cheng Tai. An effective region force for some variational models for learning and clustering. *Journal of Scientific Computing*, **74** (2018), 175–196.

[35] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. arXiv:1511.07122.

[36] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. in *Advances in Neural Information Processing Systems*, JMLR.org, (2005),1601–1608.

[37] Xin Zheng, Yong Wang, Guoyou Wang, and Jianguo Liu. Fast and robust segmentation of white blood cell images by self-supervised learning. *Micron*, **107** (2018), 55–71.

Received xxxx 20xx; revised xxxx 20xx.

E-mail address: jiafan@life.hkbu.edu.hk
E-mail address: xuechengtai@hkbu.edu.hk
E-mail address: jliu@bnu.edu.cn