Connections between Operator-splitting Methods and Deep Neural Networks with Applications in Image Segmentation

Hao Liu*, Xue-Cheng Tai†, Raymond Chan ‡

In memory of Prof. Zhongci Shi

Abstract

Deep neural network is a powerful tool for many tasks. Understanding why it is so successful and providing a mathematical explanation is an important problem and has been one popular research direction in past years. In the literature of mathematical analysis of deep neural networks, a lot of works is dedicated to establishing representation theories. How to make connections between deep neural networks and mathematical algorithms is still under development. In this paper, we give an algorithmic explanation for deep neural networks, especially in their connections with operator splitting. We show that with certain splitting strategies, operator-splitting methods have the same structure as networks. Utilizing this connection and the Potts model for image segmentation, two networks inspired by operator-splitting methods are proposed. The two networks are essentially two operator-splitting algorithms solving the Potts model. Numerical experiments are presented to demonstrate the effectiveness of the proposed networks.

1 Introduction

In past decades, deep neural network has emerged as a very successful technique for various fields. It has demonstrated impressive performances in many tasks, such as image processing, object detection, and natural language processing. In some tasks, deep neural networks even outperform humans.

Due to great successes of neural networks, over the past several years, a lot of works has been devoted to the mathematical understanding of neural networks and explaining their

*Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong. Email: haoliu@hkbu.edu.hk. The work of Hao Liu is partially supported by HKBU 179356, NSFC 12201530 and HKRGC ECS 22302123.

†Norwegian Research Centre (NORCE), Nygårdsgaten 112, 5008 Bergen, Norway. Email: xtai@norceresearch.no, xuechengtai@gmail.com. The work of Xue-Cheng Tai is partially supported by NSFC/RGC grant N-HKBU214-19 and NORCE Kompetanseoppbygging program.

‡Department of Mathematics, City University of Hong Kong and Hong Kong Center for Cerebro-Cardiovascular Health Engineering, Hong Kong SAR. raymond.chan@cityu.edu.hk. The work of Raymond Chan is partially supported by HKRGC GRF grants CityU1101120, CityU11309922, CRF grant C1013-21GF, and HKRGC-NSFC Grant N_CityU214/19.

success. Representation theories for function learning are studied in [4, 17, 38, 46, 65] for feedforward neural networks, in [69, 70] for convolutional neural networks, and in [36, 37, 54] for convolutional residual networks. Recently, theoretical results for learning operators are developed in [5, 33, 44]. The works mentioned above show that as long as the network depth and width are sufficiently large, deep neural networks can approximate any function or operator within a certain class to arbitrary accuracy. These works focus on the existence of a good approximator with desired approximation error and use techniques from approximation theory. In this paper, we investigate the power of neural networks from another perspective: the network structure. We will give an algorithmic explanation of neural networks, especially in their connections with operator-splitting algorithms.

The operator-splitting method is a class of powerful methods for numerically solving complicated problems. The general idea is to decompose a difficult problem into several subproblems which will be solved sequentially or simultaneously. Operator-splitting methods have been widely used on solving partial differential equations [24, 26, 47, 49, 63], image processing [21, 22, 40, 41], surface reconstruction [30], obstacle problem [43], inverse problem [25], and computational fluid dynamics [6, 52], etc. We refer readers to [28, 48] for some survey discussions.

Image segmentation is an important subject in many fields, such as medical imaging and object detection. Many mathematical models and algorithms have been developed for image segmentation [10, 11, 13, 14, 15, 31, 53, 62]. In [2, 67], the segmentation problem is formulated as a min cut or max flow problem. One important model for image segmentation is the Potts model, which was first proposed for statistical mechanics [56]. In fact, the well-known Chan-Vese model [15] is a special case of the Potts model. In [64], detailed explanations are given to show that the Potts model is equivalent to a continuous min cut and max flow problem [64]. Efficient algorithms for the Potts model are studied in [59, 67]. We suggest readers to survey [60] for a comprehensive discussion on the Potts model. Recently, many deep learning methods for image segmentation are also proposed [23, 57, 71].

Following [61] and [39], we focus on the building block of neural networks and operator-splitting methods and make connections between them. We first introduce the structure of a standard neural network in this work. Then we discuss popular operator-splitting strategies: sequential splitting and parallel splitting. We show that for certain splitting strategies, the resulting splitting algorithm is equivalent to a neural network, whose depth and width are determined by the splitting strategy. Such a connection is also observed and utilized in [32], which is used to solve nonlinear partial differential equations by neural networks. We will apply this connection to the Potts model for image segmentation, and propose two networks inspired by operator-splitting methods. As the proposed networks are derived from the Potts model, they contain explicit regularizers that have physical meaning. The effectiveness of the proposed networks are demonstrated by numerical experiments.

This paper is structured as follows: We briefly introduce deep neural networks and operator-splitting methods in Section 2 and 3, respectively. The Potts model for image segmentation is discussed in Section 4. We present the two networks inspired by operator-splitting methods in Section 5 and 6, respectively. This paper is concluded in Section 7.

2 Deep neural networks

Deep neural networks have been successfully applied to many problems and have achieved remarkable performances. There are many networks architectures, such as feedforward neural networks (FNN) [20], convolutional neural networks (CNN) [34] and residual neural networks (ResNet) [29]. It is easy to show that any CNN and ResNet can be realized by FNN [37, 54, 69, 70]. In this paper, we focus on FNN and show their connections with operator-splitting methods.

The building blocks of FNNs are layers. An FNN is a composition of multiple layers, each of which takes an input $\mathbf{x} \in \mathbb{R}^d$ and outputs a vector in $\mathbb{R}^{d'}$ for some integers $d, d' > 0$. Each layer has the form of

$$\mathcal{L}(W, \mathbf{b}, \sigma; \mathbf{x}) = \sigma(W\mathbf{x} + \mathbf{b}), \tag{1}$$

where $W \in \mathbb{R}^{d \times d'}$ is a weight matrix, $\mathbf{b} \in \mathbb{R}^{d'}$ is a bias term, and $\sigma(\cdot)$ is the activation function. Popular choices of σ include the rectified linear unit (ReLU), sigmoid function and tanh function. The computation of a layer consists of two parts, a linear part $W\mathbf{x}+\mathbf{b}$ and a nonlinear part $\sigma(\cdot)$. These two parts are conducted sequentially. Later we will show the similarity between this structure and operator splitting methods.

An FNN with depth L is a composition of $L-1$ layers followed by a fully connected layer:

$$f(\mathbf{x}) = W_L \mathcal{L}_L(W_{L-1}, \mathbf{b}_{L-1}, \sigma_{L-1}) \circ \cdots \circ \mathcal{L}_1(W_1, \mathbf{b}_1, \sigma_1; \mathbf{x}) + \mathbf{b}_L, \tag{2}$$

where $W_l \in \mathbb{R}^{d_{l-1} \times d_l}$, $\mathbf{b}_l \in \mathbb{R}^{d_l}$ and σ_l denote the weight matrix, bias and activation function in the l-th layer with $d_0 = d$ being the input dimension and d_L being the output dimension. We call $\max_l d_l$ the width of f. The computation of $f(\mathbf{x})$ can be taken as passing the input $\mathbf{x}$ to L layers sequentially.

3 Operator-splitting methods

The operator-splitting method is a powerful method for solving complicated problems, such as time evolution problems and optimization problems. The general idea is to decompose a complicated problem into several easy-to-solve subproblems, so that each subproblem can be solved explicitly or efficiently. The first operator-splitting method according to [19] is the famous Lie scheme introduced in [35] to solve the initial problem from the dynamical system:

$$\begin{cases} \dfrac{dX}{dt} + (A + B)X = 0 \text{ in } (0, T], \\ X(0) = X_0, \end{cases} \tag{3}$$

where $X_0 \in \mathbb{R}^d$ for some integer $d > 0$, and $A, B \in \mathbb{R}^{d \times d}$. In the Lie scheme, one solves (3) by decomposing A and B into two subproblems. In each subproblem, X is governed by one operator and evolves for a small time step. One can show that this scheme is first-order accurate in time. Later, a second order splitting scheme, the Strang scheme, was introduced in [58]. We refer this type of splitting scheme as sequential scheme as the subproblems are solved sequentially. Another type of splitting strategy is parallel splitting [47], which solves the subproblems simultaneously and then takes the average.

Such a method is proved to be first-order accurate in time. Due to the simplicity of the ideas and versatility, operator-splitting methods have been widely used in solving partial differential equations [6, 9, 26, 47, 49] in which the operators in the PDE are decomposed into subproblems.

The operator-splitting method is also a popular choice to solve optimization problems. Consider an optimization problem in which one needs to minimize a functional. One may first derive the optimality condition of the minimizer and associate it with an initial value problem in the flavor of gradient flow. Then solving the optimization problem is converted into finding the steady-state solution of the initial value problem. The initial value problem can be solved by operator-splitting methods. Such a strategy has been used in [21, 22, 30, 41, 42]. For optimization problems, the alternating direction method of multipliers (ADMM) has been extensively studied in past decades [1, 59, 66, 67]. Indeed, ADMM is a special type of operator-splitting methods. We suggest readers to [27, 28] for a comprehensive discussion on operator-splitting methods.

In the rest of this section, we focus on the following initial value problem

$$\begin{cases} \dfrac{\partial u}{\partial t} = \displaystyle\sum_{k=1}^{K} (A_k u + B_k(u) + g_k) \text{ on } \Omega \times (0, T], \\ u(0) = u_0, \end{cases} \tag{4}$$

where $K > 0$ is a positive integer, Ω is our computational domain, $A_k u$ is a linear operator of u, $B_k(u)$ is an operator on u that might be nonlinear, and g_k's are functions defined on Ω. We will discuss sequential splitting and parallel splitting schemes for solving (4), and show their connections to deep neural networks. In the following, assume the time domain is discretized into N subintervals. Denote $\Delta t = T/N$ and $t^n = n\Delta t$. We use u^n to denote our numerical solution at t^n.

3.1 Sequential splitting

For sequential splitting, a simple one is the Lie scheme. Given u^n, we can update u^{n+1} by K substeps:
For $k = 1, ..., K$, solve

$$\begin{cases} \dfrac{\partial v}{\partial t} = A_k v + B_k(v) + g_k \text{ on } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{k-1}{K}}, \end{cases} \tag{5}$$

and set $u^{n+\frac{k}{K}} = v(t^{n+1})$. One can show that when B_k's are linear operators and (5) is time-discretized by the forward Euler method, scheme (5) is first-order accurate in time.

Equation (5) is the building block for this scheme. Note that to solve (5), we can further apply a Lie scheme. Starting from $u^{n+\frac{k-1}{K}}$, we update $u^{n+\frac{k}{K}}$ by two parts $u^{n+\frac{k-1}{K}} \to \bar{u}^{n+\frac{k}{K}} \to u^{n+\frac{k}{K}}$:
Part 1: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} = A_k v + g_k \text{ on } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{k-1}{K}}, \end{cases} \tag{6}$$

and set $\bar{u}^{n+\frac{k}{K}} = v(t^{n+1})$.

Part 2: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} = B_k(v) \text{ on } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = \bar{u}^{n+\frac{k}{K}}, \end{cases} \tag{7}$$

and set $u^{n+\frac{k}{K}} = v(t^{n+1})$.

By time-discretizing (6) by the forward Euler method, we get

$$\frac{\bar{u}^{n+\frac{k}{K}} - u^{n+\frac{k-1}{K}}}{\Delta t} = A_k u^{n+\frac{k-1}{K}} + g_k, \tag{8}$$

implying that

$$\begin{aligned} \bar{u}^{n+\frac{k}{K}} &= u^{n+\frac{k-1}{K}} + \Delta t A_k u^{n+\frac{k-1}{K}} + \Delta t g_k \\ &= (I + \Delta t A_k) u^{n+\frac{k-1}{K}} + \Delta t g_k, \end{aligned} \tag{9}$$

where I denotes the identity operator: $Iu = u$.

By time-discretizing (7) using the backward Euler method, we get

$$\frac{u^{n+\frac{k}{K}} - \bar{u}^{n+\frac{k}{K}}}{\Delta t} = B_k(u^{n+\frac{k}{K}}). \tag{10}$$

Denoting the resolvant operator for $u^{n+\frac{k}{K}}$ by ρ_k, i,e. $\rho_k = (I - \Delta t B_k)^{-1}$, we have

$$u^{n+\frac{k}{K}} = \rho_k(\bar{u}^{n+\frac{k}{K}}) = \rho_k((I + \Delta t A_k) u^{n+\frac{k-1}{K}} + \Delta t g_k). \tag{11}$$

Connections to FNN. Comparing (11) and (1), we see that they have the same structure. By choosing,

$$L = K + 1, W_k = (I + \Delta t A_k), \ \mathbf{b}_k = \Delta t g_k, \ \sigma_k = \rho_k \tag{12}$$

and $W_{K+1} = I$, $\mathbf{b}_{K+1} = 0$, the sequential splitting scheme is an FNN with $K+1$ layers, where the k-th substep corresponds to the k-th layer. In other words, any FNN with $W_{K+1} = I$, $\mathbf{b}_{K+1} = 0$ and activation functions ρ_k's is a sequential splitting scheme solving some initial value problem.

3.2 Parallel splitting

Using parallel splitting [47], we solve (4) by first solving K parallel substeps:
For $k = 1, ..., K$, solve

$$\begin{cases} \dfrac{\partial v}{\partial t} = K A_k v + K B_k(v) + K g_k \text{ on } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^n \end{cases} \tag{13}$$

and set $u^{n+1,k} = v(t^{n+1})$. Then compute

$$u^{n+1} = \frac{1}{K} \sum_{k=1}^{K} u^{n+1,k}. \tag{14}$$

One can show that when B_k's are linear operators and (13) is solved by the forward Euler method, the parallel splitting scheme is first-order accurate in time. Note that the number of operators K is multiplied into the right hand side of (13).

Similar to (6)–(7), we can use the same idea to solve (13). After some derivations, we get

$$u^{n+1,k} = \rho_k \left((I + \Delta t K A_k) u^n + \Delta t K g_k \right) \tag{15}$$

and

$$u^{n+1} = \frac{1}{K} \sum_{k=1}^{K} \rho_k \left((I + \Delta t K A_k) u^n + \Delta t K g_k \right). \tag{16}$$

Connections to FNN. Again, (15) and (1) have the same structure. Set $L = 2$,

$$W_1 = \begin{bmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & \cdots & \cdots & A_K \end{bmatrix}, \ \mathbf{b}_1 = \begin{bmatrix} g_1 \\ \vdots \\ g_K \end{bmatrix} \ \sigma_1 = \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_K \end{bmatrix}, \ W_2 = \frac{1}{K} I, \ \mathbf{b}_2 = 0, \tag{17}$$

where σ_1 is applied elementwise:

$$\sigma_1 \left(\begin{bmatrix} u_1 \\ \vdots \\ u_K \end{bmatrix} \right) = \begin{bmatrix} \rho_1(u_1) \\ \vdots \\ \rho_K(u_K) \end{bmatrix}, \tag{18}$$

then the parallel splitting scheme is an FNN with 2 layers. In other words, any 2-layer FNN with $\sigma_1, W_2, \mathbf{b}_2$ given in the format of (17) is a parallel splitting scheme solving some initial value problems.

4 Potts model and image segmentation

We next focus on image segmentation and utilize the relations between operator-splitting methods and FNNs to derive new networks.

We start with the Potts model which has close relations to a large class of image segmentation models. Let Ω be the image domain. The continuous two-phase Potts model is given as [3, 7, 8, 12, 68]

$$\begin{cases} \min\limits_{\Sigma_0, \Sigma_1} \left\{ \sum\limits_{k=0}^{1} \int_{\Sigma_k} f_k(\mathbf{x}) d\mathbf{x} + \dfrac{1}{2} \sum\limits_{k=0}^{1} \mathrm{Per}(\Sigma_k) \right\}, \\ \Sigma_0 \cup \Sigma_1 = \Omega, \qquad \Sigma_0 \cap \Sigma_1 = \emptyset, \end{cases} \tag{19}$$

where $\mathrm{Per}(\Sigma_k)$ is the perimeter of Σ_k, and f_k's are nonnegative weight functions. In (19), the first term is a data fidelity term which depends on the input image f. The second term is a regularization term which penalizes the perimeter of the segmented region.

A popular choice of f_k is

$$f_k(\mathbf{x}) = (f(\mathbf{x}) - c_k)^2,$$

where c_k is the estimated mean density of $f(\mathbf{x})$ on Σ_k^*. Here we use Σ_k^* to denote the 'optimal' segmentation region. With this choice and the use of a binary function v to represent the foreground of the segmentation results, we rewrite the two-phase Potts model as

$$\min_{v\in\{0,1\}} \left\{ \int_\Omega (f(\mathbf{x})-c_0)^2 v + (f(\mathbf{x})-c_1)^2(1-v)d\mathbf{x} + \lambda \int_\Omega |\nabla v| d\mathbf{x} \right\} \tag{20}$$

for some constant $\lambda > 0$. Model (20) is the well-known Chan-Vese model [16].

The first integral in (20) is linear in v. As one can take c_k and Σ_k^* as functions depending on f, in this case, f_k's in (19) are functions of the input image f only. We can thus rewrite the Potts model as

$$\min_{v\in\{0,1\}} \int_\Omega F(f) v d\mathbf{x} + \lambda \int_\Omega |\nabla v| d\mathbf{x}, \tag{21}$$

where $F(f)$ is a region force depending on the input image f. In the following sections, we will discuss two relaxations of the Potts model (21) and correspondingly propose two new networks for image segmentation.

5 Operator-splitting method inspired networks for image segmentation: Model I

Our first model utilizes (21) and the double-well potential and follows [39].

5.1 Model formulation

Model (21) requires the function v to be binary. One relaxation of (21) using the double-well potential is

$$\min_v \int_\Omega F(f) v d\mathbf{x} + \lambda \mathcal{L}_\varepsilon(v) d\mathbf{x}, \tag{22}$$

with

$$\mathcal{L}_\varepsilon(v) = \int_\Omega \left[\frac{\varepsilon}{2} |\nabla v|^2 + \frac{1}{\varepsilon} v^2 (1-v)^2 \right] d\mathbf{x}.$$

It is shown in [50, 51] that $\mathcal{L}_\varepsilon(v)$ converges to $C\mathrm{Per}(\Sigma_1)$ in the sense of Γ-convergence as $\varepsilon \to 0$ for some constant C.

Denote the minimizer of (22) by u. It satisfies the optimality condition

$$F(f) - \lambda\varepsilon\nabla^2 u + \frac{2\lambda}{\varepsilon}(2u^3 - 3u^2 + u) = 0 \text{ in } \Omega. \tag{23}$$

The gradient flow equation for minimization problem (22) is:

$$\begin{cases} \dfrac{\partial u}{\partial t} + F(f) - \lambda\varepsilon\nabla^2 u + \dfrac{2\lambda}{\varepsilon}(2u^3 - 3u^2 + u) = 0 \text{ in } \Omega \times (0,T], \\ \dfrac{\partial u}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \\ u(0) = G(f), \end{cases} \tag{24}$$

for some initial condition $G(f)$, which is a function of f. Then solving (23) is equivalent to finding the steady state solution of (24). As was done in [61], We add control variables to (24) so that we can use these control variables to lead the final state $u(T)$ of the following equation to some desired targets:

$$\begin{cases} \dfrac{\partial u}{\partial t} + F(f) - \lambda\varepsilon\nabla^2 u + \dfrac{2\lambda}{\varepsilon}(2u^3 - 3u^2 + u) + W(t) * u + b(t) = 0 \text{ in } \Omega \times (0, T], \\ \dfrac{\partial u}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \\ u(0) = u_0. \end{cases} \tag{25}$$

Normally, the function $F(f)$ is a complicated function of f. We use a subnetwork to represent $F(\cdot)$. As usual, here and later $*$ denote the spatial convolution operator.

5.2 An operator splitting method to solve (25)

We use the Lie scheme to solve (25). Still discretize the time variable as in Section 3. Given u^n, we update $u^n \to u^{n+1/2} \to u^{n+1}$ in the following way:
Substep 1: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + F(f) - \lambda\varepsilon\nabla^2 v + W(f, t) * u + b(t) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ \dfrac{\partial v}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \\ v(t^n) = u^n, \end{cases} \tag{26}$$

and set $u^{n+1/2} = v(t^{n+1})$.
Substep 2: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + \dfrac{2\lambda}{\varepsilon}(2v^3 - 3v^2 + v) = 0 \text{ on } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+1/2}, \end{cases} \tag{27}$$

and set $u^{n+1} = v(t^{n+1})$.

5.3 Time discretization

We use a one-step forward Euler scheme to time-discretize (26) and a one-step backward Euler scheme to time-discretize (27):

$$\begin{cases} \dfrac{u^{n+1/2} - u^n}{\Delta t} + F(f) - \lambda\varepsilon\nabla^2 u^n + W^n * u^n + b^n = 0, \\ \dfrac{\partial u^{n+1/2}}{\partial \mathbf{n}} = 0, \end{cases} \tag{28}$$

$$\frac{u^{n+1} - u^{n+1/2}}{\Delta t} + \frac{2\lambda}{\varepsilon}(2(u^{n+1})^3 - 3(u^{n+1})^2 + u^{n+1}) = 0. \tag{29}$$

In image processing, the periodic boundary condition is widely used. Here we replace the Neumann boundary condition in (28) by the periodic one. In (28), index n is used at the superscript of W^n and b^n to differentiate control variables at different time.

	Model I	UNet	UNet++	MANet
accuracy	**95.91%**	94.79%	95.66%	95.33%
dice score	**0.9048**	0.8776	0.8992	0.8936

Table 1: Comparison of Model I with other networks in terms of accuracy and dice score.

In our algorithm, we choose $G(f)$ as a convolution layer of f, i.e., a convolution of f with a 3×3 kernel followed by a sigmoid function. Suppose we are given a set of images $\{f_i\}_{i=1}^I$ and the corresponding segmentation masks $\{h_i\}_{i=1}^I$. Note that both F and G are networks and contain learnable parameters. Denote the set of control variables by $\theta_1 = \{\{W^n, b^n, \}_{n=1}^N, \theta_F, \theta_G\}$, where θ_F, θ_G denote the all network parameters in F and G. Denote the procedure of numerically solving (38) with N time steps and control variables θ_1 by

$$\mathcal{N}_1(\theta_1; \cdot) : f \to u^0 \to u^1 \to \cdots \to u^N.$$

We will learn θ_1 by solving

$$\min_{\theta_1} \frac{1}{I} \sum_{i=1}^{I} \ell(\mathcal{N}_1(\theta_1; f_i), h_i), \tag{30}$$

where $\ell(\cdot, \cdot)$ is some loss function, such as the cross entropy.

5.4 Connections to neural networks

The building block for scheme (26)–(27) consists of a linear step (28) and a nonlinear step (28). Thus a time stepping of u^n corresponds to a layer of a network with (29) being the activation function. Unlike the commonly used neural networks, the equivalent network of Model I has a heavy bias term $F(f)$ which is chosen as a subnetwork in this paper. The process (30) of learning θ_1 is the same as training a network.

5.5 Numerical experiments

We demonstrate the effectiveness of Mode I. In our experiments, we choose $G(f)$ as a convolution layer of f. The functional $F(f)$ needs to be complicated enough to approximate the probability function in the Potts model. Here we set $F(f)$ as a UNet [57] subnetwork of f, i.e., F has the same architecture as a UNet, takes f as the input and ouputs a matrix having the same size as u with elements in $[0, 1]$, see [39] for details. The W^n's and b^n's are convolution kernels and biases, respectively. For parameters, we set $\Delta t = 0.2, \lambda\varepsilon = 1, \lambda/\varepsilon = 15$.

We use the MSRA10K dataset [18] which contains 10000 salient object images with manually annotated masks. We choose 2500 images for training and 400 images for testing. All images and masks are resized to 192×256.

We compare Model I with UNet [57], UNet++ [71] and MANet [23]. All models are trained with 400 epochs. The comparison of accuracy and the dice score are shown in Table 1. We observe that Model I has higher accuracy and dice score than other networks. Some segmentation results are presented in Figure 1. Model I successfully segments the targets from images with complicated structures.

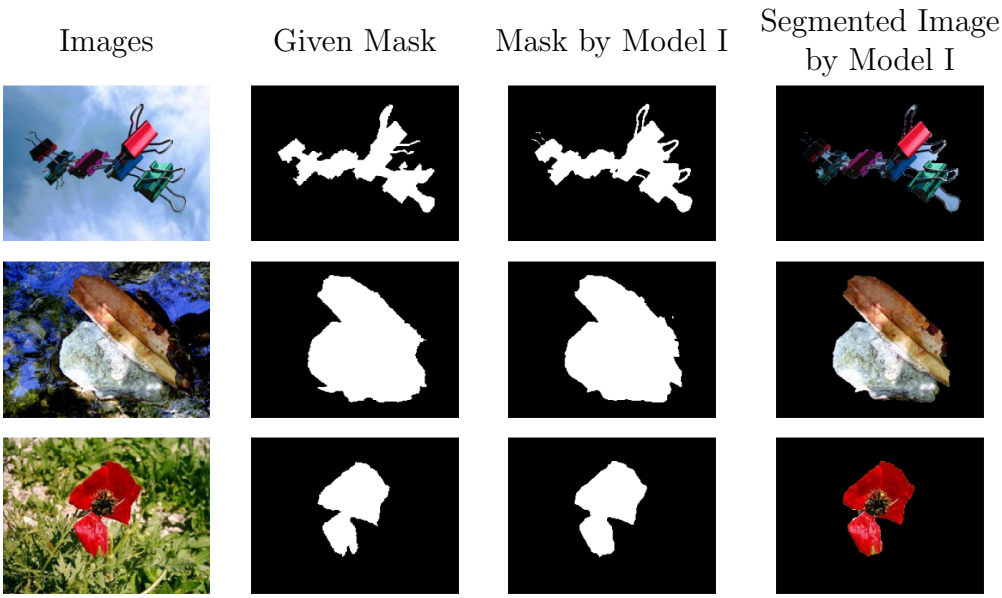

Figure 1: Some results by Model I.

6 Operator-splitting method inspired networks for image segmentation: Model II

As in [61], our second model incorporates (21) and the threshold dynamics ideas in approximating the perimeter of a region.

6.1 Model formulation

In (21), when v is binary, $\int_\Omega |\nabla v| d\mathbf{x}$ gives the perimeter of Ω_1, the support of v. Replacing $\int_\Omega |\nabla v| d\mathbf{x}$ by $\mathrm{Per}(\Omega_1)$ in (21), we get

$$\min_{v\in\{0,1\}} \int_\Omega F(f) v d\mathbf{x} + \lambda \mathrm{Per}(\Omega_1). \tag{31}$$

Then we use a threshold dynamics idea to approximate the perimeter term:

$$\mathrm{Per}(\Omega_1) \approx \sqrt{\frac{\pi}{\delta}} \int_\Omega v(\mathbf{x})(G_\delta * (1-v))(\mathbf{x}) d\mathbf{x}, \tag{32}$$

where G_δ is the two-dimensional Gaussian filter (the covariance matrix of $\mathbf{x}$ being an identity matrix)

$$G_\delta(\mathbf{x}) = \frac{1}{2\pi\delta^2} \exp\left(-\frac{\|\mathbf{x}\|^2}{2\delta^2}\right). \tag{33}$$

It is shown in [55] that the approximation in (32) Γ–converges to $\mathrm{Per}(\Omega_1)$ as $\delta \to 0$. Replacing $\mathrm{Per}(\Omega_1)$ by the approximation given in (32), the functional we are minimizing becomes

$$\min_{v\in\{0,1\}} \int_\Omega F(f) v d\mathbf{x} + \lambda \int_\Omega v(\mathbf{x})(G_\delta * (1-v))(\mathbf{x}) d\mathbf{x}. \tag{34}$$

Model (34) requires v to be binary. We relax this constraint to $v \in [0,1]$ and consider the following functional instead

$$\min_{v\in[0,1]} \int_\Omega F(f)v d\mathbf{x} + \varepsilon \int_\Omega v\ln v + (1-v)\ln(1-v) d\mathbf{x} + \lambda \int_\Omega v(\mathbf{x})(G_\delta * (1-v)(\mathbf{x}) d\mathbf{x} \quad (35)$$

for some small $\varepsilon > 0$. Such an approximate is a smoothed version of the binary constraint, and it converges to the original problem (34) as $\varepsilon \to 0$, c.f. [45].

If u is a minimizer of (35), it satisfies the optimality condition

$$\varepsilon \left(\ln \frac{u}{1-u} \right) + \lambda G_\delta * (1-2u) + F(f) = 0.$$

The gradient flow equation for this problem is:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \varepsilon \left(\ln \dfrac{u}{1-u} \right) + \lambda G_\delta * (1-2u) + F(f) = 0, \\ u(0) = u_0. \end{cases} \quad (36)$$

We then introduce control variables to (36) and solve

$$\begin{cases} \dfrac{\partial u}{\partial t} + \varepsilon \left(\ln \dfrac{u}{1-u} \right) + \lambda G_\delta * (1-2u) + F(f) + A(t) * u + g(t) = 0, \\ u(0) = u_0 = G(f) \end{cases} \quad (37)$$

for some initial condition $G(f)$ which is a function applied on f. Here we used a different notation from Section 5 to differentiate the different control variables.

Unlike Model I which treats $F(f)$ and $b(t)$ as two functions, we use a different strategy to treat Model II. Since the term $F(f)$ and g are just constant terms with respect to u, we can combine them and denote them by g which depends on f. The new problem is written as

$$\begin{cases} \dfrac{\partial u}{\partial t} + \varepsilon \left(\ln \dfrac{u}{1-u} \right) + \lambda G_\delta * (1-2u) + A(t) * u + g(f,t) = 0, \\ u(0) = u_0 = G(f). \end{cases} \quad (38)$$

6.2 An operator splitting method to solve (38)

We decompose the operator A and the function g as a sum of K terms for some positive integer K:

$$A = \sum_{k=1}^{K} A_k, \quad g = \sum_{k=1}^{K} g_k \quad (39)$$

for some operators A_k's and functions g_k's. We then use a Lie scheme to solve (38). Given u^n, we update $u^n \to \cdots \to u^{n+\frac{k}{K}} \to \cdots \to u^{n+1}$ via K substeps:
For $k = 1, ..., K-1$, solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + \varepsilon \left(\ln \dfrac{v}{1-v} \right) + A_k(t) * v + g_k(f,t) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{k-1}{K}}, \end{cases} \quad (40)$$

and set $u^{n+\frac{k}{K}} = v(t^{n+1})$.
For $k = K$, solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + \varepsilon \left(\ln \dfrac{v}{1-v} \right) + \lambda G_\delta * (1 - 2v) + A_K(t) * v + g_K(f,t) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{K-1}{K}}, \end{cases} \tag{41}$$

and set $u^{n+1} = v(t^{n+1})$.

For (40), we further apply a Lie scheme to decompose it into two substeps:
Substep 1: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + A_k(t) * v + g_k(f,t) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{k-1}{K}}, \end{cases} \tag{42}$$

and set $\bar{u}^{n+\frac{k}{K}} = v(t^{n+1})$.
Substep 2: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + \varepsilon \left(\ln \dfrac{v}{1-v} \right) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = \bar{u}^{n+\frac{k}{K}}, \end{cases} \tag{43}$$

and set $u^{n+\frac{k}{K}} = v(t^{n+1})$.

Similarly, we solve (41) by the following two substeps:
Substep 1: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + A_K(t) * v + g_K(f,t) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = u^{n+\frac{K-1}{K}}, \end{cases} \tag{44}$$

and set $\bar{u}^{n+1} = v(t^{n+1})$.
Substep 2: Solve

$$\begin{cases} \dfrac{\partial v}{\partial t} + \varepsilon \left(\ln \dfrac{v}{1-v} \right) + \lambda G_\delta * (1 - 2u) = 0 \text{ in } \Omega \times (t^n, t^{n+1}], \\ v(t^n) = \bar{u}^{n+1}, \end{cases} \tag{45}$$

and set $u^{n+1} = v(t^{n+1})$.

6.3 Time discretization

Problem (42)–(45) are only semi-constructive as we still need to solve these initial value problems. In our algorithm, we time-discretize (42) and (44) by the forward Euler method and discretize (43) and (45) by the backward Euler method. The discretized scheme is given as follows: For (40), we use the following time discretization to solve it

$$\frac{\bar{u}^{n+\frac{k}{K}} - u^{n+\frac{k-1}{K}}}{\Delta t} + A_k^n * u^{n+\frac{k-1}{K}} + g_k^n = 0, \tag{46}$$

$$\frac{u^{n+\frac{k}{K}} - \bar{u}^{n+\frac{k}{K}}}{\Delta t} + \varepsilon \left(\ln \frac{u^{n+\frac{k}{K}}}{1 - u^{n+\frac{k}{K}}} \right) = 0. \tag{47}$$

For (41), we use the following time discretization to solve it

$$\frac{\bar{u}^{n+1} - u^{n+\frac{K-1}{K}}}{\Delta t} + A_K^n * u^{n+\frac{K-1}{K}} + g_K^n = 0, \tag{48}$$

$$\frac{u^{n+1} - \bar{u}^{n+1}}{\Delta t} + \varepsilon \left(\ln \frac{u^{n+1}}{1 - u^{n+1}} \right) + \lambda G_\delta * (1 - 2u^{n+1}) = 0. \tag{49}$$

In our algorithm, we choose $G(f)$ as a convolution layer of f, i.e., a convolution of f with a 3×3 kernel followed by a sigmoid function. We will also choose g^n's as networks. Suppose we are given a set of images $\{f_i\}_{i=1}^I$ and the corresponding segmentation masks $\{h_i\}_{i=1}^I$. Denote the set of control variables by $\theta_2 = \{A^n, \theta_{g^n}, \theta_G\}_{n=1}^N$, where θ_{g^n} and θ_G denote the parameters in g^n and G, respectively. Denote the procedure of numerically solving (38) with N time steps and control variables θ_2 by

$$\mathcal{N}_2(\theta_2; \cdot) : f \to u^0 \to u^1 \to \cdots \to u^N.$$

We will learn θ_2 by solving

$$\min_{\theta_2} \frac{1}{I} \sum_{i=1}^{I} \ell(\mathcal{N}_2(\theta_2; f_i), h_i), \tag{50}$$

where $\ell(\cdot, \cdot)$ is some loss function, such as the cross entropy.

6.4 Connections to neural networks

The building block of scheme (40)–(41) is (46)–(49). Note that the first substep (46) (resp. (48)) is a linear step in $\bar{u}^{n+\frac{k}{K}}$ (resp. $\bar{u}^{n+1}$). The second step (47) (resp. (49)) is a nonlinear step. Thus this procedure is the same as a layer of a neural network, which consists a linear step and a nonlinear step (activation step). The numerical scheme for (40)–(41) that maps $f \to u^0 \to u^1 \to \cdots \to u^N$ is a neural network with KN layers and activation functions specified in (47) and (49). The process (50) of learning θ_2 is the same as training a network.

6.5 Numerical experiments

In this section, we demonstrate the robustness of Model II against noise. We show that one trained Model II can provide good segmentation results on images with various levels of noise. In our experiments, we choose $G(f)$ as a convolution layer of f. We set A_k^n's as learnable convolution kernels at different scales that extract various image features. We set g_k^n's as the convolution layers that are applied to f. For parameters, we use $\Delta t = 0.5, \varepsilon = 2, \lambda = 80$.

We use the MSRA10K data again while resizing all images to a size of 192×256. In our training, we train our model on images with noise standard deviation (SD) 1.0. This noise is big. Many existing algorithms cannot handle so big amount of noise. Some segmentation results are presented in Figure 2. For various levels of noise (even with very large noise), the trained model segments the target well. Refer to [61] for more experiments and explanations.

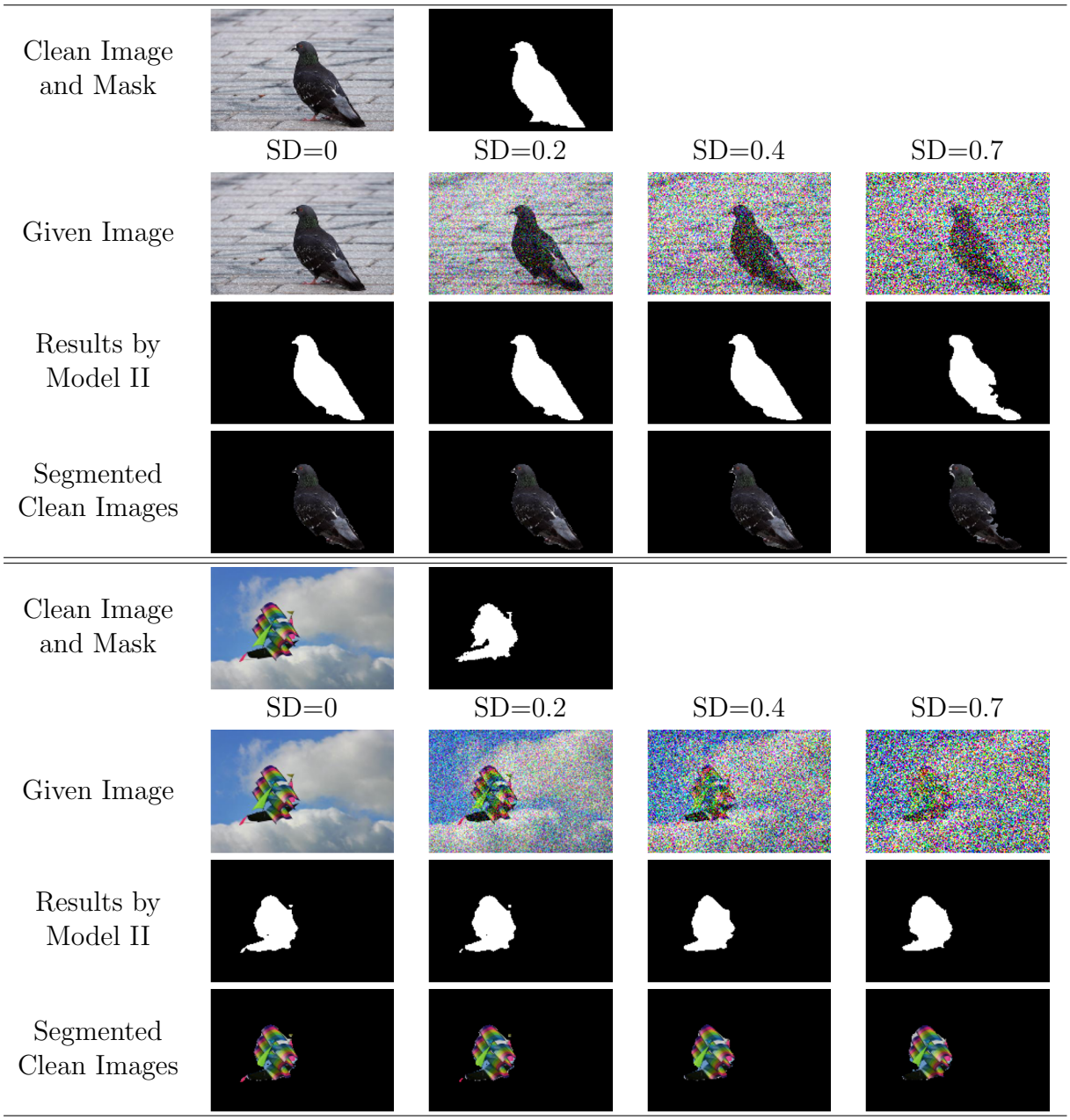

Figure 2: Results by Model II.

7 Conclusion

In this paper, the relations between deep neural networks and operator-splitting methods are discussed, and two new networks inspired by the operator-splitting method are proposed for image segmentation. The two proposed networks are derived from the Potts model, with certain terms having physical meanings as regularizers. Essentially, the two networks are operator-splitting algorithms solving the Potts model. The effectiveness of the proposed networks is demonstrated by numerical experiments.

References

[1] E. Bae, X.-C. Tai, and W. Zhu. Augmented Lagrangian method for an Euler's elastica based segmentation model that promotes convex contours. *Inverse Problems & Imaging*, 11(1), 2017.

[2] E. Bae, J. Yuan, and X.-C. Tai. Global minimization for continuous multiphase partitioning problems using a dual approach. *International Journal of Computer Vision*, 92(1):112–129, 2011.

[3] E. Bae, J. Yuan, X.-c. Tai, and Y. Boykov. A Fast Continuous Max-Flow Approach to Non-convex Multi-labeling Problems. In A. Bruhn, T. Pock, and X.-C. Tai, editors, *Efficient Algorithms for Global Optimization Methods in Computer Vision*, volume 8293 of *Lecture Notes in Computer Science*, pages 134–154, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[4] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

[5] K. Bhattacharya, B. Hosseini, N. B. Kovachki, and A. M. Stuart. Model reduction and neural networks for parametric PDEs. *The SMAI Journal of Computational Mathematics*, 7:121–157, 2021.

[6] A. Bonito, A. Caboussat, and M. Picasso. Operator splitting algorithms for free surface flows: Application to extrusion processes. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 677–729. Springer, 2017.

[7] Y. Boykov and G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[8] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[9] M. Bukač, S. Čanić, R. Glowinski, J. Tambača, and A. Quaini. Fluid–structure interaction in blood flow capturing non-zero longitudinal structure displacement. *Journal of Computational Physics*, 235:515–541, 2013.

[10] X. Cai, R. Chan, M. Nikolova, and T. Zeng. A three-stage approach for segmenting degraded color images: Smoothing, lifting and thresholding (SLaT). *Journal of Scientific Computing*, 72:1313–1332, 2017.

[11] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61, 1997.

[12] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[13] R. Chan, A. Lanza, S. Morigi, and F. Sgallari. Convex non-convex image segmentation. *Numerische Mathematik*, 138:635–680, 2018.

[14] R. Chan, H. Yang, and T. Zeng. A two-stage image segmentation method for blurry images with poisson or multiplicative Gamma noise. *SIAM Journal on Imaging Sciences*, 7(1):98–127, 2014.

[15] T. Chan and L. Vese. An active contour model without edges. In *Scale-Space Theories in Computer Vision: Second International Conference, Scale-Space'99 Corfu, Greece, September 26–27, 1999 Proceedings 2*, pages 141–151. Springer, 1999.

[16] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[17] M. Chen, H. Jiang, W. Liao, and T. Zhao. Efficient approximation of deep ReLU networks for functions on low dimensional manifolds. *Advances in Neural Information Processing Systems*, 32, 2019.

[18] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2014.

[19] A. J. Chorin, T. J. Hughes, M. F. McCracken, and J. E. Marsden. Product formulas and numerical algorithms. *Communications on Pure and Applied Mathematics*, 31(2):205–256, 1978.

[20] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[21] L.-J. Deng, R. Glowinski, and X.-C. Tai. A new operator splitting method for the Euler elastica model for image smoothing. *SIAM Journal on Imaging Sciences*, 12(2):1190–1230, 2019.

[22] Y. Duan, Q. Zhong, X.-C. Tai, and R. Glowinski. A fast operator-splitting method for Beltrami color image denoising. *Journal of Scientific Computing*, 92(3):89, 2022.

[23] T. Fan, G. Wang, Y. Li, and H. Wang. MA-Net: A multi-scale attention network for liver and tumor segmentation. *IEEE Access*, 8:179656–179665, 2020.

[24] R. Glowinski, S. Leung, H. Liu, and J. Qian. On the numerical solution of nonlinear eigenvalue problems for the Monge-Ampère operator. *ESAIM: Control, Optimisation and Calculus of Variations*, 26:118, 2020.

[25] R. Glowinski, S. Leung, and J. Qian. A penalization-regularization-operator splitting method for eikonal based traveltime tomography. *SIAM Journal on Imaging Sciences*, 8(2):1263–1292, 2015.

[26] R. Glowinski, H. Liu, S. Leung, and J. Qian. A finite element/operator-splitting method for the numerical solution of the two dimensional elliptic Monge–Ampère equation. *Journal of Scientific Computing*, 79:1–47, 2019.

[27] R. Glowinski, S. J. Osher, and W. Yin. *Splitting Methods in Communication, Imaging, Science, and Engineering*. Springer, 2017.

[28] R. Glowinski, T.-W. Pan, and X.-C. Tai. Some facts about operator-splitting and alternating direction methods. *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 19–94, 2016.

[29] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[30] Y. He, S. H. Kang, and H. Liu. Curvature regularized surface reconstruction from point clouds. *SIAM Journal on Imaging Sciences*, 13(4):1834–1859, 2020.

[31] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[32] Y. Lan, Z. Li, J. Sun, and Y. Xiang. DOSnet as a non-black-box pde solver: When deep learning meets operator splitting. *arXiv preprint arXiv:2212.05571*, 2022.

[33] S. Lanthaler, S. Mishra, and G. E. Karniadakis. Error estimates for DeepONets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1):tnac001, 2022.

[34] 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*, 1(4):541–551, 1989.

[35] S. Lie. *Theorie der Transformationsgruppen*, volume 2. American Soc., Providence, RI, 1970.

[36] H. Liu, M. Chen, S. Er, W. Liao, T. Zhang, and T. Zhao. Benefits of overparameterized convolutional residual networks: Function approximation under smoothness constraint. In *International Conference on Machine Learning*, pages 13669–13703. PMLR, 2022.

[37] H. Liu, M. Chen, T. Zhao, and W. Liao. Besov function approximation and binary classification on low-dimensional manifolds using convolutional residual networks. In *International Conference on Machine Learning*, pages 6770–6780. PMLR, 2021.

[38] H. Liu, A. Havrilla, R. Lai, and W. Liao. Deep nonparametric estimation of intrinsic data structures by chart autoencoders: Generalization error and robustness. *arXiv preprint arXiv:2303.09863*, 2023.

[39] H. Liu, J. Liu, R. Chan, and X.-C. Tai. Double-well Net for image segmentation. *In preparation.*

[40] H. Liu, X.-C. Tai, and R. Glowinski. An operator-splitting method for the Gaussian curvature regularization model with applications to surface smoothing and imaging. *SIAM Journal on Scientific Computing*, 44(2):A935–A963, 2022.

[41] H. Liu, X.-C. Tai, R. Kimmel, and R. Glowinski. A color elastica model for vector-valued image regularization. *SIAM Journal on Imaging Sciences*, 14(2):717–748, 2021.

[42] H. Liu, X.-C. Tai, R. Kimmel, and R. Glowinski. Elastica models for color image regularization. *SIAM Journal on Imaging Sciences*, 16(1):461–500, 2023.

[43] H. Liu and D. Wang. Fast operator splitting methods for obstacle problems. *Journal of Computational Physics*, page 111941, 2023.

[44] H. Liu, H. Yang, M. Chen, T. Zhao, and W. Liao. Deep nonparametric estimation of operators between infinite dimensional spaces. *arXiv preprint arXiv:2201.00217*, 2022.

[45] J. Liu, X. Wang, and X.-C. Tai. Deep convolutional neural networks with spatial regularization, volume and star-shape priors for image segmentation. *Journal of Mathematical Imaging and Vision*, 64(6):625–645, 2022.

[46] J. Lu, Z. Shen, H. Yang, and S. Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021.

[47] T. Lu, P. Neittaanmaki, and X.-C. Tai. A parallel splitting-up method for partial differential equations and its applications to Navier-Stokes equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 26(6):673–708, 1992.

[48] S. MacNamara and G. Strang. Operator splitting. *Splitting methods in Communication, Imaging, Science, and Engineering*, pages 95–114, 2016.

[49] G. I. Marchuk. Splitting and alternating direction methods. *Handbook of Numerical Analysis*, 1:197–462, 1990.

[50] L. Modica. Un esempio di γ-convergenza. *Boll. Un. Mat. Ital. B*, 14:285–299, 1977.

[51] L. Modica. The gradient theory of phase transitions and the minimal interface criterion. *Archive for Rational Mechanics and Analysis*, 98:123–142, 1987.

[52] A. Mrad, A. Caboussat, and M. Picasso. A splitting method for the numerical simulation of free surface flows with sediment deposition and resuspension. *International Journal for Numerical Methods in Fluids*, 94(10):1724–1743, 2022.

[53] D. B. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 1989.

[54] K. Oono and T. Suzuki. Approximation and non-parametric estimation of resnet-type convolutional neural networks. In *International Conference on Machine Learning*, pages 4922–4931. PMLR, 2019.

[55] M. M. J.-D. Pallara-F and P.-M. Preunkert. Short-time heat flow and functions of bounded variation in R^N. *Journal: Ann. Fac. Sci. Toulouse Math*, 6:16, 2007.

[56] R. B. Potts. Some generalized order-disorder transformations. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 48, pages 106–109. Cambridge University Press, 1952.

[57] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 234–241. Springer, 2015.

[58] G. Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

[59] H. Sun, X.-C. Tai, and J. Yuan. Efficient and convergent preconditioned ADMM for the Potts models. *SIAM Journal on Scientific Computing*, 43(2):B455–B478, 2021.

[60] X. Tai, L. Li, and E. Bae. The Potts model with different piecewise constant representations and fast algorithms: a survey. *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging: Mathematical Imaging and Vision*, pages 1–41, 2021.

[61] X.-C. Tai, H. Liu, and R. Chan. PottsMGNet: A mathematical explanation of encoder-decoder based neural networks. *arXiv preprint arXiv:2307.09039*, 2023.

[62] D. Wang, H. Li, X. Wei, and X.-P. Wang. An efficient iterative thresholding method for image segmentation. *Journal of Computational Physics*, 350:657–667, 2017.

[63] Y. Wang, X. Xiao, and X. Feng. An efficient maximum bound principle preserving p-adaptive operator-splitting method for three-dimensional phase field shape transformation model. *Computers & Mathematics with Applications*, 120:78–91, 2022.

[64] K. Wei, X.-C. Tai, T. F. Chan, and S. Leung. Primal-dual method for continuous max-flow approaches. In *Computational Vision and Medical Image Processing V: Proceedings of the 5th Eccomas Thematic Conference on Computational Vision and Medical Image Processing (VipIMAGE 2015, Tenerife, Spain, October 19-21, 2015)*, page 17. CRC Press, 2015.

[65] D. Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

[66] M. Yashtini and S. H. Kang. A fast relaxed normal two split method and an effective weighted TV approach for Euler's elastica image inpainting. *SIAM Journal on Imaging Sciences*, 9(4):1552–1581, 2016.

[67] J. Yuan, E. Bae, and X.-C. Tai. A study on continuous max-flow and min-cut approaches. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2217–2224. IEEE, 2010.

[68] J. Yuan, E. Bae, X.-C. Tai, and Y. Boykov. A continuous max-flow approach to Potts model. In *European Conference on Computer Vision*, pages 379–392. Springer, 2010.

[69] D.-X. Zhou. Theory of deep convolutional neural networks: Downsampling. *Neural Networks*, 124:319–327, 2020.

[70] D.-X. Zhou. Universality of deep convolutional neural networks. *Applied and Computational Harmonic Analysis*, 48(2):787–794, 2020.

[71] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: A nested U-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 3–11. Springer, 2018.