RESEARCH ARTICLE | JUNE 28 2022

Classification of ternary data using the ternary Allen–Cahn system for small datasets

Donghun Lee; Sangkwon Kim; Hyun Geun Lee; Soobin Kwak; Jian Wang ; Junseok Kim

AIP Advances 12, 065324 (2022)
https://doi.org/10.1063/5.0094551

Classification of ternary data using the ternary Allen–Cahn system for small datasets

Cite as: AIP Advances 12, 065324 (2022); doi: 10.1063/5.0094551
Submitted: 4 April 2022 • Accepted: 7 June 2022 •
Published Online: 28 June 2022

Donghun Lee,[1] Sangkwon Kim,[1] Hyun Geun Lee,[2] Soobin Kwak,[1] Jian Wang,[3] and Junseok Kim[1,a)]

AFFILIATIONS

[1] Department of Mathematics, Korea University, Seoul 02841, Republic of Korea
[2] Department of Mathematics, Kwangwoon University, Seoul 01897, Republic of Korea
[3] School of Mathematics and Statistics, Nanjing University of Information Science and Technology, Nanjing 210044, China

[a)] **Author to whom correspondence should be addressed:** cfdkim@korea.ac.kr

ABSTRACT

In this study, we present a classification method for ternary small data using the modified ternary Allen–Cahn (tAC) system. The governing system is the tAC equation with the fidelity term, which keeps the solution as close as possible to the given data. To solve the tAC system with the fidelity term, we apply an operator splitting method. We use an implicit-explicit finite difference method for solving the split equations. To validate the robust and superior performance of the proposed numerical algorithm, we perform the comparison tests with other widely used classifiers such as logistic regression, decision tree, support vector machine, random forest, and artificial neural network for small datasets.

I. INTRODUCTION

Recently, interest in automatic classification problems has significantly grown. In practical applications, machine learning based multi-classification algorithms have been extended to various classification problems involving data, text, image, mobile traffic, and network traffic.[1–6] Among the multi-classification systems, the most prevalent algorithms are machine learning methods such as support vector machine (SVM), random forest (RF), artificial neural network (ANN), and logistic regression (LR). SVM was originally designed to solve classification problems of two categories. This technology has successful results in pattern recognition, signal analysis, and classification of images. Suykens and Vandewalle[7] proposed least squares SVM (LS-SVM), and it has been applied to solve specific integral equations.[8] However, SVM cannot be directly adopted to solve multi-classification problems. There are two main methods to generalize SVM to solve multi-classification problems, including one-versus-rest SVMs (OVR-SVMs) and one-versus-one SVMs (OVO-SVMs). The training time of the OVR-SVMs is directly proportional to the number of categories, which makes the training difficult as the training sample increases.[9] In addition, with a large amount of the categories, OVO-SVMs break down the multiclass problem into multiple binary classification problems, which means a hyperplane is needed to separate between every two classes. Therefore, this will lead to the following disadvantages of the OVO-SVMs approach. One is that the number of classifiers increases as the number of classes. In addition, the training of each classifier only considers the data from two classes; thus, the information from all remaining classes is ignored.[10] Therefore, some optimization models based on SVM have been proposed to solve the multi-classification problem. Khemchandani and Saigal[11] proposed the ternary decision structure based multi-category twin support vector machines classifier, which is more efficient in handling multi-class data. Moreover, Chang *et al.*[12] implemented multi-class smooth SVM for a ternary classification problem, TSSVM.

Compared with traditional machine learning algorithms, the learning ability of neural networks has greatly promoted the development of artificial intelligence in a more intelligent and efficient direction. Thus, numerous neural network algorithms have been proposed recently.[13,14] For ternary classification, a convolutional neural network (CNN) based steganalytic method was presented for image steganalysis.[15] In addition, the algorithm based on neural networks is also widely used in the field of medical image classification. Using 3D CNN features and multilayer perceptron, Raju *et al.*[16] studied the multi-classification of Alzheimer's disease. In another field, multilabel emotion classification of informal text was analyzed by a multi-channel BiLSTM-CNN model.[17] In addition, a deep

learning based ternary task classification system was proposed by using the Gramian angular summation field in fNIRS neuroimaging data.[18] Other machine learning algorithms, such as random forest (RF), and logistic regression (LR), are also widely employed in multi-classification problems.[19–22]

In this study, we propose a novel and robust ternary data classification algorithm using the Allen–Cahn (AC)[23] model with a fidelity term for small datasets. Traditional machine learning usually requires large training datasets, which directly affects the performance of small datasets. To overcome this problem, various methods are being studied[24,25] and we also propose the algorithm. The proposed AC classification system, tAC, can effectively solve the problem when the training data are not enough. We perform various experimental results to show that the proposed classification algorithm has a high recognition rate in small datasets, and comparison tests with other machine learning algorithms are also presented.

This paper is organized as follows: We describe the tAC equation with the fidelity term in Sec. II. Section III explains the numerical scheme to solve the governing equation. Section IV presents the results of classification for synthetic datasets and performance comparison with other classification methods. Conclusions are given in Sec. V.

II. TERNARY AC EQUATION WITH THE FIDELITY TERM

We assume that total energy functional $\mathscr{E}$ can be written as the following equation:

$$\mathscr{E}(\mathbf{c}(\mathbf{x},t)) = \int_\Omega \sum_{k=1}^{3} \left[\frac{F(c_k(\mathbf{x},t))}{\varepsilon^2} + \frac{1}{2}|\nabla c_k(\mathbf{x},t)|^2 + \frac{\lambda}{2}(c_k(\mathbf{x},t) - f_k(\mathbf{x}))^2 \right] d\mathbf{x}, \tag{1}$$

where $\mathbf{x} = (x, y)$, $F(c_k) = 0.25c_k^2(c_k - 1)^2$, λ is positive constant, and a fidelity $f_k(\mathbf{x})$ is a preprocessed data, which satisfies the constraint $f_1(\mathbf{x}) + f_2(\mathbf{x}) + f_3(\mathbf{x}) = 1$. The L^2-gradient flow for Eq. (1) is the modified tAC equation. For $k = 1, 2, 3$, we have the equations to be

$$\frac{\partial c_k(\mathbf{x},t)}{\partial t} = -\frac{F'(c_k(\mathbf{x},t))}{\varepsilon^2} + \Delta c_k(\mathbf{x},t) - \lambda(c_k(\mathbf{x},t) - f_k(\mathbf{x})) - \beta(\mathbf{c}(\mathbf{x},t)) \quad \text{for } \mathbf{x} \in \Omega,\ t > 0, \tag{2}$$

where $\beta(\mathbf{c}(\mathbf{x},t)) = -\sum_{k=1}^{3} F'(c_k(\mathbf{x},t))/(3\varepsilon^2)$. Here, we used the constraint $c_1(\mathbf{x},t) + c_2(\mathbf{x},t) + c_3(\mathbf{x},t) = 1$. On the domain boundary, we apply the zero Neumann boundary condition $\mathbf{n} \cdot \nabla c_k(\mathbf{x},t) = 0$, for $\mathbf{x} \in \partial\Omega, t > 0$, where $\mathbf{n}$ is the unit normal vector to the domain boundary. The detailed derivation of Eq. (2) is provided in Appendix.

III. NUMERICAL SCHEME

Now, a computation method for the modified tAC system is described in computational domain $\Omega = (L_x, R_x) \times (L_y, R_y)$, which is discretized as $\Omega_h = \{(x_i, y_j) | x_i = L_x + ih, y_j = L_y + jh, 0 \le i \le N_x, 0 \le j \le N_y\}$, where N_x and N_y are integers; and $h = (R_x - L_x)/N_x$. Let $c_{k,ij}^n$ be numerical approximations of $c_k(x_i, y_j, n\Delta t)$ with time step Δt. The modified tAC equation can be split into the following two equations by the operator splitting method:

$$\frac{\partial c_k(\mathbf{x},t)}{\partial t} = -\frac{F'(c_k(\mathbf{x},t))}{\varepsilon^2} + \Delta c_k(\mathbf{x},t) - \beta(\mathbf{c}), \tag{3}$$

$$\frac{\partial c_k(\mathbf{x},t)}{\partial t} = \lambda[f_k(\mathbf{x}) - c_k(\mathbf{x},t)]. \tag{4}$$

For $0 < i < N_x, 0 < j < N_y$, we use the explicit Euler method to solve Eq. (3),

$$\frac{c_{k,ij}^{n+\frac{1}{2}} - c_{k,ij}^n}{\Delta t} = -\frac{F'(c_{k,ij}^n)}{\varepsilon^2} + \Delta_h c_{k,ij}^n + \sum_{k=1}^{3} \frac{F'(c_{k,ij}^n)}{3\varepsilon^2},$$

where $\Delta_h c_{k,ij}^n = (c_{k,i-1,j}^n + c_{k,i+1,j}^n + c_{k,i,j-1}^n + c_{k,i,j+1}^n - 4c_{k,ij}^n)/h^2$. Here, we use the linear boundary condition: $c_{k,0j} = 2c_{k,1j} - c_{k,2j}$, $c_{k,i0} = 2c_{k,i1} - c_{k,i2}$, $c_{k,N_xj} = 2c_{k,N_x-1,j} - c_{k,N_x-2,j}$, $c_{k,iN_y} = 2c_{k,i,N_y-1} - c_{k,i,N_y-2}$. Next, we solve Eq. (4) for $c_{k,ij}^{n+\frac{1}{2}}$,

$$\frac{c_{k,ij}^{n+1} - c_{k,ij}^{n+\frac{1}{2}}}{\Delta t} = \lambda(f_{k,ij} - c_{k,ij}^{n+1}). \tag{5}$$

Rewriting Eq. (5), we have

$$c_{k,ij}^{n+1} = \frac{c_{k,ij}^{n+\frac{1}{2}} + \lambda\Delta t f_{k,ij}}{1 + \lambda\Delta t}.$$

Because this scheme is explicit, there is a mild constraint for the time step size, $\Delta t < 0.25h^2$,[26] for the stability of the scheme.

IV. COMPUTATIONAL EXPERIMENTS

We present the computational test results for two synthetic datasets: three moons and three spirals in 2*D*. We consider the domain $\Omega = (0,1) \times (0,1)$, a time step $\Delta t = 0.12h^2$, a grid size $h = 1/100$, and $\varepsilon = \varepsilon_m = hm/(4\sqrt{2}\tanh^{-1}(0.9))$.[27] We propose a preprocessing process for the given data to construct the fidelity term in the modified tAC system. Because we solve Eq. (3) using the lattice-based numerical method, we construct the fidelity term by distributing the information of the given data scattered across the computational domain Ω to adjacent grid points. Herein, each data information was distributed to adjacent grid points using bilinear

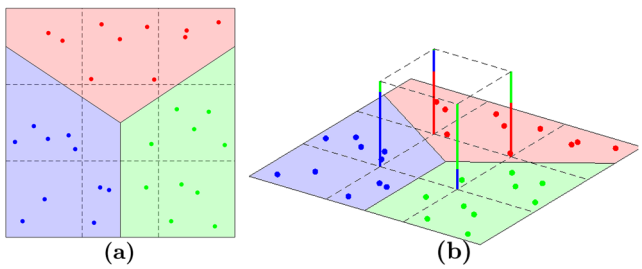

FIG. 1. Schematic illustration of defining a fidelity term. The color of each point means its class, and the length of the color bar is proportional to the size of the normalized allocation information value of each grid. (a) and (b) are the linearly weighted distribution results in 2d and 3d, respectively.

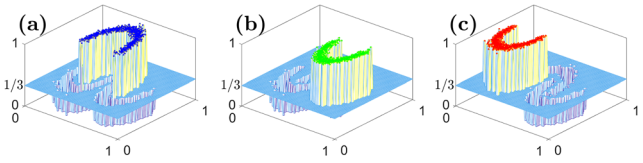

FIG. 2. Fidelity of three moons dataset: (a) c_1, (b) c_2, and (c) c_3.

weighted interpolation. Figure 1 shows the distribution process for adjacent grid points at each point, where the data classes are shown by coloring. As shown in Fig. 1(b), the information values of each class at the four grid points of the intermediate cell can be expressed as a unit length since the sum of the information values of the three classes is normalized to 1.

In addition, when multiple information of the same class is assigned to one grid point, the largest value is taken for that grid point. Finally, if there is no value assigned to a grid point, 1/3 is assigned. Figure 2 shows the fidelity term allocated information for the three moons dataset, where each class is shown in different colors.

A. Synthetic datasets

The modified tAC system can be applied to classifying the three class data as three moon (TM) data and three spiral (TS) data with numerical parameters: $\varepsilon = \varepsilon_4$ and $\lambda = 1000$.

1. Example 1: Three moons

TM dataset has three half circles. Each half circle consists of N data. In other words, there are $3N$ points in the dataset. The center of the upper semicircle is origin, and its have their centers radius is 1.2 of the two lower semicircles that are $(-1.3, 0.5)$ and $(1.3, 0.5)$ with radius 1. Then, Gaussian noise with mean 0 and standard deviation 0.1 is added to all points. The dataset can be generated as follows:

$$\begin{aligned}
\mathbb{D}_1 &= \{(x,y)|\ x = 1.2\cos(\theta) + z_1,\ y = 1.2\sin(\theta) + z_2\},\\
\mathbb{D}_2 &= \{(x,y)|\ x = 1.3 + \cos(\theta) + z_1,\ y = 0.5 - \sin(\theta) + z_2\},\\
\mathbb{D}_3 &= \{(x,y)|\ x = -1.3 + \cos(\theta) + z_1,\ y = 0.5 - \sin(\theta) + z_2\},
\end{aligned} \tag{6}$$

where $0 < \theta < \pi$ and Gaussian noise $z_1, z_2 \sim N(0, 0.1)$. Now, we consider the generated dataset $\mathbb{D} = \mathbb{D}_1 \cup \mathbb{D}_2 \cup \mathbb{D}_3$. Figure 3 shows the data generated by Eq. (6) and the number of data of each class $N = 1000$.

The color of each point means a different class. Then, we add margins and scale the axes to fit the model domain Ω. The size of each margin is proportional to the length of each axis of the generated data as shown in Fig. 4(a). Through data scaling, we have to make the range of all features the same. Specifically, the preprocessing to fit the computational domain $\Omega = (0,1) \times (0,1)$ can be written as

$$\begin{aligned}
\tilde{X} &= A(X - X_0),\\
\tilde{X} &= (\tilde{x}, \tilde{y})^T,\ X = (x,y)^T,\\
X_0 &= (-rx_{\max} + (1+r)x_{\min}, -ry_{\max} + (1+r)y_{\min})^T,\\
A &= \begin{pmatrix} 1/(1+2r)(x_{\max} - x_{\min}) & 0 \\ 0 & 1/(1+2r)(y_{\max} - y_{\min}) \end{pmatrix},
\end{aligned} \tag{7}$$

where a margin ratio $r = 0.1$. The goal is to get a decision boundary that classifies the learning data presented in Fig. 4(b) using the proposed algorithm. Figure 4(d) shows the decision curve and the classification results of the test data presented in Fig. 4(c). Here, the black points are the classification failed data.

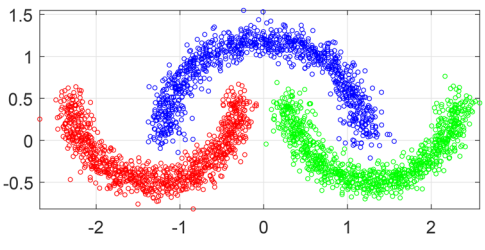

FIG. 3. Example 1: TM dataset with $N = 1000$.

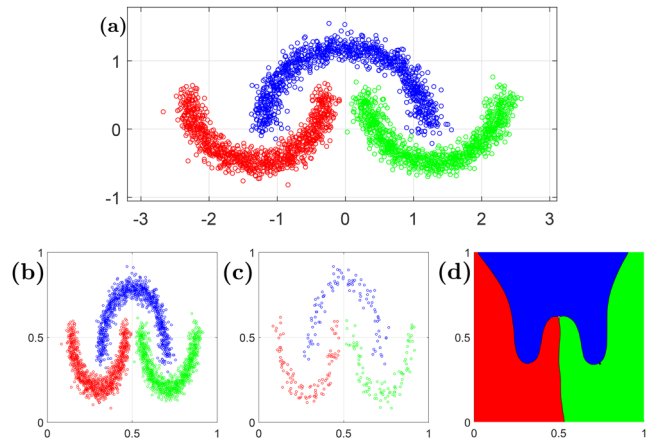

FIG. 4. Example 1: TM dataset. The color of each point means a different class. (a) is the result of adding margins on the raw dataset shown in Fig. 3, (b) is the normalized dataset showing the train data with $N_{train} = 1000$, and (c) is test data with $N_{test} = 100$. (d) shows classification results on the test data, where the black dots are points that fail to classify correctly.

2. Example 2: Three spiral

TS dataset has three Archimedean spiral structures with $N = 1000$, as shown in Fig. 5.

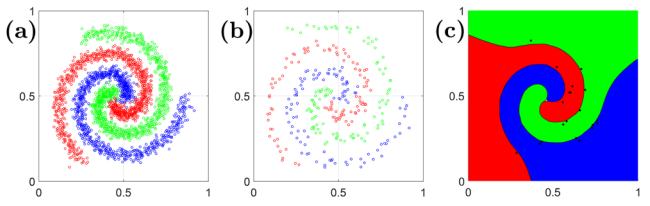

FIG. 5. example 2: TS data. The color of each point means a different class. (a) is train data with $N_{train} = 1000$, (b) is test data with $N_{test} = 100$, and (c) the result for classifying test data and the black dots are data that fail to classify.

The train and test datasets are generated by spiral equation as

$$\begin{aligned}\mathbb{D}_1 &= \{(x,y)|\ x = (a+b\theta)\cos(\theta) + z_1, y = (a+b\theta)\sin(\theta)) + z_2\},\\ \mathbb{D}_2 &= \{(x,y)|\ x = (a+b\theta)\cos(\theta + 2\pi/3) + z_1,\\ &\qquad y = (a+b\theta)\sin(\theta + 2\pi/3)) + z_2\},\\ \mathbb{D}_3 &= \{(x,y)|\ x = (a+b\theta)\cos(\theta + 4\pi/3) + z_1,\\ &\qquad y = (a+b\theta)\sin(\theta + 4\pi/3)) + z_2\}.\end{aligned} \tag{8}$$

where $a = 0.2$ and $b = 0.3$, and $0 < \theta < 2\pi$ and Gaussian noise $z_1, z_2 \sim N(0, 0.1)$.

Then, we add margins and normalize the dataset to fit the computational domain $\Omega = (0,1) \times (0,1)$ by Eq. (7) with $r = 0.1$. We get the TS data as shown in Fig. 5(a). We apply the proposed algorithm to get a decision boundary that classifies the data for Fig. 5(a). Figure 5(c) shows the decision curve and the classification results of the test data presented in Fig. 5(b). Here, the black points are the classification of failed data.

B. Comparison with other classifiers

In this subsection, the proposed algorithm is compared with classifiers that are widely used to verify the performance. We consider various classifiers: artificial neural network (ANN), random forest (RF), decision tree (DT), support vector machine (SVM), and logistic regression (LR). The parameters of the four classification algorithms (as LR: lambda, SVM: gamma and cost, DT and RF: min split, min bucket, max depth, and complexity) applied were determined as best set using a grid search method. ANN structure consisted of 8 layers and 30 nodes of each layer, and the activation functions are ReLU and softmax, and the loss function and optimizer are cross-entropy and Adam, respectively. The rest of the unmentioned parameters uses the default setting in the scikit-learn and TensorFlow package. To measure the performance of the proposed algorithm and different classifiers, we use commonly used accuracy (ACC), the area under the accuracy of the receiver operating characteristic curve (AUC), and the area under the precision–recall curve (AUPR), and F1-score. Four effective measures are based on TP, FP, TN, and FN, which denote true positive, false positive, true negative, and false negative, respectively. ACC was defined by the ratio of the samples correctly classified to the total samples, i.e., $ACC = (TP + TN)/(TP + FP + TN + FN)$. AUC represents the trade-off between the true positive rate (specificity) and the false positive rate (1-sensitivity), where specificity and sensitivity are defined as $TN/(FP + TN)$ and $TP/(TP + FN)$, respectively. AUPR represents the trade-off between the precision and the recall where precision and recall are defined as $TP/(TP + FP)$ and $TP/(TP + FN)$. Finally, the F1-score is known as a harmonic average of precision and recall and defined as $2(\text{precision} \times \text{recall})/(\text{precision} + \text{recall}) = 2TP/(2TP + FP + FN)$. The generalization performance results were reported by ACC, AUC, AUPR, and F1-score, which were measured on test data. All the performances of the classifier algorithms are measured on a computer with an Intel(R) Xeon(R) CPU @ 2.30 GHz. The test data of all tests used the data presented in Figs. 4(c) and 5(b)

We conducted numerical experiments on large datasets and small dataset of the learning dataset size. The classification performance of the proposed algorithm and different algorithms for $N_{train} = 1000$ and $N_{train} = 20$ is presented in Tables I and II, respectively. It can be observed from Table I that not only the proposed algorithm, modified tAC, but also all other algorithms show high accuracy for TM dataset of a relatively simple structure. Although the performance of all algorithms for TS data is lower than in TM data, the tAC, SVM, and ANN show high accuracy of 0.9 or more. Usually, when there is less learning data, the performance of the algorithm is lower than when the learning data are abundant. However, from Table II, it can be seen that there is little change in the performance of the proposed algorithm. Therefore, the proposed algorithm is robust and accurate better than the different algorithms for small datasets. For TM data, when $N_{train} = 1000$, the CPU times(in seconds) for ANN and tAC are 158.93 and 2.37, respectively, and for TS data, 46.30 and 2.41, respectively. For TM data, when $N_{train} = 20$, the CPU times(in seconds) for ANN and tAC

TABLE I. Performance of classification on TM and TS dataset with $N_{train} = 1000$ and $N_{test} = 100$.

		tAC	LR	SVM	DT	RF	ANN
TM	ACC	1.0	0.85	1.0	1.0	0.99	1.0
	AUC	1.0	0.97	1.0	1.0	1.0	1.0
	AUPR	1.0	0.95	1.0	1.0	1.0	1.0
	F1-score	1.0	0.85	1.0	1.0	0.99	1.0
TS	ACC	0.93	0.33	0.95	0.87	0.91	0.94
	AUC	0.98	0.61	1.0	0.90	0.97	1.0
	AUPR	0.97	0.50	1.0	0.81	0.94	0.98
	F1-score	0.93	0.33	0.95	0.87	0.91	0.94

TABLE II. Performance of classification on TM and TS datasets with $N_{train} = 20$ and $N_{test} = 100$.

		tAC	LR	SVM	DT	RF	ANN
TM	ACC	0.99	0.86	0.95	0.87	0.92	0.94
	AUC	1.0	0.97	1.0	0.91	0.99	1.0
	AUPR	1.0	0.95	1.0	0.81	0.98	0.99
	F1-score	0.99	0.85	0.95	0.86	0.92	0.94
TS	ACC	0.92	0.35	0.91	0.87	0.89	0.67
	AUC	0.98	0.60	0.97	0.90	0.97	0.86
	AUPR	0.97	0.53	0.95	0.80	0.95	0.77
	F1-score	0.92	0.35	0.91	0.87	0.89	0.67

TABLE III. Performance drop due to smaller training dataset.

		tAC	LR	SVM	DT	RF	ANN
TM	ACC	0.01	0.01	0.05	0.13	0.07	0.06
	AUC	0.00	0.00	0.00	0.09	0.01	0.00
	AUPR	0.00	0.00	0.00	0.19	0.02	0.01
	F1-score	0.01	0.00	0.05	0.14	0.07	0.06
TS	ACC	0.01	0.02	0.04	0.00	0.02	0.27
	AUC	0.00	0.01	0.03	0.00	0.00	0.14
	AUPR	0.00	0.03	0.05	0.01	0.01	0.21
	F1-score	0.01	0.02	0.04	0.00	0.02	0.27

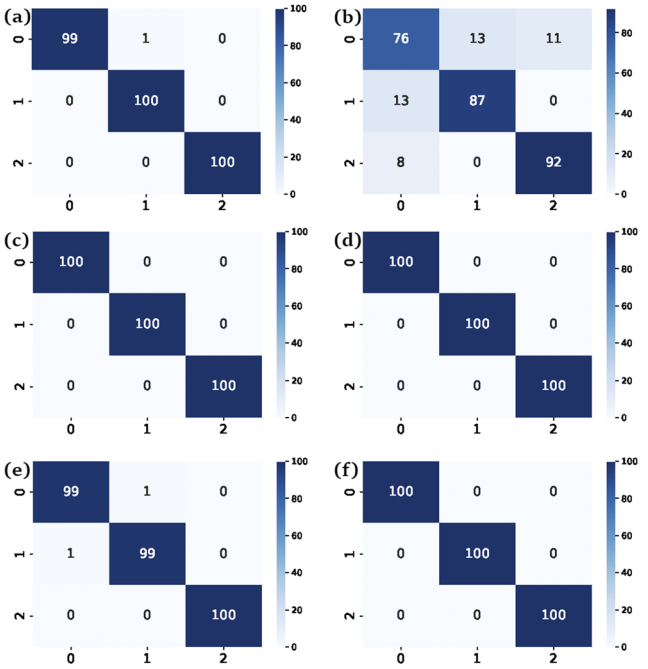

FIG. 6. Confusion matrices for TM dataset with N_{train} = 1000 and N_{test} = 100: (a) tAC, (b) LR, (c) SVM, (d) DT, (e) RF, and (f) ANN.

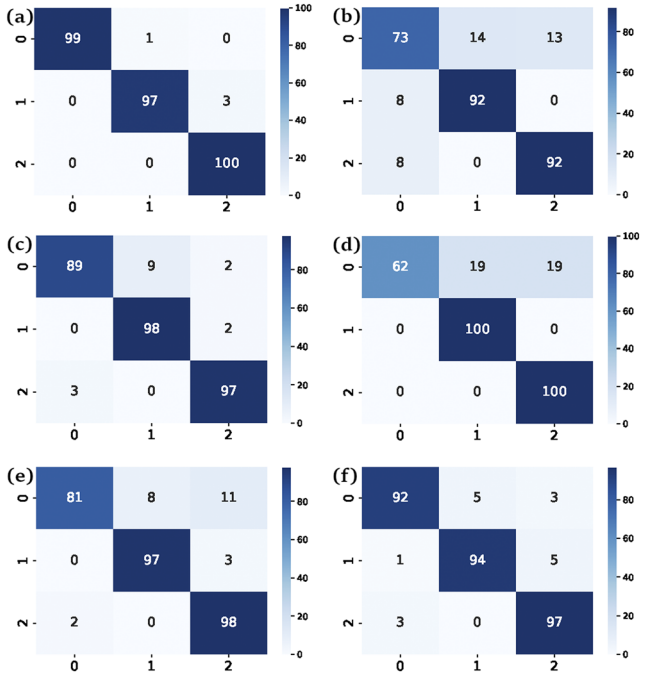

FIG. 8. Confusion matrices for the TM dataset with N_{train} = 20 and N_{test} = 100: (a) tAC, (b) LR, (c) SVM, (d) DT, (e) RF, and (f) ANN.

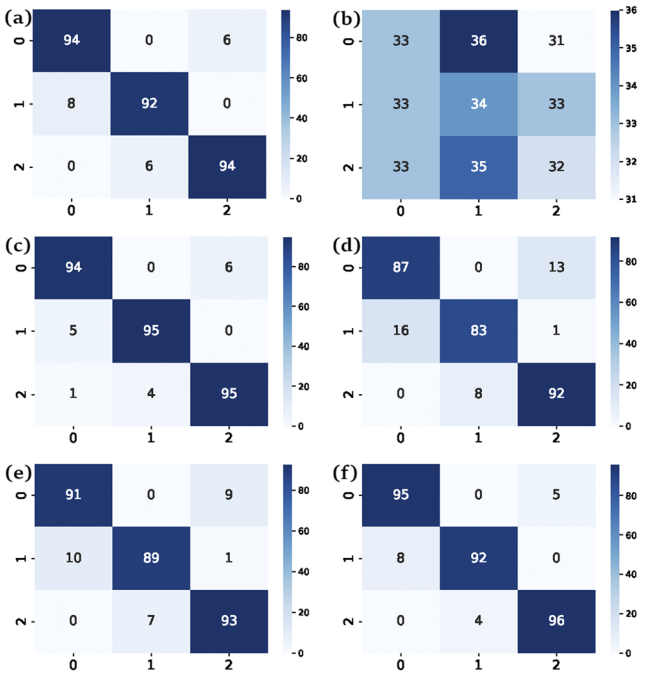

FIG. 7. Confusion matrices for TS dataset with N_{train} = 1000 and N_{test} = 100: (a) tAC, (b) LR, (c) SVM, (d) DT, € RF, and (f) ANN.

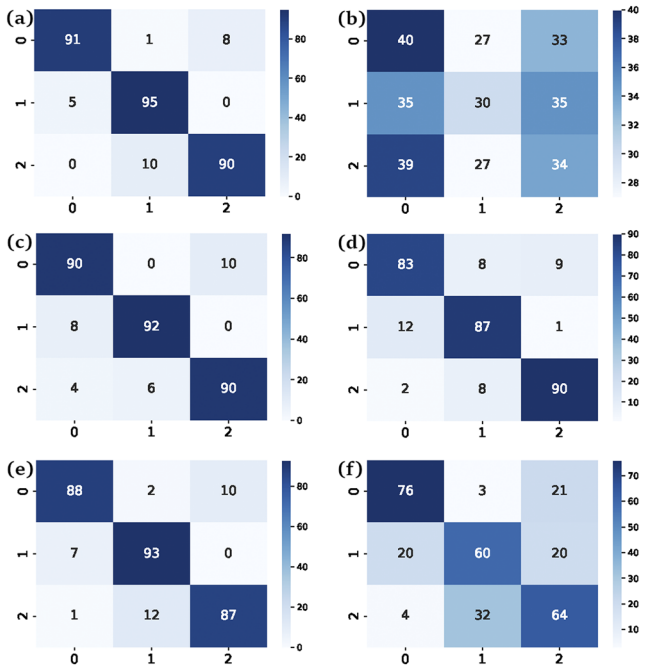

FIG. 9. Confusion matrices for the TS dataset with N_{train} = 20 and N_{test} = 100: (a) tAC, (b) LR, (c) SVM, (d) DT, (e) RF, and (f) ANN.

are 86.59 and 2.34, respectively, and for TS data, 142.52 and 2.37, respectively.

Table III shows that absolute difference between all the values in Tables I and II. Classifiers other than tAC are more affected by the drop in performance results caused by smaller training datasets, which suggests that tAC is more robust in learning from small datasets than other classifiers we tested.

Figures 6 and 7 present confusion matrices from all the algorithms for TM and TS datasets with N_{train} = 1000, respectively.

Figures 8 and 9 presented confusion matrices obtained using all the algorithms for TM and TS datasets with N_{train} = 20, respectively.

V. CONCLUSIONS

In this study, we presented a novel classification method for ternary data using the tAC system with a fidelity term for small datasets. To solve the tAC system with the fidelity term, we used the operator splitting method with an implicit-explicit FDM for solving the split equations. To validate the robust and superior performance of the proposed AC classification system, we performed the comparison tests with other widely used classifiers such as LR, SVM, DT, RF, and ANN. Here, the hyperparameters of each model are determined as the best set using the grid search method. In general, if there is not enough learning data, the performance of the model is poor. However, the proposed algorithm achieves high-accuracy performance on small datasets, which can be confirmed by the results. The number of features of input data is mostly two or more. As the number of features increases, the proposed AC classification system increases in dimension, resulting in high costs. This is called the curse of dimensionality. Therefore, for future work, we will overcome the curse of dimensionality by applying dimensional reduction methods such as PCA[28–30] or another method[31] to the data preprocessing process and expand the modified tAC to classify multi-class datasets (4 or more). In addition, introducing a dynamically reconstructed grid structure that balances the classification performance and the required computational burden of fidelity terms would result in a computationally more efficient method.

ACKNOWLEDGMENTS

S. K. Kim was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (Grant No. 2022R1C1C2005275). J. Wang sincerely thanks the Nanjing preferential subsidy project. This work was supported by the Brain Korea 21 FOUR (BK 21 FOUR) from the Ministry of Education of Korea. The authors appreciate the reviewers for their helpful comments on the revision of this paper.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Donghun Lee: Validation (equal); Writing – original draft (equal); Writing – review & editing (equal). **Sangkwon Kim:** Conceptualization (equal); Data curation (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Hyun Geun Lee:** Formal analysis (equal); Methodology (equal); Validation (equal); Writing – original draft (equal); Writing – review & editing (equal). **Soobin Kwak:** Investigation (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Jian Wang:** Investigation (equal); Writing – review & editing (equal). **Junseok Kim:** Conceptualization (equal); Formal analysis (equal); Methodology (equal); Project administration (equal); Supervision (equal); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

APPENDIX: DERIVATION OF THE GOVERNING EQUATION

To derive Eq. (2), let us consider a gradient flow such as

$$\frac{\partial \mathbf{c}(\mathbf{x},t)}{\partial t} = -\operatorname{grad} \mathscr{E}(\mathbf{c})(\mathbf{x},t), \tag{A1}$$

where $\operatorname{grad} \mathscr{E}(\mathbf{c})$ is computed as follows. Let $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), g_3(\mathbf{x}))$ be a smooth vector-valued function satisfying $\sum_{k=1}^{3} g_k(\mathbf{x}) = 0$,

$$\begin{aligned}
(\operatorname{grad} \mathscr{E}(\mathbf{c}), \mathbf{g}) &= \frac{d}{d\theta} \mathscr{E}(\mathbf{c}+\theta\mathbf{g})|_{\theta=0} \\
&= \frac{d}{d\theta} \int_\Omega \sum_{k=1}^{3} \left[\frac{F(c_k+\theta g_k)}{\varepsilon^2} + \frac{1}{2}|\nabla(c_k+\theta g_k)|^2 \right. \\
&\quad \left. + \frac{\lambda}{2}(c_k+\theta g_k - f_k)^2 \right] d\mathbf{x} \Big|_{\theta=0} \\
&= \frac{d}{d\theta} \int_\Omega \sum_{k=1}^{3} \left[\frac{1}{4\varepsilon^2}(c_k+\theta g_k)^2 (c_k+\theta g_k - 1)^2 \right. \\
&\quad + \frac{1}{2}[(c_k+\theta g_k)_x]^2 + \frac{1}{2}[(c_k+\theta g_k)_y]^2 \\
&\quad \left. + \frac{\lambda}{2}(c_k+\theta g_k - f_k)^2 \right] d\mathbf{x} \Big|_{\theta=0} \\
&= \sum_{k=1}^{3} \int_\Omega \left[\frac{1}{\varepsilon^2} c_k(c_k-0.5)(c_k-1) g_k \right. \\
&\quad \left. + (c_k)_x (g_k)_x + (c_k)_y (g_k)_y + \lambda(c_k - f_k) g_k \right] d\mathbf{x} \\
&= \int_\Omega \left[\frac{F'(\mathbf{c})}{\varepsilon^2} - \Delta\mathbf{c} + \lambda(\mathbf{c}-\mathbf{f}) \right] \cdot \mathbf{g}\, d\mathbf{x} \\
&= \int_\Omega \left[\frac{F'(\mathbf{c})}{\varepsilon^2} - \Delta\mathbf{c} + \lambda(\mathbf{c}-\mathbf{f}) + \beta(\mathbf{c})\mathbf{1} \right] \cdot \mathbf{g}\, d\mathbf{x} \\
&= \left(\frac{F'(\mathbf{c})}{\varepsilon^2} - \Delta\mathbf{c} + \lambda(\mathbf{c}-\mathbf{f}) + \beta(\mathbf{c})\mathbf{1},\ \mathbf{g} \right),
\end{aligned}$$

where $F'(\mathbf{c}) = (F'(c_1), F'(c_2), F'(c_3))$, $\mathbf{c} = (c_1, c_2, c_3)$, $\mathbf{f} = (f_1, f_2, f_3)$, $\beta(\mathbf{c}) = -\sum_{k=1}^{3} F'(c_k)/(3\varepsilon^2)$, and $\mathbf{1} = (1,1,1)$. Here, we used the

zero Neumann boundary condition and the fact that $\int_\Omega \beta(\mathbf{c})\mathbf{1}\cdot\mathbf{g}\,d\mathbf{x} = \int_\Omega \beta(\mathbf{c})\sum_{k=1}^{3} g_k(\mathbf{x})\,d\mathbf{x} = 0$. Therefore, we have

$$\text{grad}\ \mathscr{E}(\mathbf{c}) = \frac{F'(\mathbf{c})}{\varepsilon^2} - \Delta\mathbf{c} + \lambda(\mathbf{c}-\mathbf{f}) + \beta(\mathbf{c})\mathbf{1}. \tag{A2}$$

From Eqs. (A1) and (A2), we drive the governing equations for the tAC as shown in Eq. (2).

REFERENCES

[1] H. Zhang, J. Qin, Y. Wang, Y. Ma, L. Yao, and J. Lei, "Research on android multi-classification based on text," J. Phys.: Conf. Ser. **1828**(1), 012049 (2021).
[2] J. Cai, J. Li, W. Li, and J. Wang, "Deep learning model used in text classification," in *2018 15th ICCWAMTIP* (IEEE, 2018), pp. 123–126.
[3] H. H. Sultan, N. M. Salem, and W. Al-Atabany, "Multi-classification of brain tumor images using deep neural network," IEEE Access **7**, 69215–69225 (2019).
[4] H. Zhou, F. Xie, Z. Jiang, J. Liu, S. Wang, and C. Zhu, "Multi-classification of skin diseases for dermoscopy images using deep learning," in *2017 IEEE IST* (IEEE, 2017), Vols. 1–5.
[5] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Multi-classification approaches for classifying mobile app traffic," J. Network Comput. Appl. **103**, 131–145 (2018).
[6] A. Dainotti, A. Pescapé, and C. Sansone, "Early classification of network traffic through multi-classification," in *International Workshop on Traffic Monitoring and Analysis* (Springer, Berlin, Heidelberg, 2011), Vol. 6613, pp. 122–135.
[7] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," Neural Process. Lett. **9**(3), 293–300 (1999).
[8] K. Parand, A. Aghaei, M. Jani, and A. Ghodsi, "A new approach to the numerical solution of Fredholm integral equations using least squares-support vector regression," Math. Comput. Simul. **180**(2021), 114–128 (2021).
[9] V. Franc and V. Hlavác, "Multi-class support vector machine," in *2002 International Conference on Pattern Recognition* (IEEE, 2002), Vol. 2, pp. 236–239.
[10] S.-G. Chen and X.-J. Wu, "Multiple birth least squares support vector machine for multi-class classification," J. Mach. Learn. Cybern. **8**(6), 1731–1742 (2017).
[11] R. Khemchandani and P. Saigal, "Color image classification and retrieval through ternary decision structure based multi-category TWSVM," Neurocomputing **165**, 444–455 (2015).
[12] C.-C. Chang, L.-J. Chien, and Y.-J. Lee, "A novel framework for multi-class classification via ternary smooth support vector machine," Pattern Recognit. **44**(6), 1235–1244 (2011).
[13] C. Wu, T. Chen, R. Jiang, L. Ning, and Z. Jiang, "ANN based multi-classification using various signal processing techniques for bearing fault diagnosis," Int. J. Control Autom. **8**(7), 113–124 (2015).
[14] Y. Wang, Q. Guan, I. Lao, L. Wang, Y. Wu, D. Li, Q. Ji, Y. Wang, Y. Zhu, H. Lu, and J. Xiang, "Using deep convolutional neural networks for multi-classification of thyroid tumor by histopathology: A large-scale pilot study," Ann. Transl. Med. **7**(18), 468 (2019).
[15] S. Kang, H. Park, and J.-I. Park, "CNN-based ternary classification for image steganalysis," Electronics **8**(11), 1225 (2019).
[16] M. Raju, V. P. Gopi, and V. S. Anitha, "*Multi-class classification of alzheimer's disease using 3DCNN features and multilayer perceptron*," in *2021 Sixth International Conference on WiSPNET* (IEEE, 2021), pp. 368–373.
[17] Z. Rajabi, A. Shehu, and O. Uzuner, "A multi-channel BiLSTM-CNN model for multilabel emotion classification of informal text," in *2020 IEEE 14th ICSC* (IEEE, 2020), pp. 303–306.
[18] S. D. Wickramaratne and M. S. Mahmud, "A deep learning based ternary task classification system using Gramian angular summation field in fNIRS neuroimaging data," in *2020 IEEE International Conference on E-Health Networking, Application & Services (HEALTHCOM)* (IEEE, 2021), Vol. 1–4.
[19] B. J. Samajpati and S. D. Degadwala, "Hybrid approach for apple fruit diseases detection and classification using random forest classifier," in *2016* (ICCSP IEEE, 2016), pp. 1015–1019.
[20] A. Sarica, A. Cerasa, and A. Quattrone, "Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: A systematic review," Front. Aging Neurosci. **9**, 329 (2017).
[21] X. Gai and Y. Zhang, "Diagnosis of hepatobiliary disease based on logistic regression model," IOP Conf. Ser.: Mater. Sci. Eng. **490**(6), 062084 (2019).
[22] D. Liu, T. Li, and D. Liang, "Incorporating logistic regression to decision-theoretic rough sets for classifications," Int. J. Approx. Reason. **55**(1), 197–210 (2014).
[23] S. M. Allen and J. W. Cahn, "A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening," Acta Metall. **27**(6), 1085–1095 (1979).
[24] E.-S. Kim, S.-H. Choi, D.-H. Lee, K.-J. Kim, Y.-M. Bae, and Y.-C. Oh, "An oversampling method for wafer map defect pattern classification considering small and imbalanced data," Comput. Ind. Eng. **162**, 107767 (2021).
[25] Y. Okadome and T. Aizono, "Adversarial data-selection based work-hours estimation method on a small dataset in a logistics center," Comput. Ind. Eng. **164**, 107872 (2022).
[26] J. W. Thomas, *Numerical Partial Differential Equations: Finite Difference Methods* (Springer Science & Business Media, New York, 2013), Vol. 22.
[27] J. Kim, "Phase-field models for multi-component fluid flows," Commun. Comput. Phys. **12**(3), 613–661 (2012).
[28] K. Bhattacharya, B. Hosseini, N. B. Kovachki, and A. M. Stuart, "Model reduction and neural networks for parametric PDES," arXiv:2005.03180 (2020).
[29] U. Yadav, S. Pathrudkar, and S. Ghosh, "Interpretable machine learning model for the deformation of multiwalled carbon nanotubes," Phys. Rev. B **103**(3), 035407 (2021).
[30] S. Pathrudkar, H. M. Yu, S. Ghosh, and A. S. Banerjee, "Machine learning based prediction of the electronic structure of quasi-one-dimensional materials under strain," arXiv:2202.00930 (2022).
[31] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," Comput. Stat. Data Anal. **143**, 106839 (2020).