Journal of Artificial Intelligence and Data Science (JAIDA)
e-ISSN: 2791-8335
https://jaida.ikcu.edu.tr

Vol. 1, No. 2, pp. 116-124, 2021
Research Article

Comparison of Multi-Label Learning Approaches for Pie Chart Image Classification Using Deep Learning

Derya BIRANT *, Cem KOSEMEN

Dokuz Eylul University, Faculty of Engineering, Department of Computer Engineering, Turkey

Abstract
A pie chart is a powerful and circular information graphic used to display numerical proportions to the whole. However, the properties of pie charts cannot be directly noticed by machines since they are usually in an image format. To make a pie chart classifiable by machines, this paper proposes a novel solution using deep learning methods. This study is original in that it automatically and jointly classifies charts in terms of two respects: shape (pie or donut) and dimension (2D or 3D). This is the first study that compares two multi-label learning approaches to classify pie charts: *binary-class-based convolutional neural networks* (BCNN) and *multi-class-based convolutional neural networks* (MCNN). The experimental results showed that the BCNN model achieved 86% accuracy, while the MCNN model reached 85% accuracy on real-world pie chart data.
Keywords: Convolutional neural networks; deep learning; image classification; machine learning; multi-label learning; pie charts.

1. Introduction

A *pie chart* is a useful graph in which a circle container is divided into slices (or sectors) to illustrate proportions based on percentages. The principal purpose of a pie chart is to display the relationship of a part to the whole. Pie charts are widely-used due to their visual representation advantages over textual representation, both when it comes to expressive results and comprehension. They have been widely used in various types of sources such as books, reports, web pages, and newspapers. For instance, in scientific articles, pie charts have been utilized to represent the results of the experiments. Furthermore, in the presentation slides, they are frequently used as a graphical way to illustrate the information.

Currently, pie charts have been created in image format, and therefore, they can be understandable by humans, but not naturally machine-understandable or machine-readable. However, in recent years, there is an increasing need for machines to automatically interpret the properties of pie chart images. In many applications, it is needed to give the information about a pie chart image to use this knowledge for further processes such as for providing recommendations, developing better search engines, automatically tagging the pie chart images, semantically describing charts, segmenting a collection of pie chart images, and redesigning pie charts. Based on these motivations, in this study, we developed convolutional neural network (CNN) models to automatically classify pie charts based on their properties.

Pie charts on use have different characteristic features and so they can be classified with different properties. In this study, the proposed model makes predictions based on whether the charts are 2D or 3D, and whether they are pie or donut. A *donut chart* is a variant of the pie chart with a hole in the center. 3D pie charts could provide advantages over 2D pie charts since they support the comparison of multiple series. By visualizing the multi-variate information together, the 3D setup can mitigate the potential split-attention effects more effectively than the 2D setup. On the other hand, some 3D pie charts have been criticized for not carrying more information than their 2D counterparts, tending to make readers confused.

The main contributions of this study can be summarized as follows. Previous research works on pie chart classification use the standard (single-label) classification. Our study proposes a multi-label learning approach on pie charts. This is the first study that compares two multi-label learning approaches to classify pie charts: *binary-class-based convolutional neural networks* (BCNN) and *multi-class-based convolutional neural networks* (MCNN). This study is original in that it jointly classifies charts in terms of two respects: shape (pie or donut) and dimension (2D or 3D). Furthermore, a training dataset with high variety was generated for classifying pie chart images.

The proposed CNN model was developed from labeled training data; therefore, supervised learning was performed. The pie chart images in the training dataset were generated with random properties using a Python script. However, the constructed model was tested on real-world pie chart images that were collected from the Google search engine. The experimental results showed that the BCNN model achieved 86% accuracy, while the MCNN model reached 85% accuracy on real-world pie chart data.

*Corresponding author e-mail address: derya.birant@deu.edu.tr

Received: 08/12/2021; Accepted: 27/12/2021

2. Literature Review

An amount of information is presented inside chart images and inaccessible by visually impaired people and machines. To overcome this limitation, some studies have been done on information extraction from charts, including chart recognition, chart segmentation, chart classification, and chart interpretation.

Table 1 presents the previous work with regard to chart processing. Most studies on chart classification [1-3] have been concerned with determining the types of charts such as bar chart, scatter plot, area chart, and line chart. On the other hand, some studies especially focused on a single chart type and classify it according to its visual properties such as the classification of control charts [4], line charts [5, 6], and bar charts [7].

In the literature, many different machine learning approaches have been experimented with for classifying real-world chart images by their visual properties such as random forest (RF) [4], support vector machines (SVM) [4, 8], perceptually important points (PIP) [5], time series forest (TSF) [4], and decision tree (DT) [9]. Some studies have been developed neural network-based models for classifying charts such as recurrent neural networks (RNN) [7], convolutional neural networks (CNN) [10-16], and deep belief networks (DBN) [17]. Different deep learning architectures have been tested for chart classification such as residual networks (ResNet) [2, 6, 11, 15], Alex networks (AlexNet) [2, 15, 16], visual geometry group (VGG) [1, 2, 11, 14, 15], and you only look once (YOLO) [12]. For example, Tang et al. [17] developed a model for classifying charts with deep convolutional neural networks combined with deep belief networks. They focused on the following chart categories: bar, flowchart, line, pie, and scatter plot.

Table 1. *Some previous works on chart classification.*

Ref.	Year	Methods	Description	Chart Type	Single-Label	Multi-Label
[1]	2021	CNN, VGG	Chart type classification	Area, bar, box, line, map, pareto, pie, radar, scatter, table, venn charts	√	X
[2]	2021	CNN, VGG, AlexNet, LeNet, ResNet	Chart type classification	Area, bar, bubble, histogram, donut, line, pie, scatter, stacked area charts	√	X
[3]	2021	Data embedding	Reviving chart images	Bar, line, pie, scatter plots	√	X
[4]	2021	RF, TSF, SVM, CNN	Classification of control chart patterns	Control chart	√	X
[5]	2021	PIP	Chart pattern classification in financial time series	Line chart	√	X
[6]	2021	CNN, ResNet	Line chart understanding	Line chart	√	X
[7]	2021	CNN, RNN	Reverse-engineering bar charts	Bar chart	√	X
[8]	2021	SVM	Control chart pattern recognition	Control chart	√	X
[9]	2021	DT	Classification of control chart patterns	Control chart	√	X
[10]	2021	CNN	Classifying price chart images	Line chart	√	X
[11]	2020	Xception, ResNet, VGG, MobileNet	Chart recognition via classification and detection	Arc, area, bar, force-directed graph, line, parallel coordinates, pie, scatter plot, reorderable matrix, sunburst, treemap, word cloud	√	X
[12]	2020	CNN, YOLO	Object detection and classification	Candlestick charts	√	X
[13]	2020	CNN	Chart type classification	Area, bar, line, map, pareto, pie, radar, scatter, table, venn charts	√	X
[14]	2019	CNN, VGG	Chart type classification	Area, bar, line, map, pareto, pie, radar, scatter, table, venn charts	√	X
[15]	2018	AlexNet, GoogleNet, VGG, ResNet	Chart type classification	Bar, line, pie, radar, scatter	√	X
[16]	2017	SVM, CNN, AlexNet	Recovering visual encodings from chart images	Area, bar, line, scatter plots	√	X
[17]	2016	SVM, CNN, DBN	Chart type classification	Bar, flowchart, line, pie, scatter plot	√	X
Proposed Approach		CNN	Multi-label classification of charts Shape (Pie vs Donut) Dimension (2D vs 3D)	Pie and donut charts	√	√

Some previous studies [3, 18] have been conducted to interpret chart images using image processing methods and text recognition techniques such as optical character recognition. They applied some image processing methods to locate the chart elements (i.e., title, legend, and text regions) to extract data including in a chart.

Our study differs from the studies aforementioned here in several respects. This is the first study that applies a multi-label learning approach to automatically and jointly classify charts in terms of two aspects: shape (pie or donut) and dimension (2D and 3D). Our study is also original in that it compares two alternative MLC approaches combined with CNN on classifying pie charts in terms of accuracy: binary-class-based convolutional neural networks (BCNN) and multi-class-based convolutional neural networks (MCNN).

3. Methodology

A pie chart is a circular graphic that shows the relative sizes of elements to one another and to the whole. Currently, pie charts have been created as an image object, and thus, their visual properties can be only understandable by humans, but not by machines. However, recently, there is an increasing need for machines automatically classifying pie chart images. Furthermore, visually impaired people cannot classify these graphics. To overcome these limitations, in this study, we developed convolutional neural network (CNN) models to automatically classify pie charts according to their properties.

There are various representations and shapes of pie charts. This research is mainly focused on the processing of two chart shapes (pie and donut), as well as chart images in 2D and 3D formats. Since multiple class labels are assigned to the same chart image, this task requires a multiple-label learning approach.

3.1. Multi-label learning

Multi-label learning (MLL) is the task of learning from data samples that are represented by a single feature vector and its associated multiple labels. Multi-label classification (MLC) is one of the main multi-label learning paradigms. MLC is concerned with assigning a data instance to a set of categories or classes, meaning that each instance belongs simultaneously to multiple classes. MLC is a more challenging task than the standard single-label classification because output class label spaces are more complex. Moreover, annotating an object with more than one label is more difficult than annotating it with a single label.

The existing multi-label learning approaches can be divided into two main categories [19]: (i) algorithm adaptation approach and (ii) problem transformation approach. In the *algorithm adaptation approach*, an existing machine learning algorithm is adapted, extended, or customized for the task of MLC. In the *problem transformation approach*, a multi-label classification problem is transformed into more than one single-label classification problems. *Label powerset* (LP) and *binary relevance* (BR) are two of the well-known approaches in this type of multi-label classification. LP converts the multi-label data into multi-class form by considering all unique label combinations. BR transforms an MLC problem into a set of binary classification problems, depending on the one-against-all strategy. In this study, these two MLC methods were compared to each other in terms of accuracy to determine the best one for pie chart classification.

3.2. Convolutional neural networks

Our solution involves classifying pie charts by using deep learning, especially convolutional neural networks (CNN). CNN is a kind of deep neural network that is typically formed by multiple layers of convolutional operations with a filtering process followed by fully connected layers. CNN is usually applied to image data for image classification since it is a multi-layer structured deep learning model architecture designed for two-dimensional image analysis that can capture complex relationships among image pixels by reducing the required number of features [20].

The main operation of CNNs is "*convolution*". Convolution is basically mathematical operations, applying a function on the output of the previous layer. In a convolutional layer, a kernel or filter moves over the image and extracts feature maps that contain features of an image. Here, some image processing filters can also be applied to the images with convolution operations, such as edge detection, Sobel filtering, and noise reduction filters. In CNNs, important properties of the given data are extracted with convolution operations and pooling layers. The neural network model learns the appropriate filter matrices. These matrices are later used for extracting the important properties of the input data and making them ready to be processed in a dense layer. After the main features are extracted with convolution and pooling operations, a traditional artificial neural network is used with that data, called a dense layer or fully-connected layer. This final layer can have some hidden layers and has an output layer for giving the prediction result.

3.3. Proposed approach

This study compares two multi-label learning approaches to classify pie charts: *binary-class-based convolutional neural networks* (BCNN) and *multi-class-based convolutional neural networks* (MCNN).

Binary-class-based convolutional neural networks (BCNN): Based on the binary relevance approach, the multi-label classification problem is converted into a series of binary classification problems. A separate classification model is generated with a CNN architecture for each class label. For a given unseen data instance *x*, BCNN predicts its associated label set *y* by querying it on each independent binary classifier and then combing all labels together. Although the BCNN strategy may be a practical approach, a common disadvantage is that it disregards the relationship between target labels since each binary model is independent of the other.

Multi-class-based convolutional neural networks (MCNN): Based on the label powerset approach, the multi-label classification problem is converted into a multi-class classification problem. It considers each combination of class labels as if it was a new label. A multi-class classifier is built with a CNN architecture. For a given unseen data instance *x*, MCNN predicts its associated label set *y* by using this model. Although the MCNN strategy may be a practical approach, it increases computational complexity as the count of labels in the data increases, and the training dataset is large.

3.4. Proposed architecture

In this section, the proposed CNN model and its properties are discussed. Table 2 shows the general layer structure of the proposed CNN model. The architecture of the CNN model consists of convolution, pooling, flatten, dropout, and dense layers. The architecture in this work utilizes convolution layers and max-pooling layers on determining distinctive features of the charts before feeding them to the neural network. Dropout layers are also used in the feature extraction part.

Table 2. *Proposed CNN model.*

Layer Type	Description
Conv2D	Kernel (3 x 3 x 1 x 64) Bias (64) Activation = ReLu Filters = 64 Kernel_size = 3, 3
MaxPooling2D	
Conv2D	Kernel (3 x 3 x 64 x 128) Bias (128) Activation = ReLu Filters = 128 Kernel_size = 3, 3
MaxPooling2D	
Conv2D	Kernel (3 x 3 x 128 x 256) Bias (256) Activation = ReLu Filters = 256 Kernel_size = 3, 3
Flatten	
Dense	Kernel (173056 x 512) Bias (512) Activation = ReLu Units = 512
Dropout	Rate = 0.2
Dense	Kernel (512 x 256) Bias (256) Activation = ReLu Units = 256
Dropout	Rate = 0.2
Dense	Kernel (256 x 4) Bias (4) Activation = Softmax Units = 4

Convolution layers are used to extract the features (characteristics) of the input chart image data such as color, contour, and types. The three convolution layers in the proposed architecture have 64, 128, and 256 filters, respectively. The kernel size (3, 3) is kept the same for all convolutional layers.

Pooling is a part of the feature extraction process. They are generally utilized for shrinking the spatial size of the data while maintaining its main properties. Pooling operation provides prevention of overfitting in CNNs by downsampling the input images. As can be seen in Table 2, we utilized the max-pooling technique in the model.

Flatten is used to convert the matrix values to a one-dimensional array to pass it to the fully-connected layer. In other words, the output of the convolutional layer is converted into a single vector before it is given as input to the dense layer. One key component of CNN is the loss function. The loss method basically calculates the difference between the true label and the label predicted by the trained model. In the proposed CNN model, the sparse categorical cross-entropy function was applied as a loss function. This function is implemented in Keras and it is well-known for image classification problems. Another important parameter of CNN is the optimizer. The Adam optimizer was used for minimizing loss value. This optimizer is a well-known and successful example of a gradient descent algorithm. The learning rate of the optimizer was set to 0.001, which is suggested as a default value.

The *dropout* method randomly eliminates some outputs from a layer in the network during the training phase. Dropout is useful to prevent overfitting like pooling. The proposed architecture utilizes dropouts after hidden layers in the fully-connected (dense) layer. The best dropout rate was found as 20% in this model.

A *dense* (or *fully-connected*) layer runs as a traditional multiple layer perceptron neural network. An important parameter in neural networks is batch size. We did not determine a specific batch size in the proposed architecture, so the batch size is the same as the count of training samples. *Rectified linear unit* (ReLU) is usually utilized in dense layers of neural networks as an activation method. In the proposed architecture, ReLU is utilized in both convolution layers and fully connected layers. ReLU produces zero for negative inputs, and the input value goes to output without any change if it is a positive number. In the last dense layer, the softmax function is commonly used to normalize output scores over multiple categories. This function accepts a list of real numbers as inputs and turns them into probability percentages. Output matrix size of the softmax layer must have the same value as the number of labels in the training data. Each label has a percentage ratio and the sum of all must be 100%. The label with the highest ratio is determined as the prediction output for a given input data instance.

4. Results

The aim of this study is to build a deep learning model that correctly classifies pie charts. A CNN architecture was designed to build a classifier. The Python programming language was used when constructing the model. Some packages, libraries, and tools were used during the data manipulation and training stages such as Numpy, Pandas, Matplotlib, Scikit-learn, Keras, and Tensorflow. Keras is a comprehensive deep learning framework and has various deep learning functionalities. TensorFlow is an open-source machine learning platform that includes many operations for creating many different predictive models.

For experimenting with the model, a simple computer was used for training the CNN model and making benchmark tests. The hardware that was used in this study is a PC with Windows 10 64 bit operating system and with the Intel Core i7-8750H CPU with 2.20 GHz and 16 GB RAM. We also used the CUDA toolkit, which is a GPU accelerated library for increasing the training speed of neural network models with parallel computation and fast matrix multiplication operations.

The performances of the models were assessed in terms of accuracy, recall, precision, f-measure, and confusion matrix. *Accuracy* is the ratio of the instances, which are correctly predicted by the model, to the total size of the test dataset. The *precision* metric states how many of the values estimated as positive are actually positive. The *recall* is a metric that indicates how much of the test instances that are required to estimate as positive, we estimated as positive. *F-measure* is the harmonic mean of the recall and precision values.

4.1. Dataset description

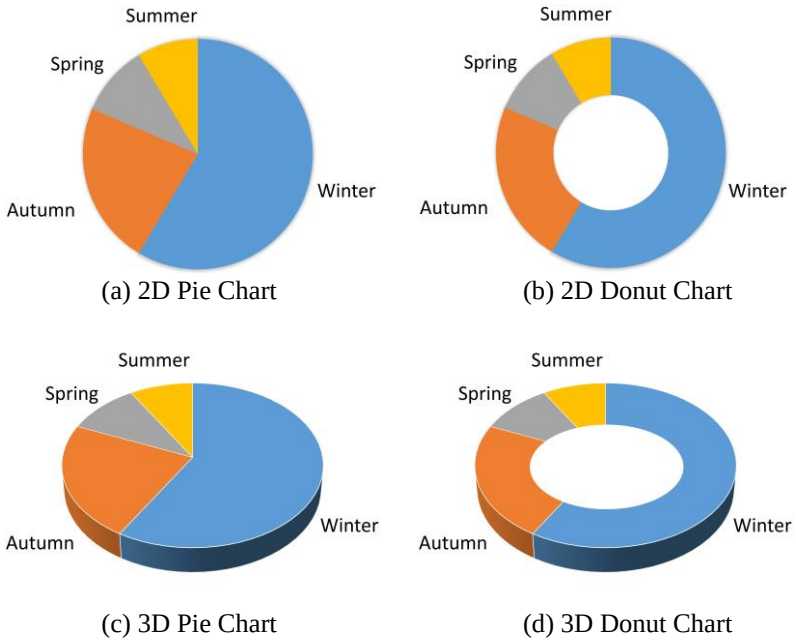
For making accurate predictions on real-world images, training data should have pie charts with different properties and features. Since a specific image repository that is suitable for the purpose of the study is not available, pie chart images in the training dataset were generated by using a computer script. We used the ChartDirector v6 chart drawing library in Python to create the training data. Chart styles (i.e., title, color, legends, and the number of slices) were designed randomly by using different visualization methods available in the library. The sizes of the pie charts in the outlining border were also designed randomly within a specified range.

The size of the image file is an important factor in the classification task. In this study, all the chart image samples were resized to 360 × 280 pixels since neural networks accept fixed inputs. Here, it is important that the converting operation preserves its aspect ratio in a chart. Changing the aspect ratio of an image can cause distortions and noises. Additionally, images with large sizes do not improve the resulting accuracy ratio. Besides, this causes longer training times (more epochs for training the neural network model) because the input vector size would be larger. Very small images can reduce classification accuracies because it causes loss in information.

With the chart generator, we created 400 pie chart image samples for each class that means 1600 images for training in total. Thereby, the training data is well-balanced for all the classes because it includes an equal number of instances from each class. The test dataset used in this study contains real-world pie charts which were collected from the Google Image Search engine. We gathered 50 images for each chart type, so there are 200 images in total in the test dataset. They differ in size and have different styles such as with or without title, legend, and marker.

In the data preprocessing phase, chart images were transformed to a greyscale palette since slices with different colors have no effect on the outcome of the classification. In other words, colored images store redundant data since we do not classify using colors. In addition, converting color images to grayscale images also reduces the training time since each pixel in greyscale images holds a single value, instead of three (RGB).

In this study, two different datasets were generated, one for the BCNN approach and one for the MCNN approach. For the BCNN approach, the training dataset was organized to be involved two target label attributes: one contains the labels of "2D" and "3D", and the other one includes the labels of "Pie" and "Donut". On the other hand, for the MCNN approach, the training dataset was organized to be involved a single target label attribute, which contains the labels of "2D Pie", "2D Donut", "3D Pie", and "3D Donut". Figure 2 shows the types of charts.



(a) 2D Pie Chart (b) 2D Donut Chart

(c) 3D Pie Chart (d) 3D Donut Chart

Figure 1. *Types of charts.*

4.2. Results obtained by the MCNN approach

The MCNN approach achieved 85% of accuracy on average on four class labels: "2D Pie", "2D Donut", "3D Pie", and "3D Donut". During the model is trained, accuracy started to converge to this test score after 25 - 30 epochs and the training loss also converged to near zero. Figure 2 shows the confusion matrix that was generated by the MCNN approach. According to the matrix, it is possible to conclude that the classifier usually had no difficulty in distinguishing chart types. For example, 43 of 50 “2D Donut” charts were correctly predicted; however, only 7 of them were misclassified by the constructed model. As can be seen in the confusion matrix, three test samples were incorrectly classified as a “3D Donut”, instead of “2D Donut”. Although chart types were identified well with high accuracy rates, “3D Pie” and “2D Pie” classes were slightly confused by the algorithm during prediction. As can be seen from the matrix, the best result on the real-world images was obtained for the "2D Pie" label with an accuracy of 98%.

		Predicted Classes			
	Class	**2D Donut**	**2D Pie**	**3D Donut**	**3D Pie**
Actual Classes	**2D Donut**	43	2	3	2
	2D Pie	0	49	0	1
	3D Donut	5	2	35	8
	3D Pie	0	6	1	43

Figure 2. *Confusion matrix that was generated by the MCNN approach.*

For further investigation of the proposed model performance, Table 3 shows the precision, recall, and f-measure values that were observed for each class. While the precision scores are ranging between approximately 0.8 to 0.9 for all classes, the recall scores are distributed between 0.7 and 0.98. Although all the results are very promising, the precision scores for the donut charts are higher than the scores for the pie chart. For instance, the precision value for the "2D Donut" chart is 0.8958, whereas it is the value of 0.8305 for the "2D Pie". On the other hand, the recall scores for the pie charts are higher compared to the donut charts. For instance, the recall value for the "3D Pie" chart is 0.86, whereas it is 0.7 for the "3D Donut". Among all the labels, MCNN achieved the best f-measure value (0.8990) on the "2D Pie" class. When the results given in Table 3 are considered as a whole, it can be deduced that the model built by MCNN has a good generalization ability to classify pie and donut charts in both 2D and 3D formats, so it can be effectively utilized to predict them well.

Table 3. *Performance values that were obtained by the MCNN approach.*

Chart Type	Precision	Recall	F-Measure
2D Donut	0.8958	0.8600	0.8775
2D Pie	0.8305	0.9800	0.8990
3D Donut	0.8974	0.7000	0.7865
3D Pie	0.7962	0.8600	0.8269

4.3. Results obtained by the BCNN approach

In this experiment, the BCNN approach was tested on the same dataset. The BR approach uses two different models at the same time. The first model predicts the dimension of the chart, whether if it is a *2D* or *3D* chart. The second model predicts the shape of the chart, whether it is a *pie* or *donut*. The average accuracy of these two models is considered as the final performance of the approach.

Figure 3 shows the confusion matrices that were generated by the BCNN approach. While the first model achieved 85% of accuracy on the prediction of chart dimension (2D or 3D), the second model reached 87% of accuracy on the prediction of chart shape (pie or donut). Thus, the average accuracy of these two models is 86%. It was observed from the experiment that the BCNN approach could distinguish the shape of the pie chart more correctly compared to the donut. Likewise, it achieved higher accuracy when distinguishing 2D charts compared to 3D charts. For example, 96% of 2D charts were classified correctly; nevertheless, only 2 out of 50 2D charts were predicted incorrectly. It was observed from the matrices that the models usually had no difficulty in identifying the charts.

		Predicted Classes	
	Class	**2D**	**3D**
Actual Classes	**2D**	48	2
	3D	13	37

		Predicted Classes	
	Class	**Pie**	**Donut**
Actual Classes	**Pie**	48	2
	Donut	11	39

Figure 3. *Confusion matrices that were generated by the BCNN approach.*

Table 4 shows the precision, recall, and f-measure values that were obtained by the BCNN approach. While the precision scores are ranging between approximately 0.78 to 0.95 for all classes, the recall scores are distributed between 0.74 and 0.96. Although all the precision scores are very promising, the best precision value (0.9512) was obtained for the pie shape. Among all the labels, BCNN achieved the best f-measure value (0.8807) on the donut-shaped charts. When the results given in Table 4 are considered as a whole, it can be deduced that the models constructed by BCNN have a good generalization ability to classify pie and donut charts in both 2D and 3D formats.

Table 4. *Performance values that were obtained by the BCNN approach.*

Chart Type	Precision	Recall	F-Measure
Donut	0.8135	0.9600	0.8807
Pie	0.9512	0.7800	0.8571
2D	0.7868	0.9600	0.8648
3D	0.9487	0.7400	0.8314

4.4. Comparison of the MCNN and BCNN approaches

Figure 4 shows the comparison of the performances of the MCNN and BCNN approaches. According to the results, BCNN (86%) slightly outperformed MCNN (85%) in terms of classification accuracy. This means that the model constructed by BCNN has a better chance of correctly classifying charts. Nevertheless, since the difference in their performances is small, these two approaches can be alternatively used in real-world applications.

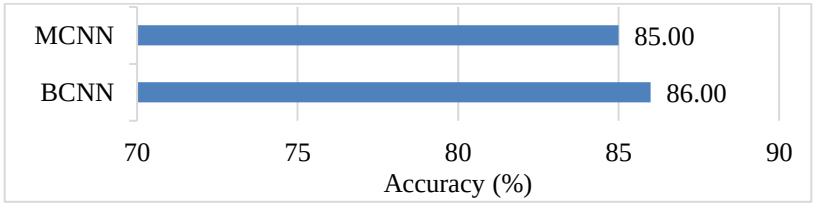


Figure 4. *Comparison of the MCNN and BCNN approaches.*

5. Conclusion

A pie chart is a useful graphic that is divided into slices to illustrate the proportions of elements to the whole. The properties of pie charts cannot be directly noticed by machines since they are usually in an image format. To make a pie chart classifiable by machines, this paper proposes a novel solution using deep learning techniques. A CNN architecture is designed and proposed for the pie chart classification task. While the previous works on pie chart classification use the standard (single-label) classification, our study proposes a multi-label classification approach on pie charts. This study is original in that it jointly classifies charts in terms of two respects: *shape* (pie or donut) and *dimension* (2D or 3D). This is the first study that compares two multi-label learning approaches to classify pie charts: binary-class-based convolutional neural networks (BCNN) and multi-class-based convolutional neural networks (MCNN). In the experimental studies, these two approaches (BCNN and MCNN) were tested and compared on the real-world pie chart images. The results showed that the BCNN model achieved 86% accuracy, while the MCNN model reached 85% accuracy.

As future work, the study can be further improved in several respects. In addition to visual properties, more information about pie chart images can also be given to visually impaired people by obtaining textual descriptions such as the values of the slices, the title of the chart, and the values reported in the legends. Moreover, similar studies can be conducted on other chart types such as bar charts.

Declaration of Interest

The authors declare that there is no conflict of interest.

References

[1] F. Bajic and J. Job, “Chart classification using Siamese CNN,” Journal of Imaging, vol. 7, no. 11, pp. 1-18, 2021.

[2] P. Mishra, S. Kumar, and M. K. Chaube, “ChartFuse: a novel fusion method for chart classification using heterogeneous microstructures,”, Multimedia Tools and Applications, vol. 80, no. 7, pp. 10417-10439, 2021.

[3] J. Fu, B. Zhu, W. Cui, S. Ge, Y. Wang, H. Zhang, H. Huang, Y. Tang, D. Zhang, and X. Ma. "Chartem: reviving chart images with data embedding," IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 2, pp. 337-346, 2021.

[4] C-S. Cheng, Y. Ho, and T-C. Chiu, "End-to-End control chart pattern classification using a 1D convolutional neural network and transfer learning," Processes, vol. 9, no. 9, pp. 1-26, 2021.

[5] Y. Zheng, Y-W. Si, and R. Wong, "Feature extraction for chart pattern classification in financial time series," Knowledge and Information Systems, vol. 63, no. 7, pp. 1807-1848, 2021.

[6] C. Sohn, H. Choi, K. Kim, J. Park, and J. Noh, "Line chart understanding with convolutional neural network," Electronics, vol. 10, no. 6, pp. 1-17, 2021.

[7] F. Zhou, Y. Zhao, W. Chen, Y. Tan, Y. Xu, Y. Chen, C. Liu, and Y. Zhao, "Reverse-engineering bar charts using neural networks," Journal of Visualization, vol. 24, no. 2, pp. 419-435, 2021.

[8] R. Ünlü, "A robust data simulation technique to improve early detection performance of a classifier in control chart pattern recognition systems," Information Sciences, vol. 548, pp. 18-36, 2021.

[9] M. Zaman, and A. Hassan, "Fuzzy heuristics and decision tree for classification of statistical feature-based control chart patterns," Symmetry, vol. 13, no. 1, pp. 1-12, 2021.

[10] M. Siper, K. Makinen, and R. Kanan, "TABot - a distributed deep learning framework for classifying price chart images," Advanced Computing, vol. 1367, pp. 465-473, 2021.

[11] T. Araujo, P. Chagas, J. Alves, C. Santos, B. Santos, and B. Meiguins, "A real-world approach on the problem of chart recognition using classification, detection and perspective correction," Sensors, vol. 20, no. 16, pp. 1-21, 2020.

[12] S. Birogul, G. Temür, and U. Kose, "YOLO object recognition algorithm and buy-sell decision model over 2D candlestick charts," IEEE Access, vol. 8, pp. 91894-91915, 2020.

[13] F. Bajic, J. Job, and K. Nenadic, "Data visualization classification using simple convolutional neural network model," International Journal of Electrical and Computer Engineering Systems, vol. 11, no. 1, pp. 43-51, 2020.

[14] F. Bajic, J. Job, and K. Nenadic, "Chart classification using simplified VGG model," In International Conference on Systems Signals and Image Processing, Osijek, Croatia, 2019, pp. 229-233.

[15] W. Dai, M. Wang, Z. Niu, and J. Zhang, "Chart decoder: generating textual and numeric information from chart images automatically," Journal of Visual Languages & Computing, vol. 48, pp. 101-109, 2018.

[16] J. Poco and J. Heer, "Reverse-engineering visualizations: recovering visual encodings from chart images," Computer Graphics, vol. 36, no. 3, pp. 353-363, 2017.

[17] B. Tang, X. Liu, J. Lei, M. Song, D. Tao, S. Sun, and F. Dong, "DeepChart: combining deep convolutional networks and deep belief networks in chart classification," Signal Process, vol. 124, pp. 156-161, 2016.

[18] J. Shtok, S. Harary, O. Azulai, A. R. Goldfarb, A. Arbelle, and L. Karlinsky, "CHARTER: heatmap-based multi-type chart data extraction," In Document Intelligence Workshop at KDD, 2021, pp. 1-5.

[19] M-L. Zhang and Z-H. Zhou, "A review on multi-label learning algorithms," IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 8, pp. 1819-1837, 2014.

[20] S. H. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of fully connected layers on performance of convolutional neural networks for image classification," Neurocomputing, vol. 378, pp. 112-119, 2020.