UDC 621
DOI: 10.15587/1729-4061.2022.254285

DEVELOPING PLASTIC RECYCLING CLASSIFIER BY DEEP LEARNING AND DIRECTED ACYCLIC GRAPH RESIDUAL NETWORK

Ahmed Burhan Mohammed
Lecturer
Department of Media
College of Arts
University of Kirkuk, Kirkuk, Iraq

Ahmad Abdullah Mohammed Al-Mafrji
Lecturer
Department of Mathematics College of Education for pure Sciences
University Of Kirkuk, Kirkuk, Iraq

Moumena Salah Yassen
Assistant Lecturer
Department of Software
College of computer science and information technology
University of Kirkuk, Kirkuk, Iraq

Ahmad H. Sabry
Corresponding author
Doctor of Control and Automation Engineering
Department ofInstitute of Sustainable Energy
Universiti Tenaga Nasional
Jalan Ikram-Uniten, Kajang, Selangor, Malaysia, 43000
E-mail: ahs4771384@gmail.com

Recycling is one of the most important approaches to safeguard the environment since it aims to reduce waste in landfills while conserving natural resources. Using deep Learning networks, this group of wastes may be automatically classified on the belts of a waste sorting plant. However, a basic set of connected layers may not be adequate to give satisfactory accuracy for such multi output classifier tasks. To optimize the gradient flow and enable deeper training for network design with multi label classifier, this study suggests a residual-based deep learning convolutional neural network. For network training, ten classes have been explored. The Directed Acyclic Graph (DAG) is a structure with hidden layers that have inputs, outputs, and other layers. The DAG network's residual-based architecture features shortcut connections that bypass some levels of the network, allowing gradients of network parameters to travel freely among the network output layers for deeper training. The methodology includes:

1) preparing the data and creating an augmented image data store;

2) defining the main serially-connected branches of the network architecture;

3) defining the residual interconnections that bypass the main branch layers;

4) defining layers, and finally;

5) creating a residual-based deeper layer graph.

The concept is to split down the multiclass classification problem into minor binary states, where every classifier performs as an expert by concentrating on discriminating between only two labels, improving total accuracy. The results achieve (2.861 %) training error and (9.76 %) a validation error. The training results of this classifier are evaluated by finding the training error, validation error, and showing the confusion matrix of validation data

Keywords: Directed Acyclic Graph (DAG), deep learning, Recycling, classification, Convolutional Neural Network (CNN)

Received date 26.02.2022
Accepted date 06.04.2022
Published date 29.04.2022

How to Cite: *Mohammed, A. B., Al-Mafrji, A. A. M., Yassen, M. S., Sabry, A. H. (2022). Developing plastic recycling classifier by deep learning and directed acyclic graph residual network. Eastern-European Journal of Enterprise Technologies, 2 (10 (116)), 42–49. doi: https://doi.org/10.15587/1729-4061.2022.254285*

1. Introduction

Recycling is one of the majority significant ways of environmental protection, to reduce waste in landfills while also conserving natural resources. The term “recycling” refers to the process of transforming materials or substances present in trash throughout the manufacturing method in order to get the material or substance for reuse or other functions, including natural recycling, but not energy recovery. The amount of rubbish produced in the previous century increased dramatically. At the same time, the idea of their reuse emerged, and one pointed out that waste disposed of in landfills is not environmentally friendly. Using computer vision algorithms, this group of debris may be automatically chosen on the belts of a waste sorting plant. The classification belt for sorting plastic bottles and types of these recycled waste are shown in Fig. 1.

Fig. 1. Recycling plant: *a* — belt of sorting; *b* — ten types of plastic bottles

The exact characteristics of locales where genuine objects will be found should be reflected in digital photographs of items that will be processed. Many real-world machine learning issues require learning representations of data of graph demonstration; hence there has been a strong push in the representation learning community to adopt deep learning approaches for graph-structured data, either as input or output of the model. Almost all of these strategies, however, only consider the input or output graph space, not both. In image segmentation, deep learning has greatly decreased the requirement for manual feature selection. Network architecture optimization and hyper parameter tuning, on the other hand, are mostly manual and time-consuming [1]. In some applications, a simple sequence of network layers is satisfactory. In contrast, other applications necessitate networks of complex graph configuration, where layers may have outputs to multiple layers and inputs from hidden layers. Such kinds of networks are usually named Directed Acyclic Graph (DAG) networks. DAG networks not only merge the feature classification and extraction stages of input estimates into a single automated learning operation, but they also use multi-scale features and automatically combine the scores of several classifiers. The performance of such models can be improved by fine-tuning them [2].

DAG-CNN sounds complex, but it's not. A graph is made up of vertices (or points) and edges (or lines) that show how the vertices are connected. The boxes are sometimes called vertices or the lines that connect the boxes in a graph are called edges or arcs. A directed graph with no directed cycles is referred to as a DAG in mathematics and computer science. That is, it is made up of edges and vertices, which are also known as arcs, that are directed from one vertex to the next in such a way that following those directions will never result in a closed-loop. If and only if the vertices of a directed graph can be topologically ranked in a linear order that is consistent across all edge directions, it is called a DAG. From biology (evolution, family trees, epidemiology) to sociology (citation networks) to computation, DAGs have a wide range of scientific and computational applications (scheduling).An acyclic graph has a rule about avoiding cycles and this is the directed part of a dag, which means that the flow has a defined direction. Therefore, all together form a graph with the rule that the flow goes in a specific direction, and that it contains no cycles. Four types of the graph are shown in Fig. 2.

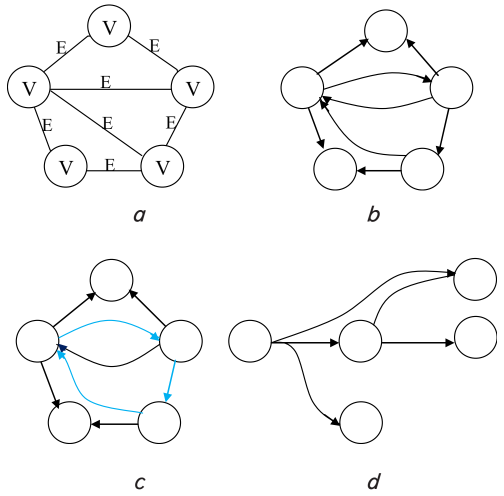


Fig. 2. Several types of graph: *a* — simple graph; *b* — simple digraph; *c* — digraph with a highlighted cycle; *d* — simple dag

The data in a graph database is stored in the form of nodes and connections. These nodes and connections are what we call a graph. Several kinds of research reveal that the Directed Acyclic Graph (DAG) is a significant approach to expand a robust automated and consistent diagnosis process for the multi-class image classifications [3, 4]. Thanks to recent advances in the field of computer vision, particularly deep learning, many fully connected and convolutional neural networks have been trained to achieve state-of-the-art performance on a wide variety of tasks, including speech recognition, image classification, and natural language processing. Most deep learning models for classification tasks, on the other hand, use the softmax activation function to anticipate and minimize cross-entropy loss [5]. This architecture has been used with child maltreatment [6]. DAG-CNN eliminates the need for hand-crafted features to be extracted [4].

Recycling is an important strategy for protecting the environment since it tries to reduce waste in landfills while conserving natural resources. This category of garbage could be automatically classified on the belts of a waste sorting factory using deep learning networks. Therefore, it is very important to conduct research on this topic, and the results of such research are necessary for environmental sustainability.

2. Literature review and problem statement

Deep convolutional Directed Acyclic Graph (DAG) is discussed by several studies in different applications. In medical applications, the paper [7] used DAG to classify leukemia disease based on microscopic images. Different deep learning algorithms have been examined including Directed Acyclic Graph (DAG) networks, MobileNet-v2, Inceptionv3, VGG-16, VGG-19, AlexNet, residual networks ResNet-18, Inception-ResNet-v2, GoogLeNet, Xception, DenseNet-201, ResNet-50 and ResNet-101. The obtained accuracy was 100 % for all except for VGG-16 AlexNet. Although the study compared different networks, the applications were limited to a relatively small number of blood smears images. In the same context, the study [3] addressed the application of Melanoma skin lesions, where dermoscopic images were acquired for nevi, seborrheic keratosis, and melanoma. The study used ISIC 2018 public dataset and obtained an accuracy of (76.6 %). However, the architecture didn't achieve high accuracy to classify the skin lesion images.

Alzheimer's disease in magnetic resonance imaging representation was discussed by [8], where three public datasets including MIRIAD, ADNI-2, and ADNI-1 were employed. Although the morphological and statistical analysis was used to identify the anatomical feature patches in the gray matter imaging to extract the discriminative deep features of picture representation using DAG, this system may only forecast the individual's risk of Alzheimer's disease and the study didn't show quantitative accuracy for the presented classifier. 3D medical image segmentation for 43 3D-brain magnetic resonance images was studied in [1]. The study achieved an average Dice coefficient of 82 % but it was implemented with the complicated architecture of more than 100 layers. The paper [4] classified heartbeats of electrocardiogram (ECG) signals. The dataset (MIT-BIH arrhythmia benchmarks) was used to verify the proposed idea. However, they predict output label was not clear and complex.

In another application, the images of time series crude oil prices were classified into "sell" or "buy" by using DAG and CNN structure [9], and in quality control systems of food in [10]. Although their results of DAG performed better with a sensitivity of 100 %, specificity of 99.19 %, and an accuracy of 99.16 %, this work presented a very complex architecture. The paper [2] applied the DAG network with facial age estimation using publicly available Morph-II and FG-NET databases. However, this work is limited to classifying limited images. This limitation has been avoided by [10], where the DAG has been applied on 1600 test images classification with 94.43 % accuracy.

In most classification problems, basic layer sequences might be sufficient to achieve acceptable accuracy. In some other applications, networks with a more sophisticated graph structure, in which layers can receive inputs from many layers and outputs from multiple layers, are required. Directed acyclic graph (DAG) networks are the most common type of network. The primary network layers of a DAG with a residual network are shortcut or bypassed the residual connections. The advantage of such residual links is to allow network parameters to gradient and move further simply between the output layer and the network earlier layers. This allows training deeper networks with flexibility. Superior accuracy can obtain on increasingly challenging tasks that may achieve therefore greater network strength.

3. The aim and objectives of research

The aim of research is to develop a plastic recycling classifier by deep learning and directed acyclic graph residual network to get accurate classification results.

To achieve this aim, the following objectives are accomplished:

– to train the developed DAG network so that it can classify 10 kinds of plastic bottles of non-uniform shape and size;

– to evaluate the training DAG network via confusion matrix for the validation images;

– to compare the developed DAG classification network with other residual networks U-net CNN.

4. Methods and materials

There are 40,000 images in the data set. Each image has three color channels and is 64×64 pixels in size (RGB) [11]. Each item in this database should be presented throughout numerous image collections, taking into consideration various lighting circumstances, positioning relative to the image recorder, and the degree of distortion. The images in the collection were sorted into groups based on the type of material used to build the unique objects.

The methodology includes the following steps:

1. Preparing the data by downloading the dataset, specifying images type, size, and creating an augmented image data store. 80 % of the dataset is used as training images, while the remaining 20 % are used for testing and validation. Fig. 3 shows a sample of the dataset.

Throughout the training stage, the data-store arbitrarily flips the trained images over the perpendicular and converts them to four vertically and horizontally pixels. Augmentation of data allows preventing the network from over-fit and memorizing the precise information of the trained images.

2. Defining the main serially-connected branches of the network architecture, convolutional and ReLu layers, and batch normalization. Fig. 4, *a* shows the serially-connected convolutional layers.

3. Defining the residual interconnections that bypass the main branch layers, as shown in Fig. 4, *b*.

4. Defining global average-pooling layers, softmax, Fully-Connected, and classification units and, as shown in Fig. 4, *c*.

5. Creating a residual-based deeper layer graph. The concept is to split down the multiclass classification problem into minor binary states, where every classifier works as a professional by focusing on discriminating between only two labels, improving total accuracy as shown in Fig. 4, *d*.

Therefore, the created network is a deep residual network of 16 widths and 9 normal convolutional elements (3 for each stage). As a result, 20 (2*9+2) is the overall network depth.

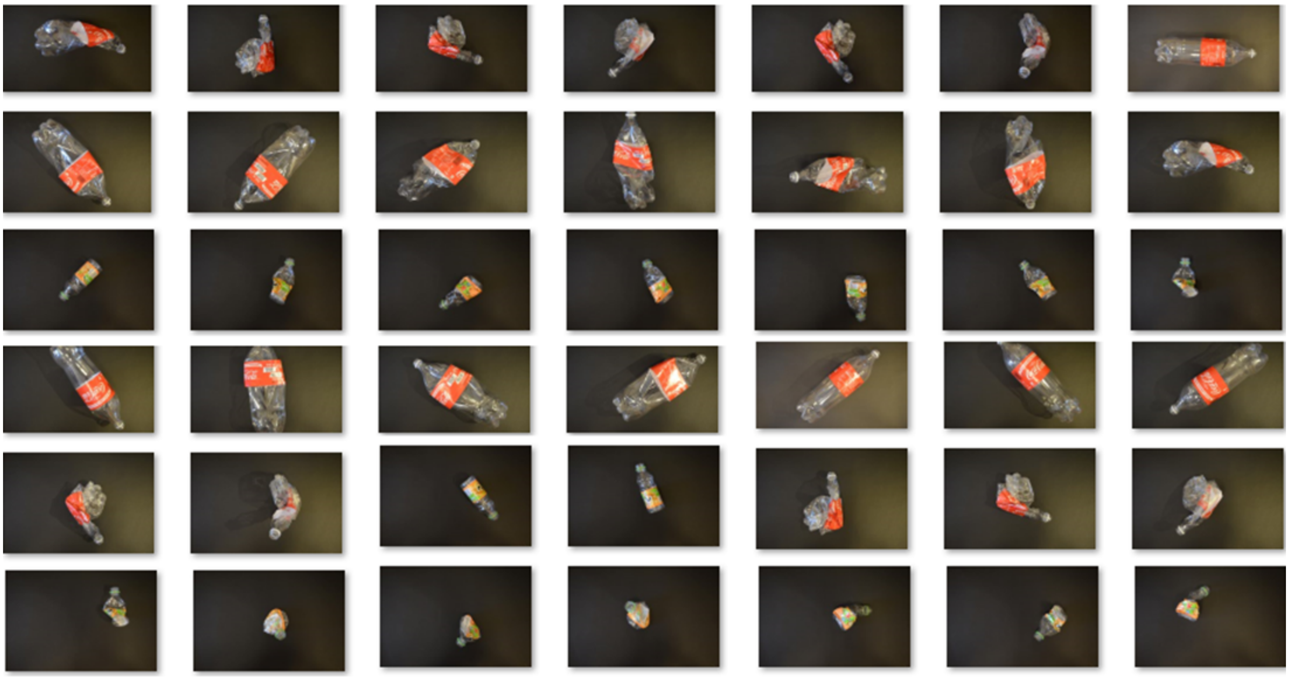


Fig. 3. Sample from the considered dataset

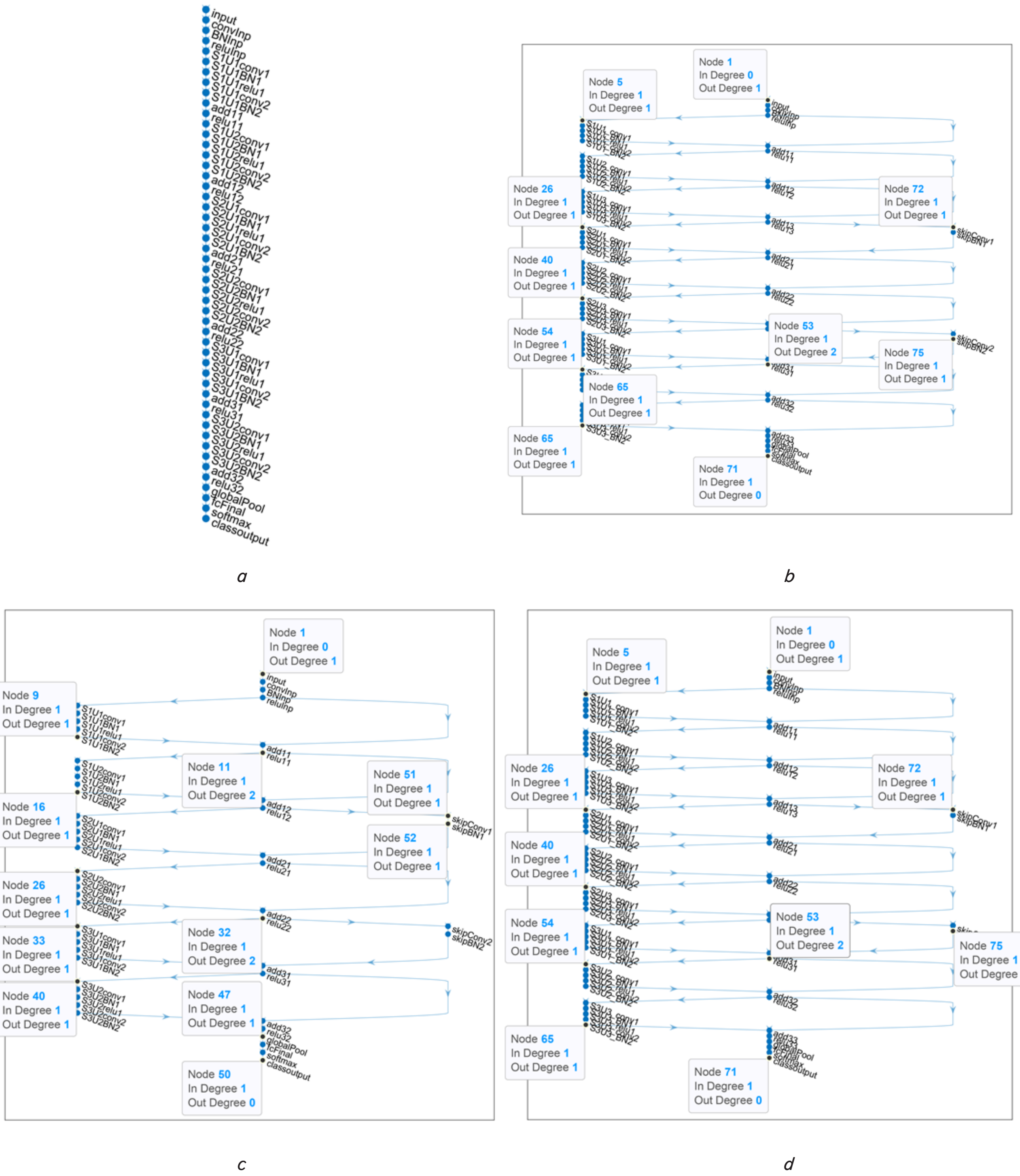

input
convInp
BNInp
reluInp
S1U1conv1
S1U1BN1
S1U1relu1
S1U1conv2
S1U1BN2
add11
relu11
S1U2conv1
S1U2BN1
S1U2relu1
S1U2conv2
S1U2BN2
add12
relu12
S2U1conv1
S2U1BN1
S2U1relu1
S2U1conv2
S2U1BN2
add21
relu21
S2U2conv1
S2U2BN1
S2U2relu1
S2U2conv2
S2U2BN2
add22
relu22
S3U1conv1
S3U1BN1
S3U1relu1
S3U1conv2
S3U1BN2
add31
relu31
S3U2conv1
S3U2BN1
S3U2relu1
S3U2conv2
S3U2BN2
add32
relu32
globalPool
fcFinal
softmax
classoutput
a
Node 1 In Degree 0 Out Degree 1
Node 5 In Degree 1 Out Degree 1
Node 26 In Degree 1 Out Degree 1
Node 72 In Degree 1 Out Degree 1
Node 40 In Degree 1 Out Degree 1
Node 54 In Degree 1 Out Degree 1
Node 53 In Degree 1 Out Degree 2
Node 75 In Degree 1 Out Degree 1
Node 65 In Degree 1 Out Degree 1
Node 65 In Degree 1 Out Degree 1
Node 71 In Degree 1 Out Degree 0
skipConv1
skipBN1
skipConv2
skipBN2
b
Node 1 In Degree 0 Out Degree 1
Node 9 In Degree 1 Out Degree 1
Node 11 In Degree 1 Out Degree 2
Node 51 In Degree 1 Out Degree 1
Node 16 In Degree 1 Out Degree 1
Node 52 In Degree 1 Out Degree 1
Node 26 In Degree 1 Out Degree 1
Node 32 In Degree 1 Out Degree 2
Node 33 In Degree 1 Out Degree 1
Node 47 In Degree 1 Out Degree 1
Node 40 In Degree 1 Out Degree 1
Node 50 In Degree 1 Out Degree 0
skipConv1
skipBN1
skipConv2
skipBN2
c
Node 1 In Degree 0 Out Degree 1
Node 5 In Degree 1 Out Degree 1
Node 26 In Degree 1 Out Degree 1
Node 72 In Degree 1 Out Degree 1
Node 40 In Degree 1 Out Degree 1
Node 54 In Degree 1 Out Degree 1
Node 53 In Degree 1 Out Degree 2
Node 75 In Degree 1 Out Degree 1
Node 65 In Degree 1 Out Degree 1
Node 71 In Degree 1 Out Degree 0
skipConv1
skipBN1
d

Fig. 4.The developed deep network: *a* — serially-connected convolutional layers; *b* — bypassing the main branch layers; *c* — defining softmax, fully-connected, and classification units; *d* — network with residual-based deeper layer graph

5. Results of the developing recycling classifier

5. 1. Training the developed DAG network classifier

The MATLAB-based environment has been used to implement the developed DAG network. The developed DAG network has been trained for 85 epochs with a learning rate proportional to the size of the mini-batch. The final DAG network of the developed structure is shown in Fig. 5, while a picture for the table describing the network parameters is shown in Fig. 6.

The training progress of the developed DAG network to classify 10 kinds of bottles is shown in Fig. 7.

The training process has been stopped on the 29th Epoch of 80 with 536 minutes elapsed time, where the accuracy converges to 98 % and the loss to 0.05.

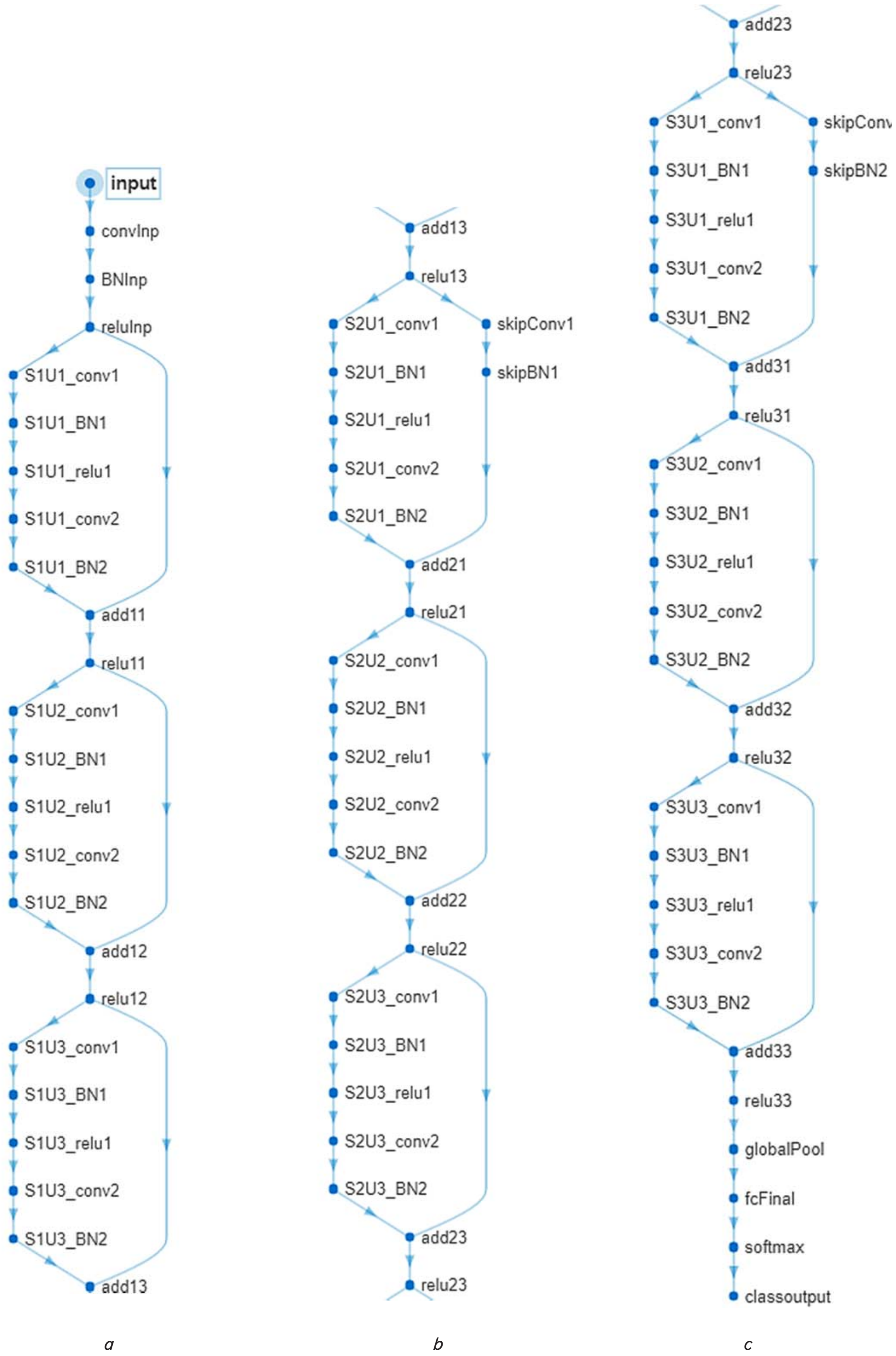
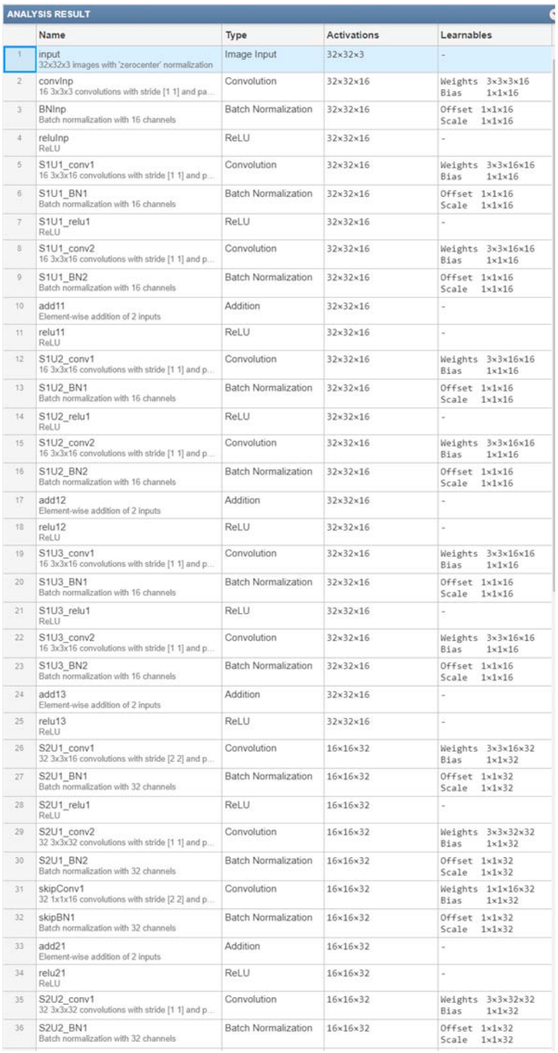


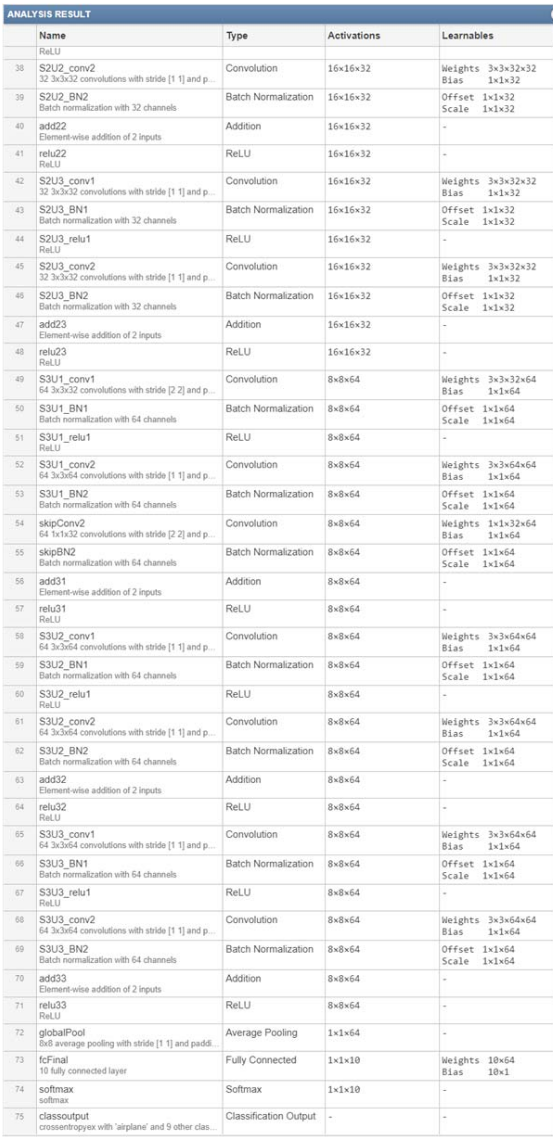
Fig. 5. DAG network of the developed structure: *a* – 1st stage; *b* – 2nd stage; *c* – 3rd stage



ANALYSIS RESULT

	Name	Type	Activations	Learnables
1	input 32x32x3 images with 'zerocenter' normalization	Image Input	32×32×3	-
2	convInp 16 3x3x3 convolutions with stride [1 1] and pa...	Convolution	32×32×16	Weights 3×3×3×16 Bias 1×1×16
3	BNInp Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
4	reluInp ReLU	ReLU	32×32×16	-
5	S1U1_conv1 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
6	S1U1_BN1 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
7	S1U1_relu1 ReLU	ReLU	32×32×16	-
8	S1U1_conv2 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
9	S1U1_BN2 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
10	add11 Element-wise addition of 2 inputs	Addition	32×32×16	-
11	relu11 ReLU	ReLU	32×32×16	-
12	S1U2_conv1 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
13	S1U2_BN1 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
14	S1U2_relu1 ReLU	ReLU	32×32×16	-
15	S1U2_conv2 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
16	S1U2_BN2 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
17	add12 Element-wise addition of 2 inputs	Addition	32×32×16	-
18	relu12 ReLU	ReLU	32×32×16	-
19	S1U3_conv1 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
20	S1U3_BN1 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
21	S1U3_relu1 ReLU	ReLU	32×32×16	-
22	S1U3_conv2 16 3x3x16 convolutions with stride [1 1] and p...	Convolution	32×32×16	Weights 3×3×16×16 Bias 1×1×16
23	S1U3_BN2 Batch normalization with 16 channels	Batch Normalization	32×32×16	Offset 1×1×16 Scale 1×1×16
24	add13 Element-wise addition of 2 inputs	Addition	32×32×16	-
25	relu13 ReLU	ReLU	32×32×16	-
26	S2U1_conv1 32 3x3x16 convolutions with stride [2 2] and p...	Convolution	16×16×32	Weights 3×3×16×32 Bias 1×1×32
27	S2U1_BN1 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
28	S2U1_relu1 ReLU	ReLU	16×16×32	-
29	S2U1_conv2 32 3x3x32 convolutions with stride [1 1] and p...	Convolution	16×16×32	Weights 3×3×32×32 Bias 1×1×32
30	S2U1_BN2 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
31	skipConv1 32 1x1x16 convolutions with stride [2 2] and p...	Convolution	16×16×32	Weights 1×1×16×32 Bias 1×1×32
32	skipBN1 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
33	add21 Element-wise addition of 2 inputs	Addition	16×16×32	-
34	relu21 ReLU	ReLU	16×16×32	-
35	S2U2_conv1 32 3x3x32 convolutions with stride [1 1] and p...	Convolution	16×16×32	Weights 3×3×32×32 Bias 1×1×32
36	S2U2_BN1 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32

a

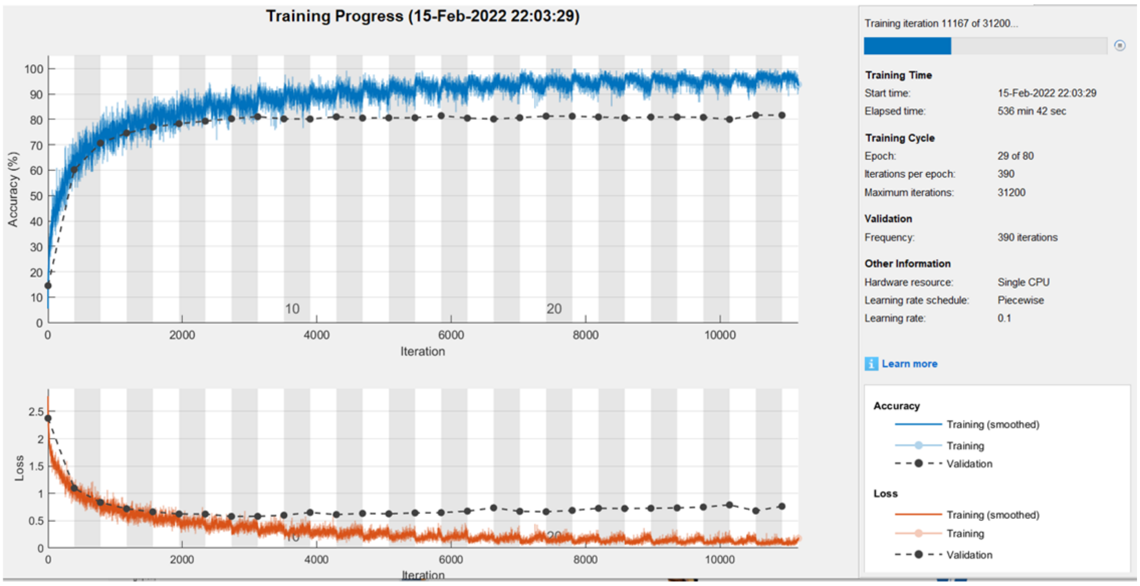


ANALYSIS RESULT

	Name	Type	Activations	Learnables
	ReLU			
38	S2U2_conv2 32 3x3x32 convolutions with stride [1 1] and p...	Convolution	16×16×32	Weights 3×3×32×32 Bias 1×1×32
39	S2U2_BN2 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
40	add22 Element-wise addition of 2 inputs	Addition	16×16×32	-
41	relu22 ReLU	ReLU	16×16×32	-
42	S2U3_conv1 32 3x3x32 convolutions with stride [1 1] and p...	Convolution	16×16×32	Weights 3×3×32×32 Bias 1×1×32
43	S2U3_BN1 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
44	S2U3_relu1 ReLU	ReLU	16×16×32	-
45	S2U3_conv2 32 3x3x32 convolutions with stride [1 1] and p...	Convolution	16×16×32	Weights 3×3×32×32 Bias 1×1×32
46	S2U3_BN2 Batch normalization with 32 channels	Batch Normalization	16×16×32	Offset 1×1×32 Scale 1×1×32
47	add23 Element-wise addition of 2 inputs	Addition	16×16×32	-
48	relu23 ReLU	ReLU	16×16×32	-
49	S3U1_conv1 64 3x3x32 convolutions with stride [2 2] and p...	Convolution	8×8×64	Weights 3×3×32×64 Bias 1×1×64
50	S3U1_BN1 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
51	S3U1_relu1 ReLU	ReLU	8×8×64	-
52	S3U1_conv2 64 3x3x64 convolutions with stride [1 1] and p...	Convolution	8×8×64	Weights 3×3×64×64 Bias 1×1×64
53	S3U1_BN2 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
54	skipConv2 64 1x1x32 convolutions with stride [2 2] and p...	Convolution	8×8×64	Weights 1×1×32×64 Bias 1×1×64
55	skipBN2 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
56	add31 Element-wise addition of 2 inputs	Addition	8×8×64	-
57	relu31 ReLU	ReLU	8×8×64	-
58	S3U2_conv1 64 3x3x64 convolutions with stride [1 1] and p...	Convolution	8×8×64	Weights 3×3×64×64 Bias 1×1×64
59	S3U2_BN1 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
60	S3U2_relu1 ReLU	ReLU	8×8×64	-
61	S3U2_conv2 64 3x3x64 convolutions with stride [1 1] and p...	Convolution	8×8×64	Weights 3×3×64×64 Bias 1×1×64
62	S3U2_BN2 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
63	add32 Element-wise addition of 2 inputs	Addition	8×8×64	-
64	relu32 ReLU	ReLU	8×8×64	-
65	S3U3_conv1 64 3x3x64 convolutions with stride [1 1] and p...	Convolution	8×8×64	Weights 3×3×64×64 Bias 1×1×64
66	S3U3_BN1 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
67	S3U3_relu1 ReLU	ReLU	8×8×64	-
68	S3U3_conv2 64 3x3x64 convolutions with stride [1 1] and p...	Convolution	8×8×64	Weights 3×3×64×64 Bias 1×1×64
69	S3U3_BN2 Batch normalization with 64 channels	Batch Normalization	8×8×64	Offset 1×1×64 Scale 1×1×64
70	add33 Element-wise addition of 2 inputs	Addition	8×8×64	-
71	relu33 ReLU	ReLU	8×8×64	-
72	globalPool 8x8 average pooling with stride [1 1] and paddi...	Average Pooling	1×1×64	-
73	fcFinal 10 fully connected layer	Fully Connected	1×1×10	Weights 10×64 Bias 10×1
74	softmax softmax	Softmax	1×1×10	-
75	classoutput crossentropyex with 'airplane' and 9 other clas...	Classification Output	-	-

b

Fig. 6. A picture of the table describing the network parameters: *a* — first stage; *b* — 2nd stage



Training iteration 11167 of 31200...

Training Time

Start time:	15-Feb-2022 22:03:29
Elapsed time:	536 min 42 sec

Training Cycle

Epoch:	29 of 80
Iterations per epoch:	390
Maximum iterations:	31200

Validation

Frequency:	390 iterations

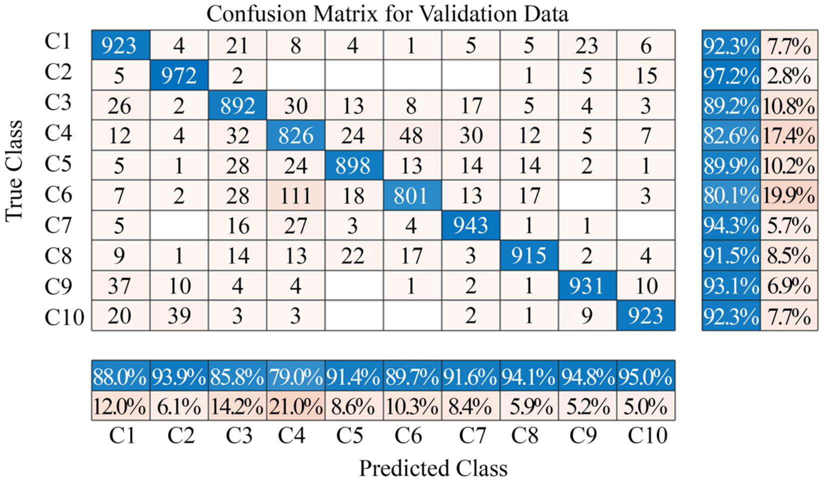
Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Piecewise
Learning rate:	0.1

Fig. 7. The training progress of the developed DAG network to classify 10 kinds of plastic bottles

5. 2. Results of the trained network evaluation

To evaluate the trained DAG network for the applied dataset, accuracy is obtained. The confusion matrix for the validation images is shown in Fig. 8.



	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10		
C1	923	4	21	8	4	1	5	5	23	6	92.3%	7.7%
C2	5	972	2					1	5	15	97.2%	2.8%
C3	26	2	892	30	13	8	17	5	4	3	89.2%	10.8%
C4	12	4	32	826	24	48	30	12	5	7	82.6%	17.4%
C5	5	1	28	24	898	13	14	14	2	1	89.9%	10.2%
C6	7	2	28	111	18	801	13	17		3	80.1%	19.9%
C7	5		16	27	3	4	943	1	1		94.3%	5.7%
C8	9	1	14	13	22	17	3	915	2	4	91.5%	8.5%
C9	37	10	4	4		1	2	1	931	10	93.1%	6.9%
C10	20	39	3	3			2	1	9	923	92.3%	7.7%
	88.0%	93.9%	85.8%	79.0%	91.4%	89.7%	91.6%	94.1%	94.8%	95.0%		
	12.0%	6.1%	14.2%	21.0%	8.6%	10.3%	8.4%	5.9%	5.2%	5.0%		

Fig. 8. The confusion matrix for the validation images

This figure shows the recall and precision of each label/class based on a row-column summary. C1 to C5 represents the classes of the bottles in the images of the dataset. It is found that the developed DAG classification network most commonly perturbs C6 with C4 images.

5. 3. Results of the comparison with ResNet50 network

Global accuracy, mean accuracy, Intersection Over-Union (IOU), mean BF-scores, and weighted IOU are used to evaluate how well the model performs in segmentation tasks. Global accuracy refers to the proportion of correctly identified pixels in each class that are higher than the total number of pixels, whereas mean accuracy refers to the average percentage of properly identified pixels in each Label/class. The mean IOU, also identified as the Jaccard similarity coefficient, is the averaged IOU of each class. IOU is defined by:

$$IOU = \frac{Tp}{Tp + Fp + Fn}.$$

The letters *Fp*, *Fn*, and *Tp* stand for false positives, false negatives, and true positives respectively. To prevent the larger class from overlapping the smaller class, the weighted IOU specifies how many pixels of each class are weighted in the inconsistent pixel label/class. The performance parameters of the developed DAG classification network as compared with the U-net CNN network are shown in Table 1.

Table 1

Performance evaluation of the developed DAG classification network

Network name	Mean IOU	Weighted IOU	Global accuracy	Mean accuracy	Training time
U-net	0.5587	0.9397	0.9625	0.9048	542 min
DAG network	0.69313	0.9846	0.9898	0.915	536 min

The values of Table 1 show that the performance parameters of the developed DAG classification network perform better as compared with the U-net CNN network.

6. Discussion of the results of developed DAG Network

The learning rate has been decreased by a factor of 10 on every 55 epochs, while the validation process occurred at each epoch with the use of the corresponding validation images. The input images have been reduced to 50 % in size to speed up the training process, which has been performed on a 1.7GHz CPU computer and elapsed about 536 minutes. Therefore, the designed trained DAG network was able to classify 10 kinds of plastic bottles of non-uniform shape and size as indicated in Fig. 7.

The evaluation of the training DAG network has been performed by the confusion matrix in Fig. 8, which shows the recall and precision of each label/class based on a row-column summary. It is found that C1 to C5 represents the classes of the bottles in the images of the dataset and the developed DAG classification network most commonly perturbs C6 with C4 images.

The comparison of the performance evaluation of the developed DAG classification network with a U-net CNN network shows that the developed network is better to classify such type of dataset as indicated in Table 1.

The advantage of the developed DAG classification network with residual layers allows network parameters to gradient and move further simply between the output layer and the network earlier layers. This allows training deeper networks with flexibility.

Although the accuracy of the constructed DAG classification network was satisfactory for building a computer-based classifier, the disadvantage of the study was the longer training time as compared with U-net CNN as indicated in Table 1. In addition, the visual satisfaction was a time-consuming way. However, these issues may be taken into account in future work, which could be with a fully automated system incorporating all of the preceding phases.

7. Conclusions

1. The training of the developed DAG classification network was able to classify ten types of bottles images from a cover belt of non-uniform shape and size. The created network was a deep residual network of 16 widths and 9 normal convolutional elements (3 for each stage).

2. The training error was about 2.862 %, while the validation error was 9.76 %. The confusion matrix evaluation of the training process of the developed DAG network showed that the developed DAG classification network most commonly perturbs C6 with C4 images as they are very similar.

3. The comparison with the U-net CNN network [12] showed that the performance parameters of the developed DAG classification network perform better when evaluated

with the parameters; Global accuracy (0.9898), mean accuracy (0.915), Intersection Over-Union (IOU) mean BF-scores (0.69313), weighted IOU (0.9846), and (536 min) for the training time.

References

1. Wong, K. C. L., Moradi, M. (2019). SegNAS3D: Network Architecture Search with Derivative-Free Global Optimization for 3D Image Segmentation. Medical Image Computing and Computer Assisted Intervention – MICCAI 2019, 393–401. doi: https://doi.org/10.1007/978-3-030-32248-9_44
2. Taheri, S., Toygar, Ö. (2019). On the use of DAG-CNN architecture for age estimation with multi-stage features fusion. Neurocomputing, 329, 300–310. doi: https://doi.org/10.1016/j.neucom.2018.10.071
3. Foahom Gouabou, A. C., Damoiseaux, J.-L., Monnier, J., Iguernaissi, R., Moudafi, A., Merad, D. (2021). Ensemble Method of Convolutional Neural Networks with Directed Acyclic Graph Using Dermoscopic Images: Melanoma Detection Application. Sensors, 21 (12), 3999. doi: https://doi.org/10.3390/s21123999
4. Golrizkhatami, Z., Taheri, S., Acan, A. (2018). Multi-scale features for heartbeat classification using directed acyclic graph CNN. Applied Artificial Intelligence, 32 (7-8), 613–628. doi: https://doi.org/10.1080/08839514.2018.1501910
5. Agarwal, N., Balasubramanian, V. N., Jawahar, C. V. (2018). Improving multiclass classification by deep networks using DAGSVM and Triplet Loss. Pattern Recognition Letters, 112, 184–190. doi: https://doi.org/10.1016/j.patrec.2018.06.034
6. Austin, A. E., Desrosiers, T. A., Shanahan, M. E. (2019). Directed acyclic graphs: An under-utilized tool for child maltreatment research. Child Abuse & Neglect, 91, 78–87. doi: https://doi.org/10.1016/j.chiabu.2019.02.011
7. Anilkumar, K. K., Manoj, V. J., Sagi, T. M. (2021). Automated detection of leukemia by pretrained deep neural networks and transfer learning: A comparison. Medical Engineering & Physics, 98, 8–19. doi: https://doi.org/10.1016/j.medengphy.2021.10.006
8. Zhu, T., Cao, C., Wang, Z., Xu, G., Qiao, J. (2020). Anatomical Landmarks and DAG Network Learning for Alzheimer's Disease Diagnosis. IEEE Access, 8, 206063–206073. doi: https://doi.org/10.1109/access.2020.3037107
9. Oyewola, D. O., Augustine, A. F. (2021). Predicting Impact of COVID-19 on Crude Oil Price Image with Directed Acyclic Graph Deep Convolution Neural Network. Journal of Robotics and Control (JRC), 2 (2). doi: https://doi.org/10.18196/jrc.2261
10. Pachón-Suescún, C. G., Pinzón-Arenas, J. O., Jiménez-Moreno, R. (2020). Fruit Identification and Quality Detection by Means of DAG-CNN. International Journal on Advanced Science, Engineering and Information Technology, 10 (5), 2183. doi: https://doi.org/10.18517/ijaseit.10.5.8684
11. Plastic Waste DataBase of Images – WaDaBa. Available at: http://wadaba.pcz.pl/
12. Jwaid, W. M., Al-Husseini, Z. S. M., Sabry, A. H. (2021). Development of brain tumor segmentation of magnetic resonance imaging (MRI) using U-Net deep learning. Eastern-European Journal of Enterprise Technologies, 4 (9 (112)), 23–31. doi: https://doi.org/10.15587/1729-4061.2021.238957