Evaluation of Shallow Convolutional Neural Network in Open-World Chart Image Classification

Filip Bajić[1,*], Marija Habijan[2] and Krešimir Nenadić[2]
[1]University Computing Centre, University of Zagreb, Zagreb, 10000, Croatia
[2]Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, Osijek, 31000, Croatia
E-mail: filip.bajic@srce.hr, marija.habijan@ferit.hr, kresimir.nenadic@srce.hr
*Corresponding author

Keywords: data visualization, machine learning, chart image, neural network, pattern recognition

Received: January 23, 2024

Data's role is pivotal in the era of internet technologies, but unstructured data poses comprehension challenges. Data visualizations like charts have emerged as crucial tools for condensing complex information. Classifying charts and applying various processing techniques are vital to interpreting visual data. Traditional chart image classification methods rely on predefined rules and have limited accuracy. The advent of support vector machines (SVMs) and convolutional neural networks (CNNs) significantly improved the accuracy of these methods. This research evaluates our previously introduced Shallow convolutional neural network (SCNN) architecture for chart image classification, comprising four convolutional layers, two max-pooling layers, and one fully-connected layer. The network achieves state-of-the-art results, requiring smaller datasets and reduced computational resources. When two networks are combined into Siamese SCNN (SSCNN), emphasizing generalization, it achieves high accuracy with small datasets and excels in open-set classification. The evaluation process encompasses the utilization of six publicly available datasets.

Povzetek: Raziskava uvaja arhitekturo plitkega konvolucijskega nevronskega omrežja (SCNN) za klasifikacijo slik grafikonov.

1 Introduction

In the contemporary digital landscape characterized by advanced internet technologies, the crucial role of data and information is evident. Despite the internet's capacity to process vast amounts of data, challenges arise with accumulating unstructured data, especially when presented in tables that demand significant mental effort for comprehension. Statistical information, often in numerical formats, can be intricate to interpret, leading to difficulties distinguishing crucial from less relevant data[1].

Data visualization, widely employed in mathematics, statistics, analytics, and pattern identification scenarios, faces accessibility challenges. Search engines struggle to include results from visualizations, and blind or visually impaired individuals encounter difficulties due to a lack of awareness about accessibility guidelines. Digital documents often lack essential elements, hindering comprehensive understanding for those relying on screen readers. Ongoing research aims to enhance accessibility, focusing on improved classification and interpretation of information in visual representations [2].

Various tools and methods, particularly data visualizations like charts and diagrams, have emerged to address these challenges. These visual representations condense complex data, aiding in transmission, understanding, and decision-making. Customized charts tailored to specific data types enhance structural clarity. The initial step in interpreting visualizations involves chart classification, followed by processing techniques like data export algorithms and optical character recognition systems [3, 4, 5].

This work comprehensively evaluates the SCNN architecture for chart image classification, utilizing six publicly available datasets throughout training and evaluation. The study explores the impact of dataset quality and quantity on results, addressing open classification through traditional SCNN and Siamese architecture, yielding positive results.

2 Related research

Unlike tables, charts enhance comprehension and simplify interpretation. Various digital tools, such as Microsoft Excel, Matlab, D3, Plotly, or manual methods, can be used for chart creation. However, challenges arise when charts are stored or digitized as a single visual entity, leading to the loss of structural information. As charts find extensive applications, algorithms have been developed to retrieve and process information from stored images. Over the years, researchers have used different methods to extract information from charts, which can be categorized into four groups: methods that use custom algorithms, model-

based methods, machine-learning (ML) based methods, and neural network-based methods. An extended review of research in this field is available in [3].

Custom algorithms involve analyzing and extracting features of the chart image at the pixel and graphic symbol level and combining it with different image preprocessing techniques. The primary task of image preprocessing is to prepare the image for future analysis and feature extraction. The most basic types of image preprocessing used for chart extraction and visualization include image resolution normalization, image color space normalization, and image noise reduction [2, 6, 7]. Advanced types of image preprocessing, such as edge detection, vectorization, and segmentation, enable separating textual and graphic image elements [1, 2, 8, 9, 10, 11, 12]. Edge detection distinguishes sudden changes in image brightness, while vectorization allows the conversion of a raster image into a vector image to extract graphic shapes. The segmentation process reduces the information in the image, making it possible to distinguish between essential and non-essential elements needed for chart image classification.

Model-based chart image classification is based on handcrafted models created for each chart-type separately. Defining the critical textual and graphic elements that a specific chart-type should contain is necessary within the model. The chart image will not be successfully classified if some of the essential elements are omitted or if their location in the image space differs from the location defined by the model [13, 14].

Machine-learning methods such as SVMs are commonly used for regression and classification. The features created using the previously mentioned methods are often used as input values of SVM. SVMs achieve successful results even with small datasets; however, setting a margin is challenging when classes share features [2, 12, 15, 16, 17, 18].

Since 2015, current research has employed CNNs for chart image classification, and a comparison of traditional methods and CNN architectures [41] showed a visible results improvement of 20%. Researchers use various CNN architectures, the most common ones being LeNet, AlexNet, VGG, GoogLeNet, ResNet, Inception, and MobileNet. A comparison of these architectures is available in [8, 42, 36, 43], and the results show a difference of up to 5%. Nevertheless, comparing the results is challenging since the authors use different variables and datasets. Even if some authors use existing, out-of-the-box solutions, they must adjust the input parameters to suit their needs and the dataset used. The dataset is the most crucial input variable for comparing different methods. Since all the datasets used are qualitatively and quantitatively different, the reported numbers should be considered additional information rather than a reference point to achieve specific results. Moreover, CNN can be used alone or in combination with SVM, where CNN is used for feature extraction, and SVM is used as a classifier. Either way, it has been shown that using CNN produces very high classification accuracy, as seen in Table 1. Several noteworthy articles from Table 1 warrant attention.

Foremost among these is ”ReVision,” acknowledged as the most cited scientific paper in the field, despite its 2011 publication date [16]. This seminal work remains state-of-the-art, elucidating a comprehensive process for classifying chart images. The methodology encompasses leveraging low-level image features, incorporating textual information to enhance classification outcomes, extracting data from bar and pie charts, and applying perceptually based design principles to reimagine data visualizations. The authors report a notable classification accuracy of 0.81 across ten chart-types.

Another notable contribution is utilizing a modified LeNet CNN to classify input images into 11 chart-types [23]. Assisted by a custom dataset, the average classification accuracy achieved is 0.89. The article provides intricate details about the dataset, including the average classification accuracy for each chart-type. Furthermore, it offers a comprehensive model description, experimental setup, and a comparative analysis involving classic LeNet, pre-trained LeNet, and the modified LeNet model.

Addressing the accessibility of data visualizations for visually impaired users, [1] introduced a fully automated system titled ”Visualizing for the Non-Visual.” This system adeptly classifies input images into ten chart-types, detects and categorizes graphical and textual elements, extracts shapes into vector format, and retrieves data from three chart-types. The authors achieved an average classification accuracy of 0.96 using ResNet. The article provides a detailed dataset overview and conducts a comparative analysis with other systems, including ”Reverse-Engineering Visualizations” [2], ”ChartSense” [10], and ”ReVision” [16]. Incorporating multiple CNNs for a specific task led to attaining state-of-the-art results.

”ChartDETR” [40] represents the latest advancement in the field, employing a transformer-based multi-shape detector for localizing keypoints at the corners of regular shapes, thereby reconstructing multiple data elements within a single chart image. The proposed methodology introduces query groups in set prediction, enabling the simultaneous prediction of all data element shapes and eliminating the necessity for subsequent postprocessing. This distinctive attribute positions ”ChartDETR” as a unified framework capable of accommodating various chart-types without necessitating alterations to the network architecture, thereby adeptly detecting data elements with diverse shapes. The method undergoes evaluation on three datasets, demonstrating competitive results across three chart-types and achieving a classification accuracy of 0.98.

3 Methods

This section explains two used methodologies involving an SCNN and an SSCNN. Both model’s architectures are available in our previously published article [39].

Authors	Year	Method	Number of chart-types	Accuracy	Dataset
Redeke [11]	2001	Custom algorithm	2	0.83	small
Mishchenko and Vassilieva [13]	2011	Model-based	5	0.92	small
Mishchenko and Vassilieva [14]	2011	Model-based	5	0.92	small
Savva et al. [16]	2011	SVM	10	0.81	medium
Gao et al. [12]	2012	Custom algorithm, SVM	3	0.97	small
Karthikeyani and Nagarajan [19]	2012	Custom algorithm, SVM	8	0.69 – 0.77	small
Cheng et al. [20]	2013	Custom algorithm	3	0.96	medium
Liu et al. [21]	2015	CNN	5	0.75	medium
Siegel et al. [15]	2016	CNN	7	0.84 – 0.86	large
Poco and Heer [2]	2017	CNN, SVM	10	0.94	medium
Junior et al. [22]	2017	CNN	10	0.70	medium
Amara et al. [23]	2017	CNN	11	0.89	medium
Jung et al. [10]	2017	CNN	10	0.91	medium
Battle et al. [24]	2018	Custom algorithm	24	0.83 – 0.94	large
Lin et al. [25]	2018	CNN	2	0.89 – 0.93	small
Dai et al. [8]	2018	CNN	5	0.99	large
Choi et al. [1]	2019	CNN	10	0.96	medium
Jobin et al. [26]	2019	CNN	28	0.88 – 0.93	large
Liu et al. [27]	2019	CNN	2	0.96	large
Davila et al. [28]	2019	CNN	7	0.88 – 0.99	large
Bajić et al. [4]	2019	CNN	10	0.81	medium
Kaur and Kiesel [29]	2020	CNN	14	0.92	medium
Bajić et al. [6]	2020	CNN	10	0.73 – 0.89	medium
Kosemen and Birant [30]	2020	CNN	1	0.93	medium
Ishihara et al. [31]	2020	CNN	1	0.97	medium
Zhu et al. [32]	2020	Custom algorithm, SVM	5	0.98	large
Huang [33]	2020	CNN	16	0.4 – 0.7	medium
Davila et al. [34]	2020	CNN	15	0.93 – 1.00	large
Dadhich et al. [35]	2021	CNN	1	0.85	medium
Thiyam et al. [36]	2021	CNN	9	0.87 – 0.98	large
Bajić and Job [37]	2021	CNN	7	1.00	medium
Thiyam et al. [38]	2021	CNN	25	0.70 – 0.90	medium
Bajić et al. [39]	2022	CNN	7	0.99 – 1.00	medium
Xue et al. [40]	2023	CNN	3	0.98	large

Table 1: An overview of research in the area. A variable classification performance value indicates the use of different methods or datasets. Datasets are classified into three categories: small (sum of images used is less than 1,000), medium (sum of images is between 1,000 and 10,000), and large (sum of images is greater than 10,000)

3.1 Used datasets

Data is fundamental in ML and critical for training models and ensuring their efficacy. Several datasets were used to train, validate, and test the SCNN and SSCNN, as seen in Figure 1. The ”ReVision” [16] dataset, created in 2011, served as a primary dataset but was downsized due to unavailability. It is specifically used to test SSCNN and undergoes complete processing with the image preprocessing algorithm presented in [4, 6, 37]. The ICDAR7 [28] dataset, artificially generated using Python Matplotlib, supports training, validating, and testing both SCNN and SSCNN, also processed by the image preprocessing algorithm. The ChartDS [44, 45] dataset, created with Python Plotly, includes high-resolution, labeled chart images, used extensively for training, validating, and testing SCNN and SSCNN, and is publicly available. The Linnaeus [46] dataset, formed from internet-searched images, serves as the H-class and is mildly preprocessed. CIFAR10 [47], another internet-collected dataset, expands the H-class, sharing Linnaeus' preprocessing method. The AT&T DoF [48] dataset, designed for human face recognition, supports SSCNN training with mild preprocessing.

For open-set classification methods, a Mendeley Data repository [49] houses chart images organized similarly to their distribution during method evaluation.

3.2 The architecture of the models

The demand for digital data processing has propelled computing advancements, notably in deep CNNs (DCNNs) for chart image classification. However, the need for a signifi-

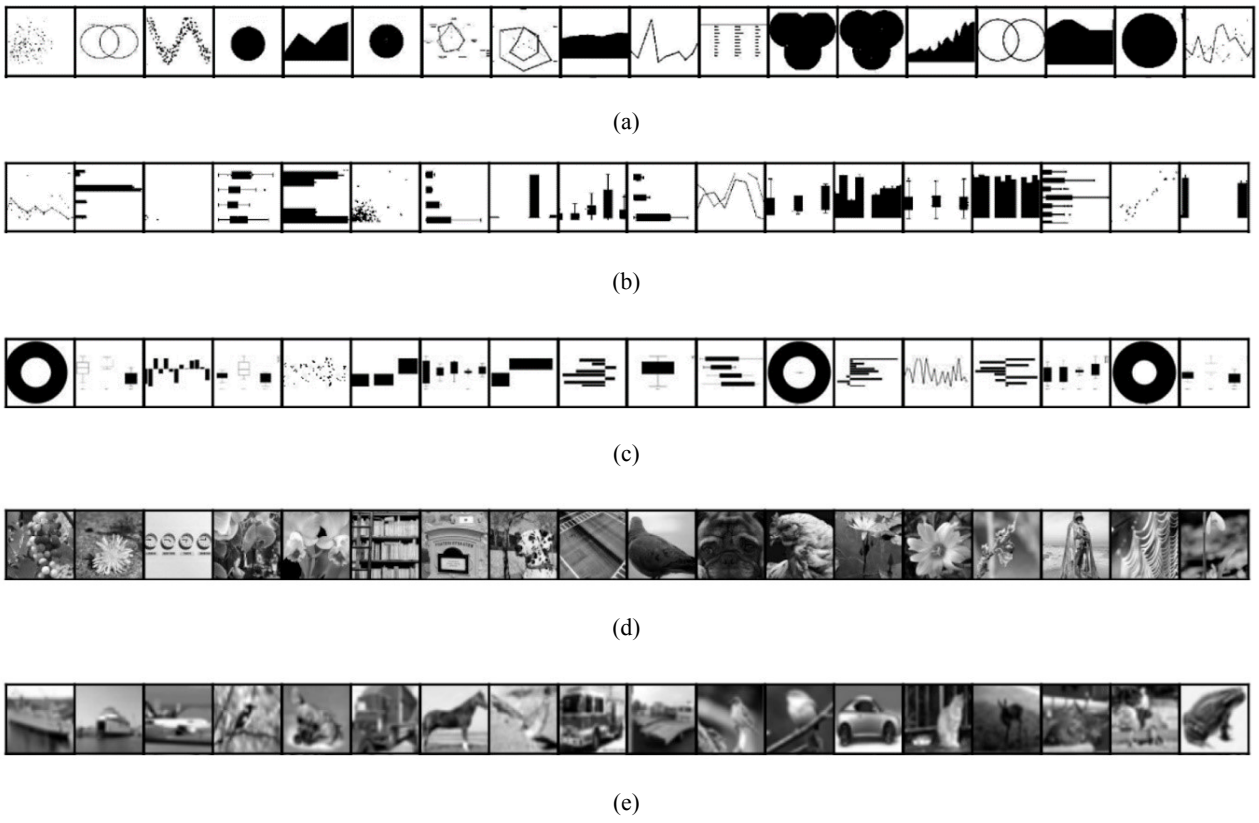


Figure 1: A sample batch of images from different datasets used for training, testing, and validation of SCNN and SSCNN: (a) ReVision, (b) ICDAR7, (c) ChartDS, (d) Linnaeus, and (e) CIFAR10

cant volume of images per chart-type, especially with limited datasets, may lead to overfitting and reduced accuracy. Deepening DCNNs introduces challenges like vanishing or exploding gradients, requiring more computational resources and a larger dataset for training. In response, the SCNN architecture has been developed, differing from traditional CNN models by offering reduced parameters and computational complexity to mitigate resource demands.

3.2.1 An overview of the SCNN architecture

SCNNs [39] represent a subset of neural networks characterized by fewer hidden layers than conventional CNNs. In contrast to DCNNs, SCNNs are tailored to address specific tasks with a minimal hidden layer configuration, making them advantageous in scenarios with limited computational resources or when handling relatively straightforward tasks. While the layer count is a defining characteristic, the performance of a neural network is influenced by various factors, including the number of neurons per layer, the activation function, and the optimization algorithm.

SCNNs offer interpretability benefits, providing a clearer understanding of decision-making processes and identifying crucial input elements. They may exhibit improved generalization and resilience to overfitting compared to deeper networks. The ideal SCNN configuration needs to be more strictly defined; practical considerations dictate a minimal number of hidden layers tailored to the task's complexity. A representative SCNN model [39] includes four convolutional layers, two max-pooling layers, and a fully-connected layer. Input chart images undergo preprocessing, enabling the network to learn intricate patterns. The network's architecture should align with the task's specific requirements.

3.2.2 An overview of the SSCNN architecture

SSCNNs [39], initially designed for signature verification, faced limitations due to substantial computing needs. Recent advancements, like CUDA technology, have accelerated their learning time, making SSCNNs widely applicable for various tasks. Its key features include positive results for unseen classes, shared weights, explainable outcomes, and effectiveness with minimal training images. Although it emphasizes dataset quality over quantity, challenges, like increased computational demands and loss function selection, must be considered during training.

SSCNNs are a specialized subtype of CNNs that use two identical (SCNN) neural networks sharing weights and updated parameters [39]. These models process different input images, producing distinct output vectors for subsequent comparison. The contrastive loss (CL) function, cru-

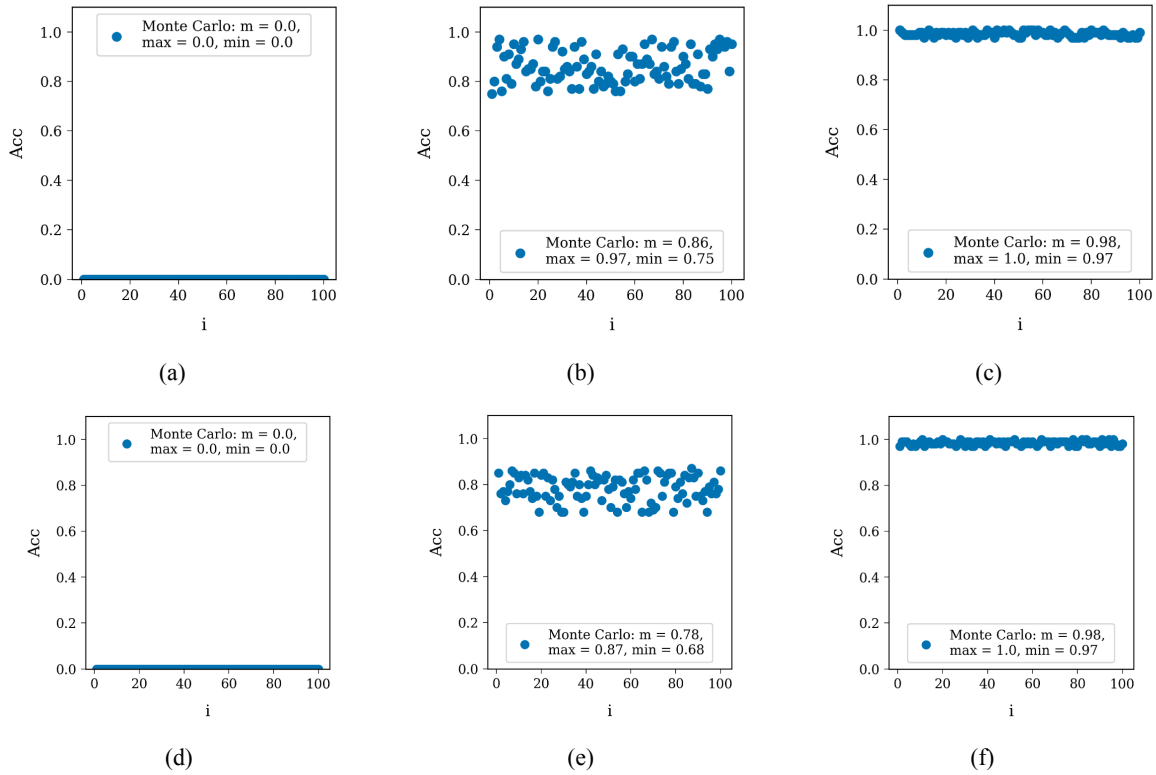


Figure 2: MCCV method shown on the three datasets that participated in the training, from left to right: dataset with 1, 20, and 500 images per chart-type. The first row shows the training process on the ChartDS dataset, and the second row shows the training process on the ICDAR7 dataset. Increasing the dataset size increases the performance of the model.

cial in SSCNNs, calculates the Euclidean distance between two vectors, determining spatial disparity. In SSCNNs, CL gauges the similarity score (SS) between input vectors, with a lower score indicating greater similarity. This approach helps discern similarities and distinctions between input vectors of the same or different classes. Unlike the SCNN model, which provides a probability score, CL yields an SS within the 0 to 1 range. Verification tests with identical images yield an SS of 0, confirming the model's correctness. Conversely, the SS approaches zero for disparate images, signifying the same class, while distinct classes yield a non-zero SS.

4 Evaluation and results

In this section, we discuss the experimental evaluation of the SCNN model on two publicly available datasets, ChartDS and ICDAR7, encompassing various chart-types and styles. The model underwent separate training on both datasets, creating seven models for each dataset with increasing image counts per chart-type in distinct subsets (1, 5, 10, 20, 50, 100, 250, and 500 images per class). Post-training, the models were evaluated on both datasets using five non-utilized testing data subsets (S1 – S5), and the outcomes were summarized. Various learning parameters, such as learning rate, number of epochs, and seed value for randomizing network weights, were adjusted during training to ensure optimal model performance. The learning rate decreased dynamically, and a validation dataset assessed the model's performance. The training spanned 20 epochs, and the seed value randomized network weights, preventing local minima convergence. This randomization influenced the network output, ensuring replicable results with consistent seed values. The random value also played a pivotal role in selecting and partitioning images in training and validation datasets, significantly impacting the model's final result.

The SCNN model's experimental evaluation demonstrated its effectiveness in accurately detecting and classifying various chart-types and styles. Additionally, to enhance the SCNN model's learning, an SSCNN model with similar learning parameters was introduced. The SSCNN model, trained over 100 epochs, utilized an additional dataset, AT&T DoF, containing 40 classes. This supplementary dataset improved the learning of similarities between pairs of images. The experimental evaluation of the SSCNN model differed notably from the SCNN model, providing SS instead of probability values. The SSCNN model employed N-way-K-shot learning, where N is the number of classes and K is the number of samples from each class. In this evaluation, 7-way-1-shot learning was used. The SSCNN model compared known and unknown images, calculating SS for each pair. The dataset's quality proved crucial, with the model matching the unknown

image against all training images to eliminate anomalies. The evaluation comprised 140 pairs of images for one class and 980 for all seven classes, emphasizing the significance of dataset quality over quantity in SSCNN model performance.

4.1 Monte-Carlo

Monte Carlo cross-validation (MCCV) is a method that involves iteratively dividing the training set and validation on the original dataset, potentially including duplicate samples in the validation set. The division ratio between training and validation data varies randomly from 70% to 80% for training and 20% to 30% for validation. Increasing iterations (recommended from 10 to 1,000) reduces uncertainty and model bias. MCCV was repeated 100 times in this model training due to the smaller dataset.

Optimal results for ChartDS were achieved with a random value of 1529; for ICDAR7, it was 635. Figure 2 illustrates the 100-time repeated model training, showing classification accuracy on the validation dataset for datasets with 1, 20, and 500 images per chart-type. Learning was challenging with just one image per chart-type but improved with 20 images, showing oscillation in accuracy, with slightly higher average values for ChartDS. Maximum accuracy was achieved with 500 images per chart-type, regardless of the dataset. Cross-validation and MCCV significantly enhance ML model accuracy and robustness.

4.2 Confusion matrix

The confusion matrix visually presents actual and predicted classes, with the diagonal indicating accurately predicted classes. Standard metrics like Accuracy, Precision, Recall, and F1-score are derived from this matrix across chart-types. Accuracy represents the percentage of correctly classified images, Precision is the proportion of true images correctly classified, Recall represents the proportion of correctly classified true images, and F1-score is the harmonic mean of Precision and Recall. Probability values for each class introduce statistical representations denoted as R1, R2, R3, R4, and R5, representing performance scores based on average probability class values.

4.3 Open-set classification

Open-set classification is a widely recognized challenge in the field of ML. It refers to the ability of a model to adapt to unexpected images that it has not been trained on. This task is relatively simple for humans - we can quickly determine if an object belongs to a particular class just by looking at it. However, this presents a unique computer challenge that has yet to be solved by a single method. In open-set classification, the model is expected to classify images that do not belong to any pre-defined classes as H-class. This prevents unknown images from being classified incorrectly as a known class. There have been several proposed solutions to this challenge, such as the use of a binary SVM classifier [50], the use of an SVM classifier with a defined threshold [51], the use of statistical knowledge about the probability of including or excluding a vector based on a threshold value [52], rejecting specific inputs during network training [53], creating a separate class that will contain all expected unknown classes or creating a separate class that will contain the largest possible number of known images [54, 55], and using Zero/One/Few shot learning [56, 57]. Despite these methods, open-set classification is still a challenging problem.

4.4 Obtained results using SCNN

Figure 3a shows the model's performance on seven subsets belonging to the ChartDS dataset, and Figure 3b shows the model's performance on seven subsets belonging to the ICDAR7 dataset. The models were tested on both datasets. The test results are consistent with MCCV testing on the validation dataset. When one image per chart-type is used for training the model, the classification accuracy is approximately 0. When 20 images per chart-type are used for training, it is approximately 0.85; when 500 images per chart-type are used for training, it is approximately 1. The previously published journal article always used 20 images per chart-type for testing [39]. This identical dataset is labeled S1. The research was extended to five datasets (S1 – S5), i.e., 100 images per chart-type, and minimal deviation of the results is visible. When the model was trained on one dataset and tested on another, a significant drop in classification accuracy was visible, even though the same charts were tested. A manual check of the achieved results shows that the model classifies most images in the H-class.

Figure 3c, shows the H-class classification accuracy, which for the ChartDS dataset follows the Figure 3a, and Figure 3d, shows the H-class classification accuracy, which for the ICDAR7 dataset follows the Figure 3b. The model successfully recognizes seven chart-types from the H-class. When the ICDAR7 dataset is used to test the model trained on the ChartDS dataset, the model achieves significantly worse results than Figure 3d. Although both datasets contain an H-class constructed from the CIFAR10 and Linnaeus datasets, in the case of the ICDAR7 dataset, the model fails to distinguish chart images from H-class images successfully. The reason is that the model, in this case, has an additional class, so the charts should be recognized, but due to the low probability, they are classified in the H-class. Due to the low probability of the model on the chart classes, more than 80% of the images are classified into the H-class. When the ChartDS dataset is used to test the model trained on the ICDAR7 dataset, the model achieves approximate values, as shown in Figure 3b. The structure of ICDAR7 images is unified and significantly different from the H-class; therefore, the model successfully recognizes images belonging to the H-class.

A detailed presentation of the classification performance values for the model trained on the ChartDS dataset and

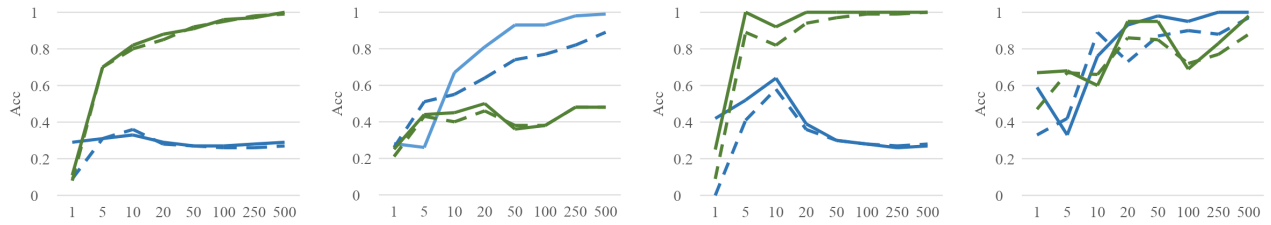


Figure 3: SCNN classification accuracy for seven chart-types with H-class (a,b). Classification accuracy for the H-class only (c,d). The model was trained on ChartDS (a,c). The model was trained on ICDAR7 (b,d). The evaluation results are consistent with MCCV testing on the validation dataset. The blue line represents testing on ICDAR7, while the green line represents testing on ChartDS. A continuous line is for S1, and a dashed line is for S1 - S5.

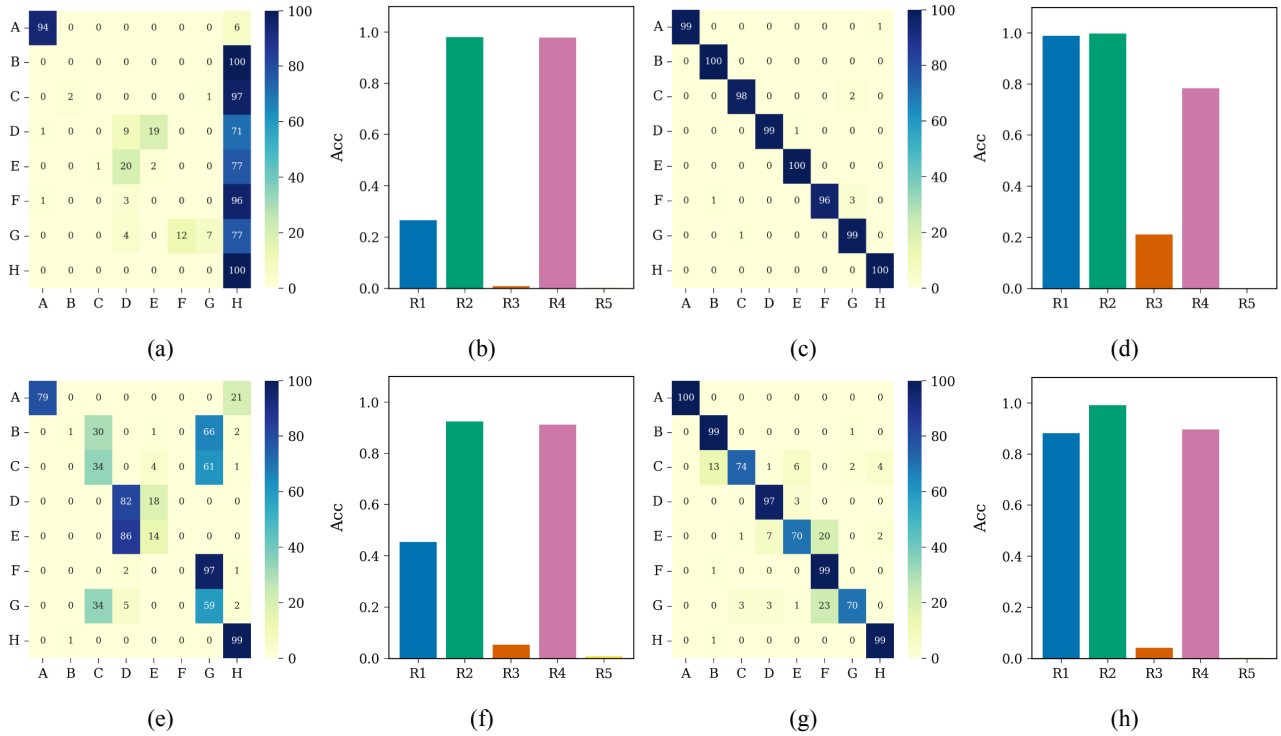


Figure 4: Confusion matrix for the model trained on the ChartDS dataset, and tested on the ICDAR7 dataset (a), for the model trained and tested on the ChartDS (c), for the model trained on the ICDAR7 dataset and tested on the ChartDS (e), and for the model trained and tested on the ICDAR7 dataset (g). The model was trained using 500 images per class and tested using 100 images per class. Statistical analysis of realized probabilities in the confusion matrix (b,d,f,h). A: pie and donut, B: horizontal bar, C: horizontal box, D: line, E: scatter, F: vertical bar, G: vertical box, H: other, R1: actual class, R2: predicted & actual class, R3: actual class & not predicted class, R4: not actual class & predicted class, R5: not predicted & not actual class.

tested on the ICDAR7 dataset is shown by the confusion matrix Figure 4a and Figure 4b. If we were to isolate the H-class, the accuracy would be 0.16, the recall would be 1.00, and the F1-score would be 0.28. Statistical analysis shows that the model achieves a very high probability, but in the case of this dataset, the model fails to distinguish the charts from the H-class. The results are above average when the model is trained on the ChartDS dataset and tested on the same dataset, as shown in Figure 4c. The model achieves a classification accuracy greater than 0.99 and uses a neural network with significantly reduced depth compared to other research in this area, as shown in Table 1.

Additionally, this is the first application of SCNN for open-set classification in chart classification. From the confusion matrix shown in Figure 4c, the H-class's accuracy is 0.99, recall 1.00, and F1-score 1.00. From the statistical analysis shown in Figure 4d, the model predicts where the prediction is expected, and the probability value

is very high. In the case of incorrect predictions, the model achieves responses with high probability (R3), but the prediction probability is still lower than for true predictions (R1, R2). By setting a threshold value, it is possible to make an additional exclusion and thus achieve the maximum accuracy of the H-class.

On the other hand, a detailed representation of the classification accuracy for the model trained on the ICDAR7 dataset and tested on the ChartDS dataset is shown by the confusion matrix, Figure 4e, and Figure 4f. If we were to isolate the H-class, the achieved classification accuracy would be 0.79, recall 0.99, and F1-score 0.88. By comparative analysis with previous testing, the model recognizes three types of charts (A, D, G). Figure 4f shows the statistical analysis of the confusion matrix shown in Figure 4e.

As in the previous testing, the model achieves a high probability but fails to recognize specific charts. When the model was trained on the ICDAR7 dataset and tested on the same dataset, the results were comparable to those from the research area, as shown in Figure 4g. The model achieves a classification accuracy of 0.90. The model does not distinguish B-class from C-class and F-class from G-class. By checking the E-class, it is visible that it cannot be distinguished from all other classes, which causes a total loss of classification accuracy. The results of the statistical analysis, shown in Figure 4h, are based on the previous tests. The model achieves a high probability of correct and incorrect predictions. The overall classification performance would increase significantly by increasing the variety of images in the training set.

4.5 Obtained results using SSCNN

The same dataset used to test the SCNN model was also used to test the SSCNN model. The SSCNN model was tested on both the ChartDS and ICDAR7 datasets. Figures 5a and 5b show that the SSCNN model has an advantage over SCNN. The model successfully works with a minimal number of images per class. When only one image per chart-type is used for training the model, it achieves a classification accuracy of 0.4 to 0.6. However, when 20 images per chart-type are used for model training, the classification accuracy drops, despite the increased number of examples. The model reaches the maximum classification accuracy when the input image is compared with one from each class, K = 1. However, when the entire dataset is used for comparison, the progress is slower, and more examples are needed to achieve the same result. The model trained on one dataset performs better when tested with data from another dataset, but the results are still below average. By performing manual inspection, the model does not distinguish between B-class and C-class, and F-class and G-class. SSCNN can also generalize for inputs not participating in the training process. Figures 5c and 5d show the current H-class. SSCNN was not trained on the H-class in the training process, and it sees the entire H-class for the first time. The H-class classification accuracy for any dataset is more than 0.8, and the maximum classification accuracy is achieved with a model trained on 50 or more images per class. The success of the SSCNN model on the H-class can be seen in Figures 6a and 6b. The left image compares two slightly preprocessed images from the Linnaeus dataset, and the right image compares the same image from the Linnaeus dataset with a scatter chart from the ChartDS dataset. SSCNN successfully classifies the image into the H-class with an SS of 0.21. In contrast, the scatter chart SS is significantly higher at 0.60 (a value closer to 0 indicates the same class). Additionally, the SSCNN was tested on the ”ReVision” dataset, using five chart-types: area chart, bar chart, line chart, radar, and Venn diagram. All images underwent the same preprocessing as the ChartDS and ICDAR7 dataset images. From each chart-type, 100 images were randomly selected for testing. The total achieved classification accuracy is 0.96. It is essential to note that the models did not see images from the ”ReVision” dataset in the training process.

4.6 In-Depth analysis of the networks

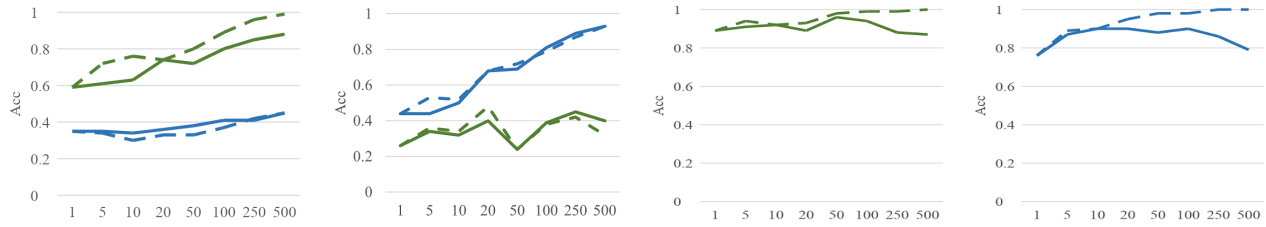
A comprehensive examination of the network’s capabilities is imperative before assessing the dataset using the previously delineated network architecture. This scrutiny aims to ascertain whether the convolutional layers can effectively discern crucial information from less significant details within the image. The analysis employs the method of feature extraction from convolutional layers, utilizing Feature Visualization to represent learned abstract features graphically.

Feature Visualization involves maximizing the activation function to visually display the features of a single neuron, channel, or layer in a neural network. While visualizing individual neurons is impractical due to their foundational nature, a computationally efficient approach involves visualizing channel features for an individual layer or the entire neural network. Features comprise receptive field filters and activation matrices, each corresponding to a specific filter, with the activation matrix representing the degree of feature presence in the input image.

In the presented SCNN architecture, four convolutional layers with varying numbers of channels and a shared 3 $\times$ 3 point receptive field filter are employed. An analysis of the donut, horizontal bar, and line chart progression through these layers is detailed in Figure 7. The visualization reveals that the first convolutional layer learns primitive shapes like lines and edges, with subsequent layers focusing on patterns and textures. The final layer captures object elements bounded by boundaries, emphasizing essential information for the subsequent fully-connected layer.

The visualization underscores the network’s proficiency in recognizing simple elements, defining chart-types, and leveraging them for classification decisions. Image preprocessing enhances the observation of critical primitive elements, while background and fill color are not emphasized.

Comparative analysis based on time and space complex-



(a) (b) (c) (d)

Figure 5: SSCNN classification accuracy for seven chart-types without H-class (a,b). Classification accuracy for the H-class only (c,d). The model is trained on ChartDS (a,c). The model is trained on ICDAR7 (b,d). The evaluation results are consistent with MCCV testing on the validation dataset. The blue line represents testing on ICDAR7, while the green line represents testing on ChartDS. A continuous line is for K=1, and a dashed line is for K=dataset.

(a) (b)

Figure 6: Comparison of two H-classes (Linnaeus). The SS is 0.21 (a). Comparison of H-class (Linnaeus) and scatter chart (ChartDS). The SS is 0.60 (b).

Architecture	Number of param. ($\times 10^6$)	MACs ($\times 10^9$)	Number of weight layers	Time to train (CPU) [s]	Time to train (GPU) [s]	Accuracy
SCNN (Bajić et al. [39])	0.50	0.33	5	4,400	40	0.99
AlexNet	61.10	0.77	8	6,000	60	0.98
SSCNN (Bajić et al. [39])	1.01	0.67	10	9,000	100	1.00
VGG16	138.36	15.61	16	-	800	0.95
MobileNet v1	4.24	0.56	23	13,200	240	0.99
Inception v3	27.16	5.75	48	-	200	0.99
ResNet50 v2	25.56	4.14	50	-	600	0.30
Xception	42.8	16.80	71	-	-	-
DenseNet121	7.98	2.90	121	-	700	1.00
EffNetB0	5.28	0.42	237	26,000	520	0.99

Table 2: Comparison of time and spatial complexities for different architectures. The symbol ”-” denotes the networks ability to learn with available computational resources.

ity, defined by the sum of all weights and biases, indicates that convolutional layers influence both factors. Time complexity includes convolutional layers, max-pooling, and fully-connected layers, with convolutional layers accounting for 90% of computational time. The increase in convolutional layers significantly escalates space and time complexity.

The comparison of time and space complexity and evaluation of developed SCNN, SSCNN, and selected architectures is listed in Table 2. The process was conducted using well-established open-source frameworks, including TensorFlow, Keras, and PyTorch. All models underwent training using the AMD EPYC 7B12 central processing unit (CPU) with 13 GB of available memory, supported by the NVIDIA T4 graphics processing unit (GPU) with 16 GB of available memory. The parameters used during the training phase remained consistent across different models to ensure the consistency and repeatability of the results obtained during the evaluation. For the purpose of comparing the achieved results with other architectures, the ChartDS dataset was used for training, consisting of a total of 500 images per chart-type, while testing utilized 100 images per chart-type.

When compared with other state-of-the-art architectures

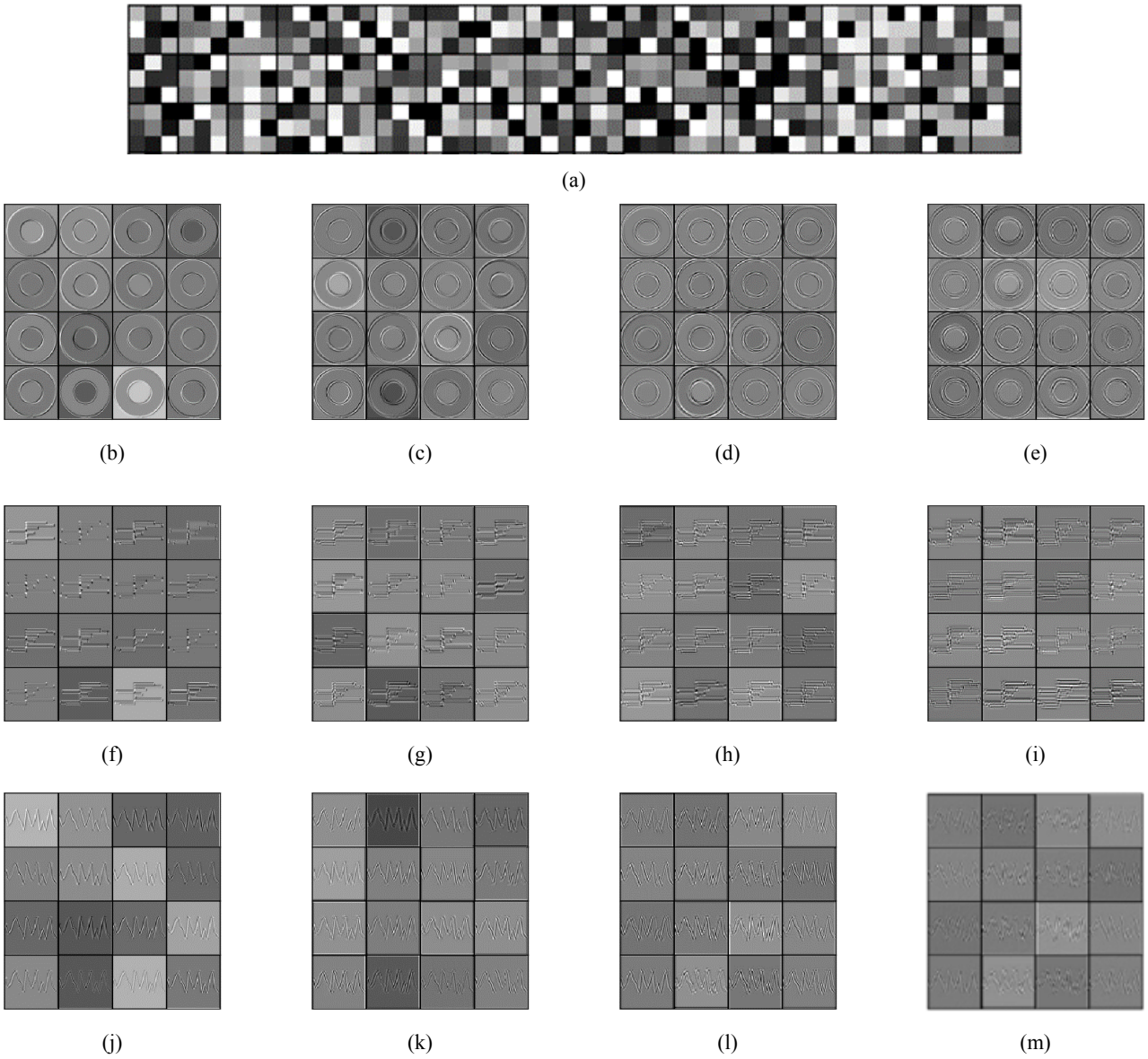


Figure 7: Progress display of donut (b,c,d,e), horizontal bar (f,g,h,i), and line (j,k,l,m) chart through convolutional layers. The first row (a) shows an example of a receptive field filter with a resolution of 3 × 3 points. The further rows show several channels of each convolutional layer: (b,f,j) Conv2d-1, (c,g,k) Conv2d-2, (d,h,l) Conv2d-3, (e,i,m) Conv2d-4.

in the field, the proposed SCNN architecture excels in chart image classification through its innovative design. Comprising four convolutional layers, two max-pooling layers, and one fully-connected layer, it strategically processes visual information. This architecture demonstrates superior efficiency, achieving significantly higher classification accuracy than other methodologies in the field, as shown in Table 1.

The success of SCNN can be attributed to its ability to capture and hierarchically process intricate features present in chart images. Furthermore, SCNN's performance benefits from its adaptability to smaller datasets, reducing the computational resources required for training. This efficiency is crucial in real-world scenarios where large, labeled datasets may be challenging to obtain. The architecture's effectiveness is highlighted in its successful training on publicly available datasets, ChartDS and ICDAR7, and its consistently high accuracy across various chart-types.

The SCNN architecture achieves optimal results by combining advanced convolutional neural network principles with a design that prioritizes efficiency, adaptability to smaller datasets, and robust feature extraction mechanisms.

5 Discussion

The evolution of employed methodologies can be observed through the examination of related research. Presently, the predominant methods hinge on CNNs, which have demonstrated state-of-the-art performance in chart-type classifi-

cation. Notably, prevalent CNN models are rooted in deep architectures, demanding substantial computational resources that may not be universally accessible. Addressing this computational constraint, we introduce the SCNN architecture tailored for chart image classification. The examination of related research also delves into the significance of image preprocessing, elucidating its profound influence on the neural network's learning outcomes. Empirical findings reveal a positive correlation between adept image preprocessing and neural network learning, even within a restricted dataset. Furthermore, an extensive dataset, denoted as ChartDS, has been curated. To facilitate comparative analyses with existing research, the SCNN is evaluated on renowned publicly available datasets, including one from "ReVision" and the ICDAR7 competition dataset. The attained state-of-the-art results underscore the SCNN's capacity for learning across diverse datasets, provided the images contain essential information exclusively.

In the context of employing SCNN within a system dedicated to objectives such as chart description generation, chart data extraction, or the identification and extraction of scientific figures from documents, a fundamental limitation arises. The inherent deficiency lies in the system's lack of knowledge concerning entities beyond chart images, leading to its incapacity in such diverse tasks. To address this limitation, we introduced two supplementary datasets, CIFAR10 and Linnaeus, encompassing various images encountered in authentic systems. While SCNN exhibits utility in open-set classification, superior outcomes are discerned through SSCNN, a composite structure comprising two distinct SCNNs.

Lastly, an examination of the SCNN encompassed an evaluation of its time and space complexity. Comparative analyses were conducted with contemporary networks such as ResNet, Inception, Xception, among others, revealing a notable reduction in both parameters and computational operations. Remarkably, the SCNN demonstrated adaptability to training on machines with or without GPU support, featuring significantly diminished real-time requirements compared to alternative deep architectures.

Further scrutiny included subjecting the SCNN to limitations and unveiling potential avenues for future research. Specifically, attention is warranted towards refining the classification of chart sub-types, such as horizontally grouped or stacked charts. Additionally, avenues for fine-tuning the SCNN include enhancing its capability to classify partially visible charts or discerning multiple charts within a singular image.

6 Conclusion

Methods for classifying chart images have evolved over time. Traditional methods used before 2015 had limited accuracy and required chart images to follow predefined rules. However, with the advent of SVMs and CNNs, classification accuracy has increased significantly. In particular, "ReVision" was the first journal article to introduce multi-class classification using SVM. The authors used various image preprocessing methods and algorithms to achieve the highest possible chart image classification accuracy. With the emergence of CNN in the field, the methods and algorithms for preprocessing the chart image were retained, and the number of classes was significantly increased. This research presents the SCNN architecture used for chart image classification. The architecture comprises four convolutional layers, two max-pooling layers, and one fully-connected layer. It can be used independently or in combination with another network, such as SSCNN. The experimental evaluation shows that the SCNN architecture is efficient and achieves significantly higher classification accuracy than other research in the field. Additionally, it requires smaller datasets for training and reduces the necessary computer resources. Using the SSCNN architecture, high classification accuracy values can be achieved with small datasets. Based on the SCNN architecture, 32 models were created and trained on two publicly available datasets, ChartDS and ICDAR7. Each of the models was tested on both datasets, allowing for the creation of a reference starting point for future research. The models were also tested on open-set classification, where they achieved high accuracy in recognizing chart images from other images. However, the SSCNN architecture and N-way-K-shot learning showed a significant advantage in open-set classification. The SCNN architecture needs an additional class, while the SSCNN architecture achieves generalization, even for images that have never participated in model training.

In the future, we plan to expand our evaluation to support more chart-types. The H-class should also be expanded to include a more diverse selection of images not belonging to any chart-types.

References

[1] J. Choi, S. Jung, D. G. Park, J. Choo, and N. Elmqvist, "Visualizing for the non□visual: Enabling the visually impaired to use visualization," *Computer Graphics Forum*, vol. 38, 2019, https://doi.org/10.1111/cgf.13686.

[2] J. Poco and J. Heer, "Reverse□engineering visualizations: Recovering visual encodings from chart images," *Computer Graphics Forum*, vol. 36, 2017, https://doi.org/10.1111/cgf.13193.

[3] F. Bajić and J. Job, "Review of chart image detection and classification," *International Journal on Document Analysis and Recognition (IJDAR)*, pp. 1–22, 2023, https://doi.org/10.1007/s10032-022-00424-5.

[4] F. Bajić, J. Job, and K. Nenadić, "Chart classification using simplified vgg model," *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 229–233, 2019, https://doi.org/10.1109/IWSSIP.2019.8787299.

[5] K. C. Shahira and A. Lijiya, "Document image classification: Towards assisting visually impaired," *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pp. 852–857, 2019, https://doi.org/10.1109/TENCON.2019.8929594.

[6] F. Bajić, J. Job, and K. Nenadic, "Data visualization classification using simple convolutional neural network model," *International Journal of Electrical and Computer Engineering*, vol. 11, pp. 43–51, 2020, https://doi.org/10.32985/ijeces.11.1.5.

[7] C. Rane, S. M. Subramanya, D. S. Endluri, J. Wu, and C. L. Giles, "Chartreader: Automatic parsing of barplots," *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)*, pp. 318–325, 2021.

[8] W. Dai, M. Wang, Z. Niu, and J. Zhang, "Chart decoder: Generating textual and numeric information from chart images automatically," *J. Vis. Lang. Comput.*, vol. 48, pp. 101–109, 2018, https://doi.org/10.1016/J.JVLC.2018.08.005.

[9] A. Balaji, T. Ramanathan, and V. Sonathi, "Chart-text: A fully automated chart image descriptor," *ArXiv*, vol. abs/1812.10636, 2018, https://doi.org/10.48550/arXiv.1812.10636.

[10] D. Jung, W. Kim, H. Song, J. Hwang, B. Lee, B. H. Kim, and J. Seo, "Chartsense: Interactive data extraction from chart images," *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, https://doi.org/10.1145/3025453.3025957.

[11] I. Redeke, "Image & graphic reader," *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, vol. 1, pp. 806–809 vol.1, 2001, https://doi.org/10.1109/ICIP.2001.959168.

[12] J. Gao, Y. Zhou, and K. E. Barner, "View: Visual information extraction widget for improving chart images accessibility," *2012 19th IEEE International Conference on Image Processing*, pp. 2865–2868, 2012, https://doi.org/10.1109/ICIP.2012.6467497.

[13] A. Mishchenko and N. Vassilieva, "Model-based recognition and extraction of information from chart images," *J. Multim. Process. Technol.*, vol. 2, pp. 76–89, 2011.

[14] ——, "Model-based chart image classification," in *International Symposium on Visual Computing*, 2011, https://doi.org/10.1007/978-3-642-24031-7_48.

[15] N. Siegel, Z. Horvitz, R. Levin, S. K. Divvala, and A. Farhadi, "Figureseer: Parsing result-figures in research papers," in *European Conference on Computer Vision*, 2016, https://doi.org/10.1007/978-3-319-46478-7_41.

[16] M. Savva, N. Kong, A. Chhajta, L. Fei-Fei, M. Agrawala, and J. Heer, "Revision: automated classification, analysis and redesign of chart images," *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, https://doi.org/10.1145/2047196.2047247.

[17] R. R. Nair, N. Sankaran, I. Nwogu, and V. Govindaraju, "Automated analysis of line plots in documents," *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 796–800, 2015, https://doi.org/10.1109/ICDAR.2015.7333871.

[18] Y. Shi, Y. Wei, T. Wu, and Q. Liu, "Statistical graph classification in intelligent mathematics problem solving system for high school student," *2017 12th International Conference on Computer Science and Education (ICCSE)*, pp. 645–650, 2017, https://doi.org/10.1109/ICCSE.2017.8085572.

[19] V. Karthikeyani and S. Nagarajan, "Machine learning classification algorithms to recognize chart types in portable document format (pdf) files," *International Journal of Computer Applications*, vol. 39, pp. 1–5, 2012, https://doi.org/10.5120/4789-6997.

[20] B. Cheng, R. J. Stanley, S. K. Antani, and G. R. Thoma, "Graphical figure classification using data fusion for integrating text and image features," *2013 12th International Conference on Document Analysis and Recognition*, pp. 693–697, 2013.

[21] X. Liu, B. Tang, Z. Wang, X. Xu, S. Pu, D. Tao, and M. Song, "Chart classification by combining deep convolutional networks and deep belief networks," *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 801–805, 2015, https://doi.org/10.1109/ICDAR.2015.7333872.

[22] P. R. S. C. Junior, A. A. de Freitas, R. D. Akiyama, B. P. Miranda, T. Araújo, C. G. R. Santos, B. S. Meiguins, and J. M. de Morais, "Architecture proposal for data extraction of chart images using convolutional neural network," *2017 21st International Conference Information Visualisation (IV)*, pp. 318–323, 2017, https://doi.org/10.1109/IV.2017.37.

[23] J. Amara, P. Kaur, M. Owonibi, and B. Bouaziz, "Convolutional neural network based chart image classification," 2017.

[24] L. Battle, P. Duan, Z. Miranda, D. Mukusheva, R. Chang, and M. Stonebraker, "Beagle: Automated extraction and interpretation of visualizations from the web," *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2017.

[25] A. Y. Lin, J. Ford, E. Adar, and B. J. Hecht, "Vizbywiki: Mining data visualizations from the web to enrich news articles," *Proceedings of the 2018 World*

Wide Web Conference, 2018, https://doi.org/10.1145/3178876.3186135.

[26] K. V. Jobin, A. Mondal, and C. V. Jawahar, "Docfigure: A dataset for scientific document figure classification," *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, vol. 1, pp. 74–79, 2019, https://doi.org/10.1109/ICDARW.2019.00018.

[27] X. Liu, D. Klabjan, and P. N. Bless, "Data extraction from charts via single deep neural network," *ArXiv*, vol. abs/1906.11906, 2019.

[28] K. Davila, B. U. Kota, S. Setlur, V. Govindaraju, C. Tensmeyer, S. Shekhar, and R. Chaudhry, "Icdar 2019 competition on harvesting raw tables from infographics (chart-infographics)," *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1594–1599, 2019, https://doi.org/10.1109/ICDAR.2019.00203.

[29] P. Kaur and D. Kiesel, "Combining image and caption analysis for classifying charts in biodiversity texts," in *VISIGRAPP*, 2020, https://doi.org/10.5220/0008946701570168.

[30] C. Kosemen and D. Birant, "Multi-label classification of line chart images using convolutional neural networks," *SN Applied Sciences*, vol. 2, pp. 1–20, 2020, https://doi.org/10.1007/S42452-020-3055-Y.

[31] T. Ishihara, K. Morita, N. C. Shirai, T. Wakabayashi, and W. Ohyama, "Chart-type classification using convolutional neural network for scholarly figures," in *Asian Conference on Pattern Recognition*, 2019, https://doi.org/10.1007/978-3-030-41299-9_20.

[32] J. Zhu, J. Ran, R. K.-W. Lee, K. Choo, and Z. Li, "Autochart: A dataset for chart-to-text generation task," in *Recent Advances in Natural Language Processing*, 2021.

[33] S. Huang, "An image classification tool of wikimedia commons," 2020.

[34] K. Davila, F. Xu, S. Ahmed, D. A. Mendoza, S. Setlur, and V. Govindaraju, "Icpr 2022: Challenge on harvesting raw tables from infographics (chart-infographics)," *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 4995–5001, 2022, https://doi.org/10.1109/ICPR56361.2022.9956289.

[35] K. Dadhich, S. C. Daggubati, and J. Sreevalsan-Nair, "Barchartanalyzer: Digitizing images of bar charts," in *International Conference on Image Processing and Vision Engineering*, 2021, https://doi.org/10.5220/0010408300170028.

[36] J. Thiyam, S. R. Singh, and P. K. Bora, "Challenges in chart image classification: a comparative study of different deep learning methods," *Proceedings of the 21st ACM Symposium on Document Engineering*, 2021, https://doi.org/10.1145/3469096.3474931.

[37] F. Bajić and J. Job, "Chart classification using siamese cnn," *Journal of Imaging*, vol. 7, 2021, https://doi.org/10.3390/jimaging7110220.

[38] J. Thiyam, S. R. Singh, and P. K. Bora, "Effect of attention and triplet loss on chart classification: a study on noisy charts and confusing chart pairs," *Journal of Intelligent Information Systems*, vol. 60, pp. 731 – 758, 2022, https://doi.org/10.1007/s10844-022-00741-5.

[39] F. Bajić, O. Orel, and M. Habijan, "A multi-purpose shallow convolutional neural network for chart images," *Sensors (Basel, Switzerland)*, vol. 22, 2022, https://doi.org/10.3390/s22207695.

[40] W. Xue, D. Chen, B. Yu, Y. Chen, S. Zhou, and W. Peng, "Chartdetr: A multi-shape detection network for visual chart recognition," *ArXiv*, vol. abs/2308.07743, 2023, https://doi.org/10.48550/arXiv.2308.07743.

[41] P. Chagas, R. D. Akiyama, A. S. G. Meiguins, C. G. R. Santos, F. de Oliveira Saraiva, B. S. Meiguins, and J. M. de Morais, "Evaluation of convolutional neural network architectures for chart image classification," *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2018, https://doi.org/10.1109/IJCNN.2018.8489315.

[42] T. Araújo, P. Chagas, J. B. Alves, C. G. R. Santos, B. S. Santos, and B. S. Meiguins, "A real-world approach on the problem of chart recognition using classification, detection and perspective correction," *Sensors (Basel, Switzerland)*, vol. 20, 2020, https://doi.org/10.3390/S20164370.

[43] A. Dhote, M. H. Javed, and D. S. Doermann, "A survey and approach to chart classification," in *ICDAR Workshops*, 2023, https://doi.org/10.1007/978-3-031-41498-5_5.

[44] F. Bajić, "Chartds - chartdataset.zip. figshare," 2022, https://doi.org/10.6084/m9.figshare.19524844.v1.

[45] F. Bajić, "Chartds 2022 - chartdataset.zip srce dabar," 2022, https://urn.nsk.hr/urn:nbn:hr:102:276396.

[46] G. Chaladze and K. L, "Linnaeus 5 dataset for machine learning," 2017.

[47] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[48] S. Z. Li and A. K. Jain, "Handbook of face recognition," 2011, https://doi.org/10.1007/978-0-85729-932-1.

[49] M. H. F. Bajić and K. Nenadić, "A synthetic dataset of different chart types for advancements in chart identification and visualization," 2024, https://doi.org/10.1016/j.dib.2024.110233.

[50] F. de Oliveira Costa, M. Eckmann, W. J. Scheirer, and A. Rocha, "Open set source camera attribution," *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 71–78, 2012, https://doi.org/10.1109/SIBGRAPI.2012.19.

[51] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, 2013, https://doi.org/10.1109/TPAMI.2012.256.

[52] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The extreme value machine," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 762–768, 2015, https://doi.org/10.48550/arXiv.1506.06112.

[53] T. E. Boult, S. Cruz, A. R. Dhamija, M. Günther, J. Henrydoss, and W. J. Scheirer, "Learning and the unknown: Surveying steps toward open world recognition," in *AAAI Conference on Artificial Intelligence*, 2019, https://doi.org/10.1609/aaai.v33i01.33019801.

[54] H. Zhang and V. M. Patel, "Sparse representation-based open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1690–1696, 2017, https://doi.org/10.1109/TPAMI.2016.2613924.

[55] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 2015, https://doi.org/10.48550/arXiv.1506.01497.

[56] Z. Chen, Y. Fu, Y. Zhang, Y.-G. Jiang, X. Xue, and L. Sigal, "Multi-level semantic feature augmentation for one-shot learning," *IEEE Transactions on Image Processing*, vol. 28, pp. 4594–4605, 2018, https://doi.org/10.48550/arXiv.1804.05298.

[57] Y. Fu, T. Xiang, Y.-G. Jiang, X. Xue, L. Sigal, and S. Gong, "Recent advances in zero-shot recognition: Toward data-efficient understanding of visual content," *IEEE Signal Processing Magazine*, vol. 35, pp. 112–125, 2017, https://doi.org/10.1109/MSP.2017.2763441.