Available online at http://scik.org

Commun. Math. Biol. Neurosci. 2025, 2025:141

https://doi.org/10.28919/cmbn/9635

ISSN: 2052-2541

NEUROCOGNITIVE PREDICTION OF DYSLEXIC HANDWRITING PATTERN USING AN EXPLAINABLE AI-DRIVEN CUSTOM LITEBINARYNET-CNN

KARUNIA EKA LESTARI[1,*], SRI WINARNI[2], ADITYA PRIHANDHIKA[1], EDWIN SETIAWAN NUGRAHA[3], MOKHAMMAD RIDWAN YUDHANEGARA[1]

[1]Department of Mathematics Education, Universitas Singaperbangsa Karawang, Karawang 41361, Indonesia

[2]Department of Statistics, Universitas Padjadjaran, Sumedang 45363, Indonesia

[3]Actuarial Science Study Program, President University, Bekasi 17530, Indonesia

Abstract: Artificial intelligence (AI) based on deep learning, particularly convolutional neural networks (CNNs), shows strong potential in handwriting recognition. However, their limited transparency constrains use in sensitive domains such as dyslexia prediction. From a neurocognitive standpoint, dyslexia stems from atypical neural processing reflected in handwriting irregularities, making handwriting prediction a neurocognitive inference task. This study introduces a neurocognitively informed framework, Custom LiteBinaryNet, a lightweight CNN integrated with Explainable AI (XAI) for transparent dyslexia prediction from handwriting. Custom LiteBinaryNet-CNN was evaluated in baseline and tuned configurations, the latter optimized through aggressive augmentation and hyperparameter tuning. Compared to LeNet-5 (60.49% accuracy, AUC 0.56), the baseline achieved 78.73% accuracy and AUC 0.87, while the tuned model reached 83.36% accuracy and AUC 0.91. Loss analysis confirmed improved stability and generalization. XAI methods, including Grad-CAM and Occlusion Sensitivity, revealed neurocognitive interpretability by highlighting handwriting traits, such as letter reversals and spatial inconsistencies, which linked to dyslexic motor patterns. These results align computational predictions with cognitive evidence, enhancing transparency and diagnostic value. The proposed model offers a practical and explainable approach for early neurocognitive prediction of dyslexia through handwriting analysis.

Keywords: convolutional neural network; custom litebinarynet; dyslexia; explainable-AI; handwriting prediction.

2020 AMS Subject Classification: 68T07, 68T09.

---

*Corresponding author

E-mail address: karunia@staff.unsika.ac.id

Received Nov. 07, 2025

1. INTRODUCTION

Dyslexia is a specific learning disability characterized by neurobiological differences in the brain's processing of written and phonological language, leading to difficulties in reading, writing, spelling, and speech [1]. It affects approximately 5–10% of the global population [2], with around five million school-aged children in Indonesia estimated to experience dyslexia [3]. Dyslexia can negatively impact a student's self-image, leading to insecurity and reduced self-esteem. Many students with dyslexia develop a sense of inferiority, perceiving themselves as less capable than their peers, which may ultimately discourage them from pursuing education [4]. However, dyslexia is not linked to a lack of intelligence or motivation; rather, individuals with dyslexia can learn effectively when provided with the right learning strategies.

Early identification of dyslexia is therefore vital, as it has profound academic, social, and emotional implications. Early detection enables educators and parents to design personalized interventions, improve learning outcomes, and enhance children's long-term development. Empirical evidence has revealed that handwriting patterns can serve as a potential indicator of dyslexia. Studies on writing dynamics have reported significant differences in writing speed, pause frequency, and pen pressure variation between dyslexic and non-dyslexic children [5]. Digital analyses further show that handwriting features such as letter size variation and stroke irregularity can distinguish dyslexic children with sufficient accuracy [6]. These findings support the idea that handwriting can serve as an early detection tool before further clinical assessment. With the assistance of computing technology and machine learning algorithms, handwriting patterns that are difficult to observe manually can now be quantitatively extracted, opening new opportunities for early dyslexia screening [6], [7].

Several studies have explored the use of artificial intelligence to detect dyslexia through handwriting analysis. Asselborn et al. [6] employed traditional machine learning algorithms such as Random Forest and Support Vector Machine to analyze digital handwriting dynamics, including pen pressure, letter size, and stroke spacing, with encouraging results in distinguishing children with graphomotor disorders and dyslexia. Aldehim et al. [8] applied deep learning based on Convolutional neural networks (CNNs) to process handwriting images and reported high accuracy in classifying dyslexic groups. Liu et al. [9] leveraged long short-term memory (LTSM) to develop a CNN-Positional-LSTM-Attention (DysDiTect) model that combined CNN's spatial extraction capabilities with LSTM sequential modeling and attention mechanisms, thereby increasing sensitivity to variations in graphomotor patterns. Such studies suggest the strong potential of CNNs in handwriting recognition, as this approach can identify distortion patterns in handwriting, including variations in letter shape, size, spacing, slant, thickness, spelling errors, and spatial inconsistencies [8].

CNNs are widely used for computer vision tasks such as image classification, object detection, semantic segmentation, image labeling, and visual question answering [10], [11]. CNN model architectures have become research hotspots for solving text and image classification problems [12], [13]. The CNN operates by performing a series of convolutions, pooling, and nonlinear transformations, producing highly abstract feature representations that do not always correspond to human-understandable concepts. The complexity of these layers makes it difficult to determine which features are most important for making predictions. Thus, although CNNs often achieve high predictive accuracy, their internal mechanisms are difficult for humans to interpret directly [14]. This lack of interpretability hinders understanding of how specific cognitive or perceptual patterns are encoded and processed by the network, limiting the model's applicability in educational and neurocognitive contexts.

Previous researchers have highlighted the limitations of CNNs in terms of model transparency and interpretability [15], [16], [17]. For instance, among dyslexic children, confusion between letters such as "b" and "d" is often observed. CNNs can extract and classify such visual features, enabling automated analysis of handwriting patterns associated with dyslexia. However, when a CNN classifies a piece of handwriting as "dyslexic," it is not always clear which features most influenced that decision. Questions arise as to whether the decision was driven by inconsistent letter height, irregular spacing, or letter reversal tendencies. For professionals in education and therapy, such interpretive precision is crucial: the model's classification should not merely label handwriting as dyslexic or normal but also provide neurocognitively meaningful insights into the writer's perceptual–motor processes.

From an AI perspective, addressing this "black box" problem is essential to ensure transparency and interpretability in the decision-making process. Therefore, an approach is needed to simplify and clarify the interpretation of CNN model architectures. Explainable Artificial Intelligence (XAI) enhances model interpretability by providing insight into how decisions are made, identifying model strengths and weaknesses, and offering explanations of learned representations [18]. XAI enables researchers to visualize and understand the features learned from image data, ensuring that

AI models recognize meaningful patterns rather than memorize superficial details. Early attempts to integrate XAI methods have included applying Gradient-weighted Class Activation Mapping (Grad-CAM) to highlight handwriting areas considered influential in the classification process. Using transfer learning with pre-trained architectures such as MobileNetV3, these studies successfully identified subtle handwriting patterns that are difficult to detect manually [19]. Although these approaches achieved high accuracy and improved interpretability, they remained supplementary to the classification process and were not fully integrated into the CNN pipeline.

The literature review reveals that most prior studies have placed greater emphasis on enhancing classification accuracy [20], whereas the clarity of prediction has been largely overlooked, despite its critical importance for educational and psychological practitioners. Integrating XAI is proposed as a solution to overcome the black box problem and increase the interpretive relevance of prediction results in dyslexia detection. Combining CNNs with Grad-CAM provides heatmaps that highlight handwriting regions most influential in classification decisions. Grad-CAM can clearly visualize letter areas or strokes most relevant to CNN decisions, such as distortions in the letter "b" or abnormal slanting in the letter "d." In addition, Occlusion Sensitivity Analysis (OSA) complements Grad-CAM by systematically occluding small regions of an input image and measuring the resulting changes in prediction scores, thereby quantitatively identifying the areas most critical to the model's decision. Such analyses offer a clearer understanding of why CNNs produce certain classification outcomes, helping researchers, psychologists, and educators interpret handwriting error patterns in dyslexic children.

Given these considerations, it is essential to strengthen the mathematical and statistical framework that supports the integration of XAI-driven Custom LiteBinaryNet-CNN to ensure both reliability and interpretability in addressing the black box problem. Therefore, this study focuses on the neurocognitive prediction of dyslexic handwriting patterns using an XAI-driven Custom LiteBinaryNet-CNN architecture. The research questions guiding this study are as follows:

1. How can a CNN-based Custom LiteBinaryNet be designed and trained to predict handwriting from children with and without dyslexia effectively from a neurocognitive perspective?
2. How can XAI methods, such as Grad-CAM and Occlusion Sensitivity, be applied to open the black box of CNN models and provide understandable visual explanations?

3. How does the Custom LiteBinaryNet-CNN model integrated with XAI perform in terms of accuracy and interpretability when predicting handwriting data from children with dyslexia?

This study introduces original contributions in three primary domains. First, it proposes an XAI-driven Custom LiteBinaryNet-CNN architecture, a lightweight yet efficient CNN variant optimized for handwriting prediction in children. Second, it integrates XAI methodologies, namely Grad-CAM and OSA, to mitigate the "black box" problem and provide interpretable, neurocognitively meaningful visualizations. Third, the framework applies these methods to the underexplored domain of neurocognitive prediction of dyslexic handwriting patterns, contributing to both efficient CNN modeling and transparent interpretive analytics that can support early dyslexia detection.

2. PRELIMINARIES

This section introduces the XAI-driven Custom LiteBinaryNet-CNN framework designed to predict dyslexic handwriting from a neurocognitive perspective. The method addresses three main objectives: developing an efficient CNN-based architecture, applying XAI techniques such as Grad-CAM and Occlusion Sensitivity to enhance interpretability, and evaluating the model's accuracy and transparency. This approach aims to bridge computational performance with neurocognitive insight, supporting reliable early dyslexia prediction.

2.1 Research Design

The research method employed is an exploratory design, aiming to examine and explore a relatively novel topic that has not yet been extensively studied. In this context, the topic under discussion is XAI-driven Custom LiteBinaryNet-CNN for recognizing handwriting patterns in the detection of children with dyslexia. The exploratory approach was adopted due to the fact that this field is not yet fully developed, both conceptually and empirically. As such, an initial mapping of the method's potential and practical implications was deemed necessary. This exploratory approach is expected to make a significant contribution in two aspects. First, it will broaden the theoretical understanding of the integration of CNN and XAI in the domains of education and child health. Secondly, it will provide a practical framework with the potential to support educators, psychologists, and health workers in the early identification of children with dyslexia. This will allow for the provision of interventions that are more timely, precise, and evidence-based.

The methodological framework of this study comprised five sequential stages. All five stages are illustrated in Figure 1. First, an extensive data collection process was initiated through the systematic search and selection of datasets comprising handwriting samples from dyslexic and non-dyslexic (normal) children as a control group. The objective of this process was to obtain a representative database, thereby enabling the model to differentiate between the distinctive patterns of handwriting exhibited by children with dyslexia and the typical handwriting variations observed in the general population. Second, the data were preprocessed with auto-orientation based on EXIF metadata, conversion to grayscale, inversion of pixel values, and resizing to ensure image quality and consistency of format. Third, a CNN model was developed as the core of the prediction system. This model was trained to achieve high classification accuracy and was integrated with the XAI framework to produce interpretable outputs. Fourth, post-mining knowledge extraction focused on identifying clinically and educationally relevant handwriting patterns from trained models. Finally, model interpretation involves generating visual explanations, such as Grad-CAM and occlusion sensitivity maps, to support the prediction and early detection of dyslexia.

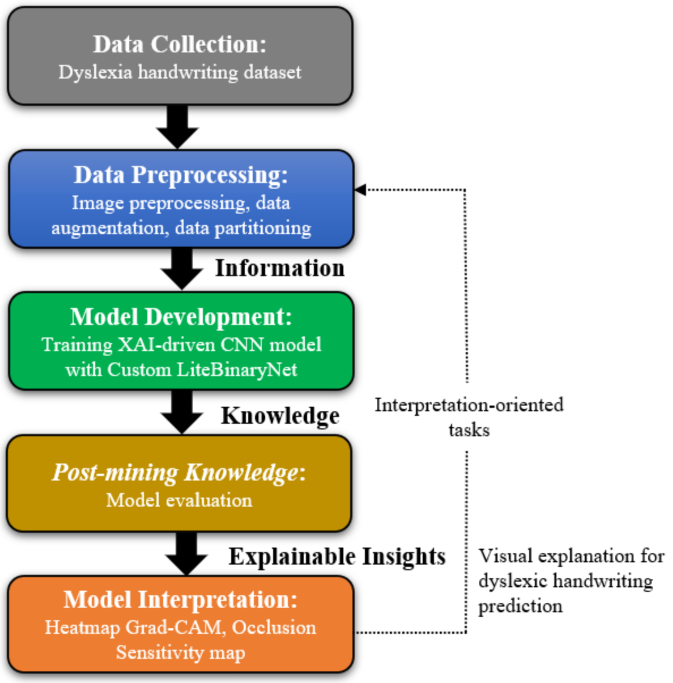

Data Collection:
Dyslexia handwriting dataset
Data Preprocessing:
Image preprocessing, data augmentation, data partitioning
Information
Model Development:
Training XAI-driven CNN model with Custom LiteBinaryNet
Knowledge
Interpretation-oriented tasks
Post-mining Knowledge:
Model evaluation
Explainable Insights
Visual explanation for dyslexic handwriting prediction
Model Interpretation:
Heatmap Grad-CAM, Occlusion Sensitivity map

FIGURE 1. Workflow of the modified data-driven approach for dyslexia handwriting analysis [21]

2.2 Dataset Description

The dataset used in this study was obtained from the Roboflow Universe platform in the Dyslexia Classification project, which is available at https://universe.roboflow.com. The original dataset contains 6954 handwritten images that have undergone preprocessing, including auto-orientation and adjustment of the image size to 640×640 pixels. The data is divided into three subsets, including training data (70%, 4867 images), validation data (20%, 1390 images), and test data (10%, 697 images). However, not all images were analyzed, since one subfolder included images with ambiguous class information, making it impossible to reliably assign them to either the dyslexic or the normal group. Therefore, only 5,513 images with an unambiguous label were used for subsequent analysis, with a proportion of 70% training data, 17.5% validation data, and 12.5% test data. Specifically, this study used 4,822 images for training and 691 images for testing. Each included children's handwriting samples from dyslexic and normal children. The training data was split 0.80: 0.20 into training and validation sets, yielding 3,857 images for training and 965 for validation. In the original version of the dataset, no augmentation techniques were applied. This dataset was used to develop and evaluate a CNN-based classification model designed to distinguish the handwriting of children with and without dyslexia.

2.3 Data Preprocessing and Augmentation

Prior to the analysis of handwriting images by CNN, a series of preprocessing steps must be executed to ensure that the data is in a consistent format and meets the neural network input standards. These steps are intended to normalize image variations, reduce irrelevant information, and prepare the appropriate size and scale so that important features can be extracted optimally.

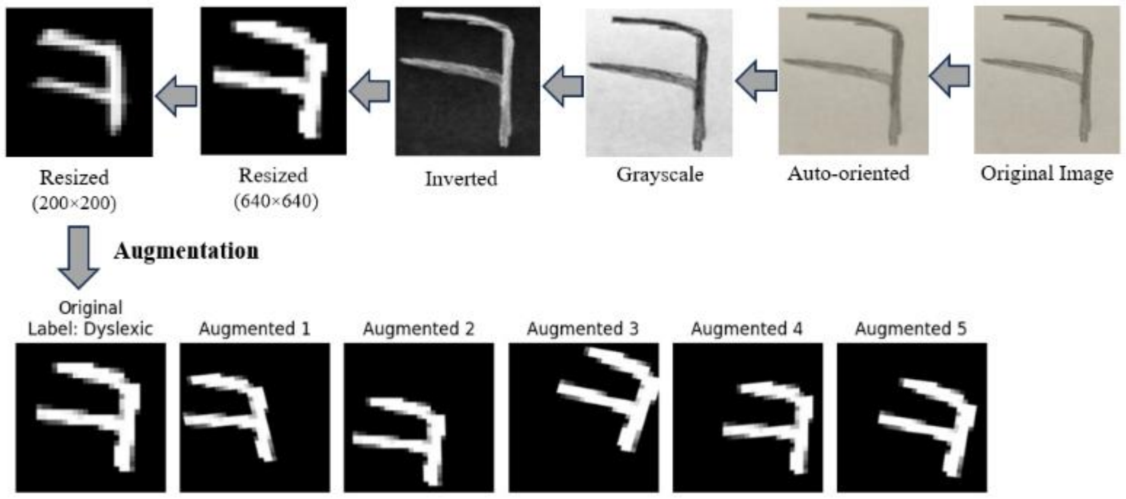


FIGURE 2. Handwritten image preprocessing stages

Figure 2 outlines the stages of preprocessing handwritten images before their use in neural network-based analysis. The stages include: (1) original, raw images that are scanned or photographed, (2) auto-orientation to align the image for uniform positioning, (3) conversion to grayscale to simplify color information, (4) color inversion so that the handwriting appears white on a black background, (5) resizing to 640×640 to adjust the large image size for feature extraction needs, and (6) resizing to 28×28 to scale the image according to standard neural network input formats, such as MNIST [22].

Data augmentation was performed during pre-processing to increase dataset diversity. Basic data augmentation techniques, such as small rotations and translations, were applied in the baseline model. For the tuned model, more aggressive augmentations were applied, including random rotations up to $20°$, random zooming in or out up to 25%, random horizontal and vertical translations up to 25% of the total width and height, random shear transformations up to $20°$, nearest-neighbor interpolation, random brightness adjustments, and the addition of Gaussian noise. Any augmentation that could alter the orientation of letters, such as horizontal flipping, was avoided to preserve the handwriting attributes. Subject-wise data partitioning was performed to prevent data leakage. The selected model was evaluated on the validation set, while the test set was reserved solely for the final evaluation.

2.4 Convolutional Neural Network (CNN)

CNNs represent a particular type of deep learning architecture that learns directly from data without the necessity of manually extracting features. CNN can also be viewed as an artificial neural network that involves convolution. In other words, CNN = ANN + convolution. There are three main layers of CNN, including the convolutional layer, the pooling layer, and the fully-connected layer [23]. CNN works by extracting important patterns in the data, such as edges, textures, and shapes, through filters or kernels in the convolutional layer. The pooling layer reduces the dimensions of the data while retaining important information. This speeds up the computation process and reduces the risk of overfitting. Finally, the fully-connected layer integrates the extracted features to generate predictions or classifications according to the desired task [13].

Convolutional layers apply convolution operations to input images using small filters, or kernels, typically 3×3 or 5×5. Each filter is designed to identify particular patterns, such as edges, textures, or line orientations. These patterns are captured in the feature map, indicating their positions at specific locations in the image. Despite their common origin in the same input image, different filters will yield distinct feature maps because each one is "sensitive" to particular visual characteristics. This process enables CNNs to construct hierarchical representations, from simple features in the early layers to complex features in the deeper layers [22].

Suppose the input image is a tensor $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ where H denotes the height dimension, W denotes the width dimension, and C denotes the channel dimension. Each convolution layer uses a kernel, also referred to as a filter, denoted by $\mathbf{K} \in \mathbb{R}^{h_k \times w_k \times C}$ with kernel size h_k and w_k. The convolution operation at location (i, j) is defined as

(1) $$(\mathbf{X} * \mathbf{K})_{i,j} = \sum_{m=1}^{h_k} \sum_{n=1}^{w_h} \sum_{c=1}^{C} \mathbf{X}_{i+m-1,j+n-1,c} \cdot \mathbf{K}_{m,n,c}.$$

To enable the model to learn more patterns, it is necessary to add the term b to Eq. (1):

(2) $$Z_{i,j} = (\mathbf{X} * \mathbf{K})_{i,j} + b.$$

Thus, even if the input is zero, the output can still be non-zero, that is $Z_{i,j} = b$.

Following the convolution operation, a non-linear activation function is generally employed to improve the network's representation capabilities. The Rectified Linear Unit (ReLU) is one of the most common activation functions. From a mathematical perspective, ReLU is defined as a function that converts all negative input values to zero, while preserving positive values unchanged.

(3) $$\text{ReLU} = A_{i,j} = f(Z_{i,j}) = max(0, Z_{i,j}).$$

According to Eq. (3), the ReLU function is comprised of two linear components. For $Z_{i,j} > 0$, $f(Z_{i,j}) = Z_{i,j}$ is the identity function, and for $Z_{i,j} \leq 0$, $f(Z_{i,j}) = 0$ is a constant. However, since these two components are connected at zero without a smooth gradient continuation, the ReLU function as a whole cannot be considered a linear function.

In the realm of artificial neural networks, the non-linear nature of the ReLU plays a pivotal role. If only linear activation functions are used, such as $f(\mathbf{Z}) = a\mathbf{Z} + b,$ even stacking multiple linear layers will still yield a simple linear transformation. As a result, the network is unable to increase its representation capacity and remains limited in its ability to learn complex data patterns. Conversely, leveraging non-linear functions like ReLU enables the network to discern and learn non-linear relationships between features in the data, leading to more comprehensive representations. Compared to classic non-linear functions, such as sigmoid or tanh, ReLU has the advantage of reducing the risk of vanishing gradients while speeding up the training process [24]. ReLU operates according to a simple rule, which maps every negative value to zero, while positive values remain unchanged. This operation is applied to each element, or pixel, in the feature map. Thus, ReLU improves computational efficiency and maintains gradient propagation stability. Consequently, network training becomes faster and more effective at overcoming the vanishing gradient problem that often arises in deep neural networks.

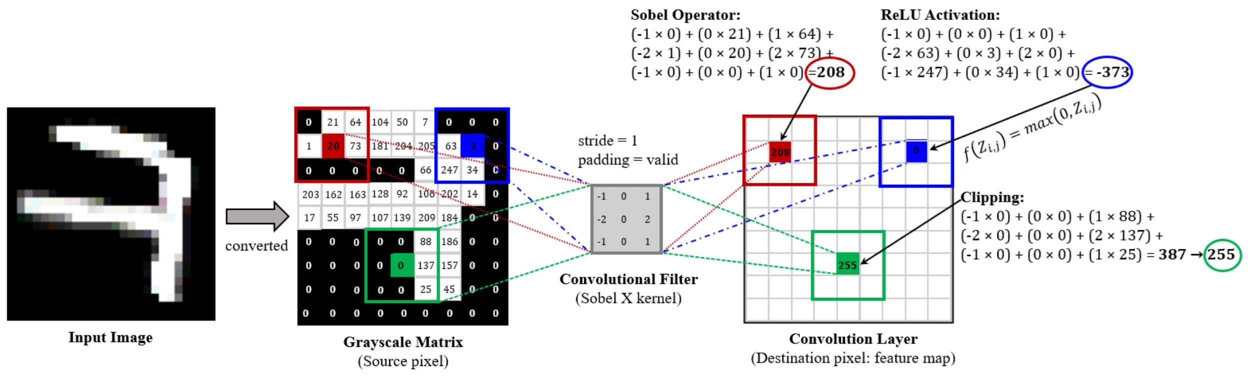


FIGURE 3. Illustration of the convolution and ReLU activation on grayscale images

Figure 3 demonstrates the process of extracting features from grayscale images using the Sobel X kernel. The pixel values in the source image are multiplied by the kernel weights to produce the values in the feature map of the convolution layer. The convolution results can be positive or negative. In this example, the convolution result is 208. The ReLU activation function in Eq. (3) is then applied, ensuring that negative values -373 are mapped to 0, while positive values are retained. This process helps the neural network extract edge features while preventing the vanishing gradient problem.

When performing image convolution with a specific kernel, the calculated pixel values may fall outside the standard grayscale intensity range of 0 to 255. This occurs because the multiplication and addition operations performed on the kernel can produce negative values or values greater than 255. This is especially true for kernels that emphasize edges or enhance contrast, such as sharpening or Laplacian kernels. To maintain the validity of the image representation, two common approaches are used. The first approach is the clipping method, which limits pixel values to remain within the range from 0 to 255 such that the convolution results can be visualized as conventional grayscale images. The second approach is normalization, which scales the convolution result values into a specific range, such as $[0,1]$ or $[-1,1]$. This range is more suitable for further processing in a CNN. Therefore, it is important to handle convolution results that fall outside the grayscale range not only to maintain image readability but also to ensure stability and consistency in the machine learning process.

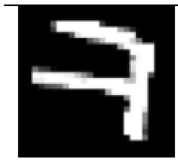
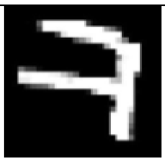
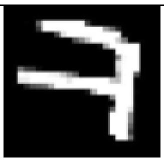
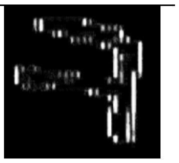
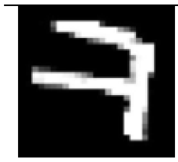
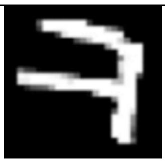
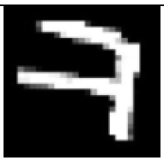
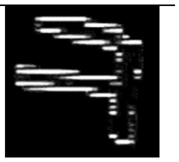
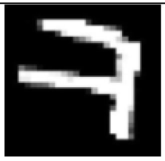
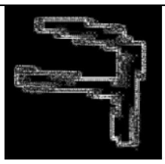
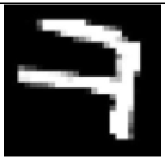
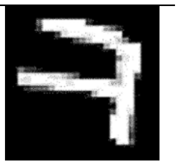
Two important concepts that determine the size and representation of convolution results are stride and padding. Stride refers to the size of the kernel shift when performing convolution operations on images. If the stride value is set to 1, then the kernel shifts one pixel each time, resulting in a relatively large and detailed feature map. Setting the stride value to greater than one, such as stride = 2, causes the kernel to skip several pixels with each shift. As a result, the feature map size is smaller, but the spatial detail is reduced. Padding is a technique of adding rows or columns of pixels to the edges of an image before convolution is performed. Padding is important in preserving output dimensions and avoiding significant reductions after each convolution operation. The most common type of padding is "valid padding", which does not add any pixels. In this approach, the size of the feature map decreases in proportion to the kernel size, which often leads to the loss of information at the image edges. Alternatively, "same padding" maintains the feature map size, allowing the CNN to learn patterns throughout the image area, including at the edges [10]. The stride and padding settings significantly affect the balance between output dimensions, computational efficiency, and the depth of spatial information captured by the network.

Consider an input feature map with dimensions (h_{in}, w_{in}, c_{in}), where h_{in} and w_{in} represent the height and width of the input, respectively, and c_{in} represents the number of channels. For grayscale images, $c_{in} = 1$, and for RGB (red-green-blue) images, $c_{in} = 3$. Given a kernel of size (h_k, w_k) with a padding of p and the stride of s, the height and width of the output can be formulated as follows:

$$h_{out} = \left\lfloor \frac{h_{in} - h_k + 2p}{s} \right\rfloor + 1 \text{ and } w_{out} = \left\lfloor \frac{w_{in} - w_k + 2p}{s} \right\rfloor + 1, \tag{4}$$

where $\lfloor \cdot \rfloor$ denotes the floor function. In the convolution layer, the number of output channels (c_{out}) equals the number of filters (k) used. Thus, the output size of the convolution result is (h_{out}, w_{out}, k). In the pooling layer, however, the number of channels does not change; thus, the output of the pooling layer has $(h_{out}, w_{out}, c_{in})$ dimensions. This formulation is important for determining the CNN network architecture because it is directly related to computational complexity and the number of parameters produced.

TABLE 1. EXAMPLE OF IMAGE CONVOLUTION USING DIFFERENT FILTERS

Input image	Operation/ filter	Convolved image	Input image	Operation/ filter	Convolved image
	Identity: $\begin{bmatrix}0 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}$			Sobel X: $\begin{bmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{bmatrix}$	
	Gaussian: $\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\2 & 4 & 2\\1 & 2 & 1\end{bmatrix}$			Sobel Y: $\begin{bmatrix}-1 & -2 & -1\\0 & 0 & 0\\1 & 2 & 1\end{bmatrix}$	
	Laplacian: $\begin{bmatrix}0 & 1 & 0\\1 & -4 & 1\\0 & 1 & 0\end{bmatrix}$			Sharpen: $\begin{bmatrix}0 & -1 & 0\\-1 & 5 & -1\\0 & -1 & 0\end{bmatrix}$	

The choice of kernel function is crucial in determining the information that can be extracted from an image. Kernels, or filters, serve as small matrices that move across an image to capture specific local patterns [25]. For example, the identity kernel preserves the central pixel value without altering the contribution of neighbouring pixels. The Sobel kernel detects horizontal or vertical edges, which helps recognize object boundaries. The Gaussian kernel works as a smoothing agent, reducing noise and making feature detection more stable. Conversely, the Laplacian kernel detects sharp changes in intensity in various directions; therefore, it is often used to extract complex edges. The sharpen kernel highlights fine details and contours, which can enhance differences between objects. Table 1 provides a comparison of the results of handwritten image convolution using several common filters in digital image processing. In the context of CNN, the kernels used in the initial layers often resemble these classical functions. However, in practice, the filters are learned automatically through the training process. CNNs have the capability for the production of kernels with increased adaptability to specific data and tasks, varying from simple edge detection to complex patterns such as textures or object shapes.

Similar to convolutional layers, pooling layers play an important role in CNN architecture. Pooling layers primarily reduce the spatial dimensions of the feature map resulting from convolution. This reduction process seeks to minimize computational complexity and the number of parameters while reducing the risk of overfitting without losing important image information. Hence, pooling layers contribute to training efficiency while strengthening the model's generalization capabilities. Max pooling and average pooling are the two most frequently used types of pooling operations. Max pooling extracts the maximum value from the image area covered by the kernel, emphasizing dominant features such as edges or strong textures. Meanwhile, average pooling calculates the average value of the area, resulting in a smoother representation that retains more comprehensive information. The choice of pooling type depends on the analysis objectives and data attributes. Max pooling is generally used in the feature detection stage, while average pooling is often applied to preserve the global information distribution.

Given a pooling $\mathbf{P}$ of size $h_p \times w_p$. The max-pooling and avg-pooling operations are determined by

$$P_{i,j} = \max_{\substack{0 \le m \le h_p \\ 0 \le m \le w_p}} A_{i \cdot h_p + m,\, j \cdot w_p + n} \quad \text{and} \quad P_{i,j} = \operatorname*{avg}_{\substack{0 \le m \le h_p \\ 0 \le m \le w_p}} A_{i \cdot h_p + m,\, j \cdot w_p + n}. \tag{5}$$

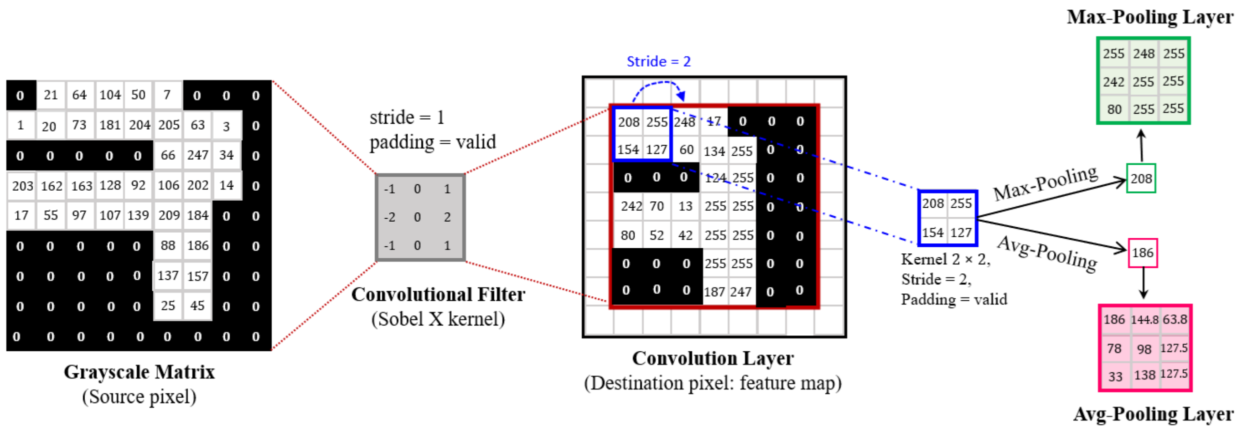


FIGURE 4. The sequence of operations in CNN involving convolution and pooling

Figure 4 presents the feature extraction stages in CNN. First, the input digital image is converted into a grayscale matrix and convolved with a Sobel X filter. The filter uses a stride of 1 and valid padding to produce a feature map. Next, a pooling process is performed with a 2×2 kernel, a stride of 2, and a valid padding. Max pooling takes the maximum value in each patch, while average pooling takes the average value. This process aims to reduce dimensions, suppress redundancy, and retain the input image's important characteristics.

Following the completion of the feature extraction stage, the CNN architecture transitions to the classification stage. Before reaching the fully connected layer, a flattening process is employed, which transforms the three-dimensional feature map resulting from convolution and pooling into a one-dimensional vector [26], [27]. CNN's dense layer is only compatible with input in vector form, not spatial matrices. In other words, flattening serves as a bridge that connects spatial representations to vector representations.

2.5 Proposed Architecture: Custom LiteBinaryNet

This study proposes Custom LiteBinaryNet, a lightweight model for binary classification of handwriting (dyslexic vs. normal). The architecture is designed for a balance between computational complexity and feature extraction capabilities. It generally consists of three main stages, including feature extraction, regularization, and classification. The Custom LiteBinaryNet architecture is structured into two main variants: a baseline model and a tuned model. The baseline network consists of three convolutional-pooling blocks with ReLU activation, followed by a flattening process and two fully connected layers for classification. To enhance performance, the configuration is refined through hyperparameter tuning, including adjusting the number of neurons in the dense layer, setting the dropout rate, and selecting an optimization algorithm.

In order to ensure optimal generalization performance, the proposed Custom LiteBinaryNet architecture implements several regularization techniques, such as dropout, $L2$ weight decay, and early stopping. These techniques are specifically designed to reduce overfitting by limiting model complexity and stabilizing the training process. In addition, hyperparameter tuning is also performed to optimize overall model performance. This adjustment involves searching for the best configuration of parameters such as learning rate, batch size, number of filters, kernel size, and regularization strength.

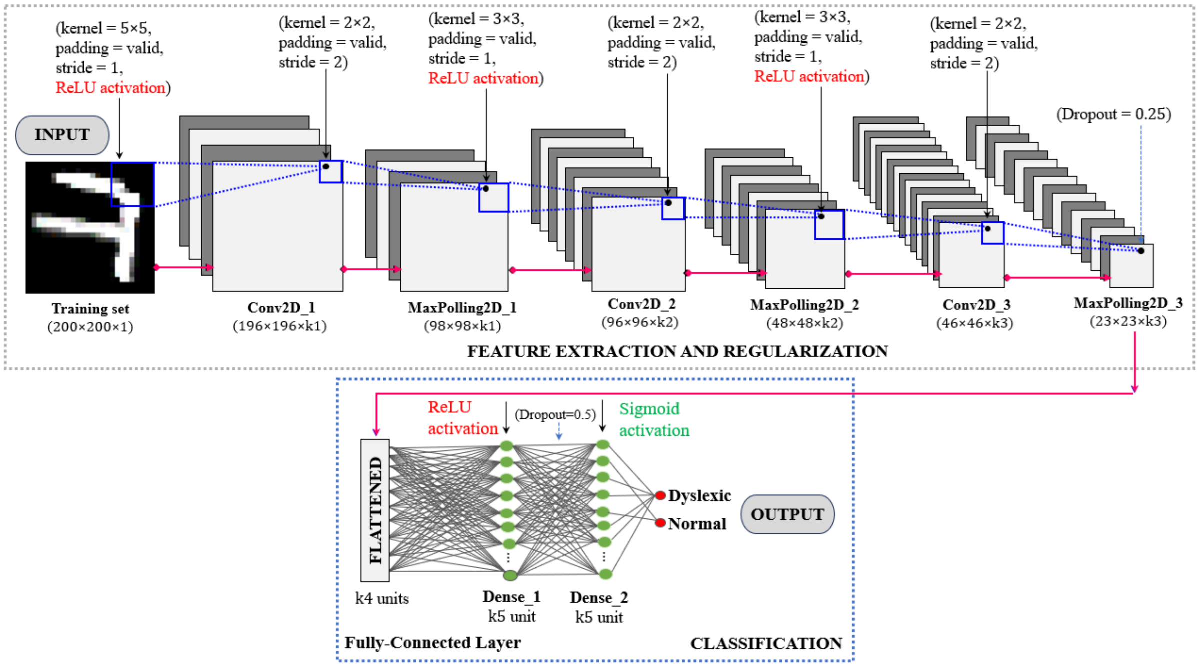

(kernel = 5×5, padding = valid, stride = 1, ReLU activation)
(kernel = 2×2, padding = valid, stride = 2)
(kernel = 3×3, padding = valid, stride = 1, ReLU activation)
(kernel = 2×2, padding = valid, stride = 2)
(kernel = 3×3, padding = valid, stride = 1, ReLU activation)
(kernel = 2×2, padding = valid, stride = 2)
(Dropout = 0.25)
INPUT
Training set (200×200×1)
Conv2D_1 (196×196×k1)
MaxPolling2D_1 (98×98×k1)
Conv2D_2 (96×96×k2)
MaxPolling2D_2 (48×48×k2)
Conv2D_3 (46×46×k3)
MaxPolling2D_3 (23×23×k3)
FEATURE EXTRACTION AND REGULARIZATION
ReLU activation
(Dropout=0.5)
Sigmoid activation
FLATTENED
Dyslexic
Normal
OUTPUT
k4 units
Dense_1 k5 unit
Dense_2 k5 unit
Fully-Connected Layer
CLASSIFICATION

FIGURE 5. CNN architecture for dyslexic handwriting classification using Custom LiteBinaryNet

During training, dropout layers with rates of 0.5 and 0.25 were incorporated into the architecture. As a regularization strategy, dropout randomly suppresses a proportion of neuron activations, thereby reducing overfitting and mitigating excessive dependency on specific neurons within the network. Suppose $\ell_i = f\left(y_i^{(1)}\right)$ is the activation of the i-th neuron in the hidden layer that occurs after the ReLu activation function and before dropout, where f is the ReLU function. The dropout can be conceptualized as a transition of ℓ_i to $\tilde{\ell}_i = \ell_i \cdot z_i$, where z_i follows a Bernoulli distribution with probability p, representing the probability of retaining the neuron. For instance, when $p = 0.5$ or $p = 0.75$, as determined by the specified dropout rate.

In binary classification, the prevailing loss function is binary cross-entropy (BCE). This function measures the difference between the true label in class i, denoted by $y_i \in \{0,1\}$, and the predicted probability $\hat{y}_i = \sigma(\mathbf{z})_i$ generated by the model. The BCE function can be expressed by

$$\mathcal{L}_{BCE}(y_i, \hat{y}_i) = -[y_i \cdot log(\hat{y}_i) + (1 - y_i) \cdot log(1 - \hat{y}_i)]. \tag{6}$$

The loss will decrease when $\hat{y}_i$ is close to 1 if the true label is $y_i = 1$ and when $\hat{y}_i$ is close to 0 if the true label is $y_i = 0$. For a dataset with N samples, the total loss is calculated using the following average.

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N} -[y_i \cdot log(\hat{y}_i) + (1 - y_i) \cdot log(1 - \hat{y}_i)]. \tag{7}$$

BCE is widely used for binary classification because it conforms to the Bernoulli probabilistic model, making it mathematically equivalent to maximum likelihood estimation (MLE) in binary distributions. $L2$ regularization (weight decay) was applied to specific layers to penalize large weights. The total loss function is defined as

$$\mathcal{L}_{total} = \mathcal{L}_{BCE} + \lambda \sum_{i} \omega_i^2, \tag{8}$$

where ω_i is the network weight, and λ is the regularization coefficient. This regularization encourages an even distribution of weights, simplifying the model and enabling better generalization. Additionally, early stopping is implemented as a callback, serving as a mechanism for the early termination of the training process if the loss value in the validation data does not manifest a significant improvement after a specified number of epochs. Early stopping can be conceptualized as an optimal time selection function, denoted by t^*.

$$t^* = \arg\min_t \mathcal{L}_{val}(t). \tag{9}$$

Here, $\mathcal{L}_{val}(t)$ denotes the validation loss value at epoch t. This process yields a tuned model that achieves higher classification accuracy without increasing network complexity. Collectively, the baseline and tuned versions of Custom LiteBinaryNet demonstrate a trade-off between computational efficiency and prediction accuracy in detecting handwriting patterns of children with dyslexia. Furthermore, the model training process adopted the Adam (Adaptive Moment Estimation) algorithm as the optimization method. Adam generated adaptive and efficient parameter updates, thus assisting the performance of the Custom LiteBinaryNet architecture in predicting the handwriting of children with and without dyslexia.

Moreover, the Custom LiteBinaryNet architecture was developed as a probabilistic predictive model, rather than as a simple binary classifier. This model produces both discrete labels indicating whether a child has dyslexia or not and continuous probability scores representing the likelihood that a child has dyslexia based on handwriting input. Thus, this approach enables the system to function as a screening tool that provides more informative risk estimates than conventional binary outputs.

2.6 Evaluation Metrics

Model performance is evaluated using accuracy, precision, recall, $F1$-score, receiver operating characteristic (ROC) curve, and area under the curve (AUC), which are calculated from the confusion matrix. The calculation formulas are as follows in Eqs. (10) to (13):

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{10}$$

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{11}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{12}$$

$$F1 - \text{score} = \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{13}$$

In this case, TP (true positive) and TN (true negative) represent the number of accurate predictions for the positive and negative classes, respectively. FP (false positive) and FN (false negative) indicate the number of incorrect predictions. Accuracy measures the proportion of correct predictions compared to all test data. Precision shows the accuracy of the model in classifying positive samples. Recall (sensitivity) measures the model's ability to detect correct positive samples to avoid false negative cases. F1-score is a measure that balances precision and recall to assess model performance when the data is imbalanced. These metrics provide a comprehensive overview of the model's performance. Additionally, the ROC curve is constructed from pairs of true positive rate (TPR) and false positive rate (FPR) values at various thresholds τ.

(14) $$\text{TPR}(\tau) = \frac{TP(\tau)}{TP(\tau) + FN(\tau)} = \frac{\text{True positives}}{\text{All actual positives}},$$

(15) $$\text{FPR}(\tau) = \frac{FP(\tau)}{FP(\tau) + TN(\tau)} = \frac{\text{False positives}}{\text{All actual negatives}}.$$

By varying τ from 0 to 1, the ROC curve is obtained as follows

(16) $$\text{ROC} = \{(\text{FPR}(\tau), TP(\tau)) | \tau \in [0,1]\}.$$

The area under the ROC curve is defined as

(17) $$\text{AUC} = \int_0^1 TPR\left(FPR^{-1}(\tau)\right) d\tau.$$

The AUC value lies within the range of $[0,1]$, where higher values reflect a greater discriminatory capacity of the model. In the context of early dyslexia screening, AUC serves as a metric for evaluating the reliability of the Custom LiteBinaryNet model, particularly in capturing the trade-off between sensitivity and specificity.

2.7 Explainable Artificial Intelligence (XAI)

The CNN architecture has proven effective in extracting features and performing classification on visual data via convolution layers, pooling, and fully connected layers. However, this model frequently poses interpretability challenges due to the high computational demands of CNNs, which hinder the transparent understanding of the connection between the input and output [14]. Building on these challenges, XAI has been developed to improve the transparency, interpretability, and trustworthiness of AI models [16], [28].

One of the primary objectives of this study is to integrate XAI and CNN to enhance the interpretability of the CNN model architecture, particularly the Custom LiteBinaryNet. In the framework of handwriting prediction, it becomes suitable to apply the XAI post-hoc techniques, such as Grad-CAM and OSA. The main idea behind Grad-CAM is to use the gradient information from the target class score against the feature map on the last convolutional layer. Grad-CAM generates a heatmap highlighting the areas of an image most influential to a CNN's decision [29]. For instance, Grad-CAM can demonstrate that the model focuses more on the lower curve than the vertical stem of the letter "b." This visualization allows researchers to evaluate the importance of the features extracted by the CNN and provides a clearer interpretation of the classification process. Meanwhile, OSA is a feature attribution-based post-hoc explanation method that evaluates the contribution of each part of the input to the model's prediction by occluding small regions of the input image. It operates by systematically masking portions of the image and measuring the corresponding changes in the model's prediction. The more critical a region is for the model's decision, the larger the decrease in the prediction score when that region is occluded.

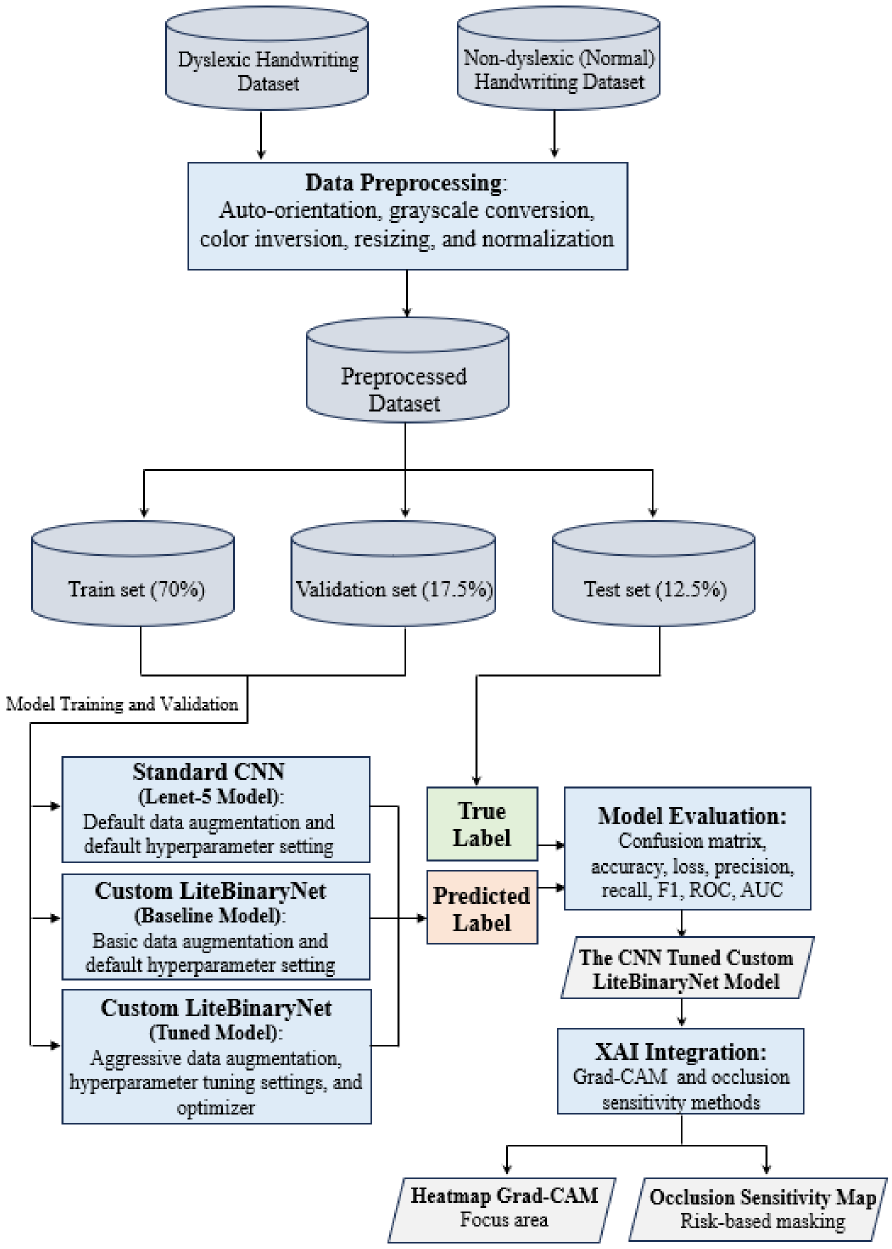

Dyslexic Handwriting Dataset
Non-dyslexic (Normal) Handwriting Dataset
Data Preprocessing: Auto-orientation, grayscale conversion, color inversion, resizing, and normalization
Preprocessed Dataset
Train set (70%)
Validation set (17.5%)
Test set (12.5%)
Model Training and Validation
Standard CNN (Lenet-5 Model): Default data augmentation and default hyperparameter setting
Custom LiteBinaryNet (Baseline Model): Basic data augmentation and default hyperparameter setting
Custom LiteBinaryNet (Tuned Model): Aggressive data augmentation, hyperparameter tuning settings, and optimizer
True Label
Predicted Label
Model Evaluation: Confusion matrix, accuracy, loss, precision, recall, F1, ROC, AUC
The CNN Tuned Custom LiteBinaryNet Model
XAI Integration: Grad-CAM and occlusion sensitivity methods
Heatmap Grad-CAM Focus area
Occlusion Sensitivity Map Risk-based masking

FIGURE 6. XAI-driven Custom LiteBinaryNet-CNN flowchart for dyslexia handwriting prediction

The proposed XAI-based CNN framework for predicting dyslexic handwriting is exhibited in the flowchart in Figure 6. The process begins with data preprocessing, including auto-orientation, grayscale conversion, color inversion, rescaling, and normalization, followed by splitting the dataset into training, validation, and test data. The baseline Custom LiteBinaryNet model with basic augmentation and the tuned Custom LiteBinaryNet model with aggressive augmentation and hyperparameter optimization are compared with the standard LeNet-5 model employed as a conventional CNN benchmark. The LeNet-5 architecture consists of two convolutional layers with 5×5 kernels, a subsampling layer with average pooling, and two fully connected layers with sigmoid/tanh activation functions, without using dropout regularization or hyperparameter tuning [30].

3. MAIN RESULTS

This section presents the findings of the proposed XAI-driven Custom LiteBinaryNet for dyslexic handwriting prediction. The results are further compared with previous studies to highlight methodological improvements and practical relevance in educational and clinical contexts

3.1 Experimental Environment

The experiment was conducted in a computing environment with an NVIDIA A100-SXM4-40GB GPU (40 GB VRAM) and CUDA 12.4 support. It was implemented using Python 3.10, TensorFlow 2.12, and Keras 2.12.

3.2 Custom LiteBinaryNet Model Training Results

The Custom LiteBinaryNet model was trained to classify handwritten images of children with and without dyslexia. During this process, key metrics, including training accuracy, validation accuracy, and loss values, were generated and used to monitor the model's stability and generalization capabilities. The training results curve depicted the convergence dynamics of the model from one epoch to the next. These results are used to evaluate the final model before testing on the test dataset.

The accuracy curves in Figure 7 provide a clear comparison of the three models. The LeNet-5 Model achieves moderate performance, with a maximum training accuracy of approximately 0.719, while the validation accuracy remains relatively stable around 0.67–0.68, suggesting that the model can learn the training data but with limited ability to generalize. The baseline model achieves a considerably higher training accuracy, exceeding 0.90. However, the validation accuracy fluctuates significantly and does not follow the training trend, which strongly indicates overfitting. In contrast, the tuned model demonstrates consistent improvement, as training accuracy stabilizes at around 0.85, and validation accuracy increases progressively, surpassing 0.80 by the final epochs. These results highlight that the tuned model achieves the best trade-off between training performance and validation stability, thereby providing the best generalization capability compared to the other two models.

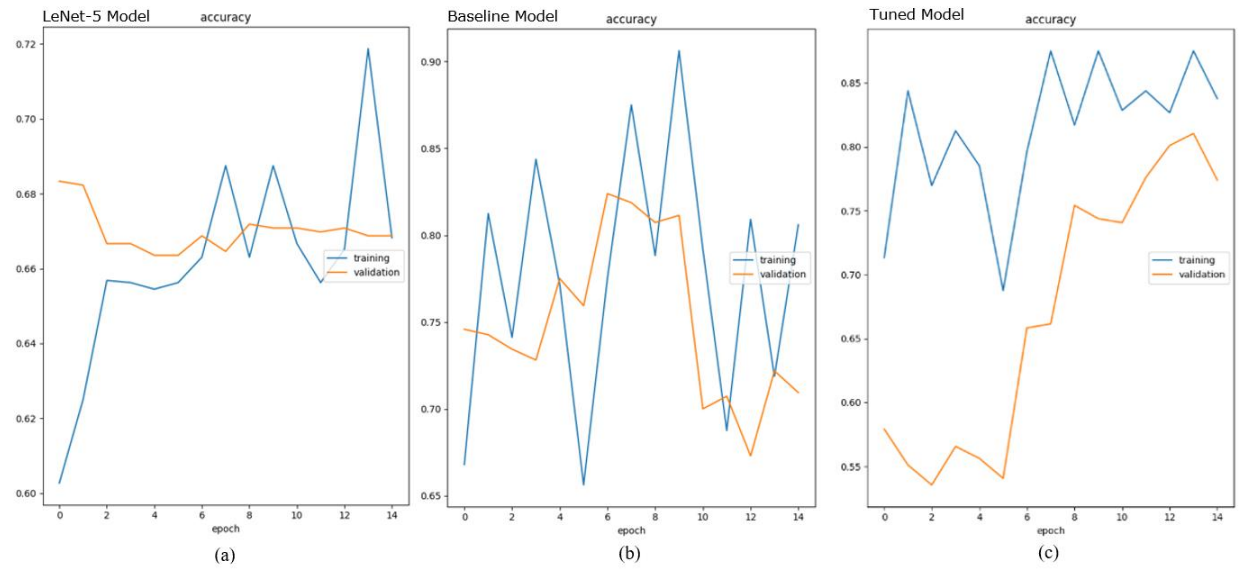


FIGURE 7. Comparison of training and validation accuracy curves across three models: (a) LeNet-5 model, (b) baseline Custom LiteBinaryNet model, and (c) tuned Custom LiteBinaryNet model, across 15 training epochs

The loss curves in Figure 8 further support the findings from the accuracy analysis. For the LeNet-5 model, training loss decreases steadily with epochs, while validation loss remains relatively flat, which indicates limited overfitting but insufficient improvement on unseen data. The baseline model shows a sharp reduction in training loss, whereas validation loss remains nearly constant across epochs, which confirms the presence of overfitting observed in the accuracy curves. In contrast, the tuned model presents a steady decrease in training loss together with stable validation loss. Although the validation loss does not decrease significantly, the model's stability is enhanced. In addition, the steadily increasing validation accuracy suggests that the model is better optimized and more robust.

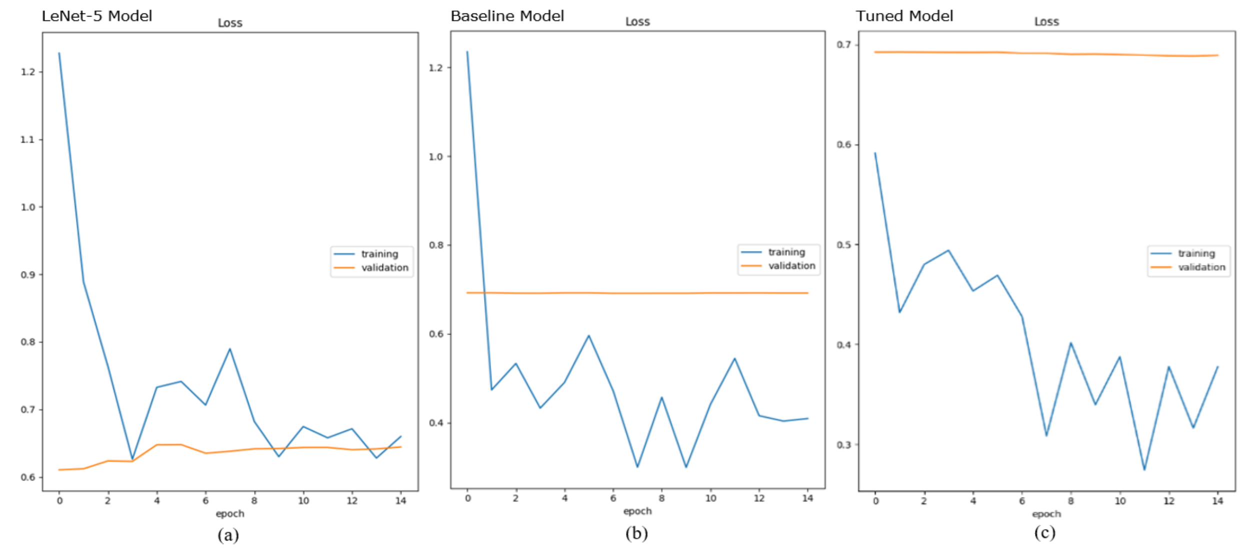


FIGURE 8. Comparison of training and validation loss curves across three models: (a) LeNet-5 model, (b) baseline Custom LiteBinaryNet model, and (c) tuned Custom LiteBinaryNet model, across 15 training epochs

The quantitative results in Table 2 reinforce these observations. The LeNet-5 model exhibits relatively low training from 0.603 to 0.719 and validation accuracies range from 0.510 to 0.617, with a minimal train-validation gap of about 0.001, which reflects limited learning capacity. The baseline model achieves higher accuracies (training $0.656-0.906$ and validation $0.673-0.824$), but the train-validation gap is the widest, reaching about 0.097, which substantiates the overfitting behavior observed in Figures 7 and 8. The tuned model, on the other hand, reaches the highest accuracy ranges ($0.688-0.875$ for training and $0.535-0.810$ for validation), with a moderate train-validation gap of about 0.064, indicating a better compromise between learning and generalization.

In terms of model complexity, the standard (LeNet-5) model has 25.61 million parameters with a storage size of 97.71 MB. Despite its large parameter count, the model shows relatively limited accuracy improvement and almost no train-validation gap (~ 0.001), suggesting underfitting due to inefficient parameter utilization. The baseline model is more compact, with 13.07 million parameters and a storage size of 49.88 MB. However, the reduced complexity comes at the cost of overfitting, as indicated by the wide train-validation accuracy gap (~ 0.097). The tuned model represents the most complex configuration, with 39.22 million parameters and a storage requirement of 149.64 MB. Despite this larger size, the model demonstrates the best trade-off between accuracy and generalization, supported by the integration of customized regularization and systematic hyperparameter tuning. Techniques such as dropout, $L2$ weight decay, and early stopping were employed to mitigate overfitting by preventing the model from relying excessively on specific neurons, penalizing large weights, and halting training once validation performance stagnated. Furthermore, hyperparameter tuning was carried out through targeted adjustments of key parameters, including the learning rate, batch size, number of filters, kernel size, and regularization strength. This combination of strategies led to a model that not only achieved higher accuracy but also maintained more stable validation loss, thereby striking an optimal balance between learning capacity and generalization.

TABLE 2. COMPARISON OF MODEL TRAINING RESULTS

Aspects	Standard (LeNet-5) model	Baseline model	Tuned model
Training accuracy (min−max)	$0.603 - 0.719$	$0.656 - 0.906$	$0.688 - 0.875$
Validation accuracy (min−max)	$0.664 - 0.669$	$0.673 - 0.824$	$0.535 - 0.810$
Last epoch accuracy (train/val)	$0.668/0.667$	$0.806/0.709$	$0.838/0.774$
Last epoch loss (train/val)	$0.653/0.644$	$0.409/0.692$	$0.377/0.689$
Gap train−val accuracy	~ 0.001	~ 0.097	~ 0.064
Trainable parameters	25,613,494	13,075,937	13,075,937
Non-trainable parameters	0	0	0
Total parameters	25,613,494	13,075,937	39,227,813
Model size (storage)	97.71 MB	49.88 MB	149.64 MB

Generally, the comparative analysis demonstrates that increasing model complexity alone, as in the baseline model, does not guarantee improved performance, as it may exacerbate overfitting. Instead, incorporating regularization techniques alongside hyperparameter optimization yields a tuned model that achieves superior predictive accuracy, enhanced stability, and robust generalization. From a practical standpoint, these findings highlight the critical role of careful model tuning in producing architectures suitable for deployment in real-world applications, while from a theoretical perspective, they reinforce the importance of principled regularization in the mathematical modeling of neural networks.

3.3 Comparative Performance Evaluation of the Custom LiteBinaryNet

To further examine the classification capability of the proposed Custom LiteBinaryNet model, a confusion matrix analysis is conducted and compared against the standard LeNet-5 and the baseline configuration. Confusion matrices enable a more detailed inspection of class-level predictions by highlighting the distribution of true positives, false positives, true negatives, and false negatives. This comparison is crucial in the context of dyslexia prediction, where misclassification of either normal or dyslexic cases could lead to significant practical implications. Figure 9 illustrates the confusion matrices of the three evaluated models, providing a visual comparison of their predictive reliability and error distribution.

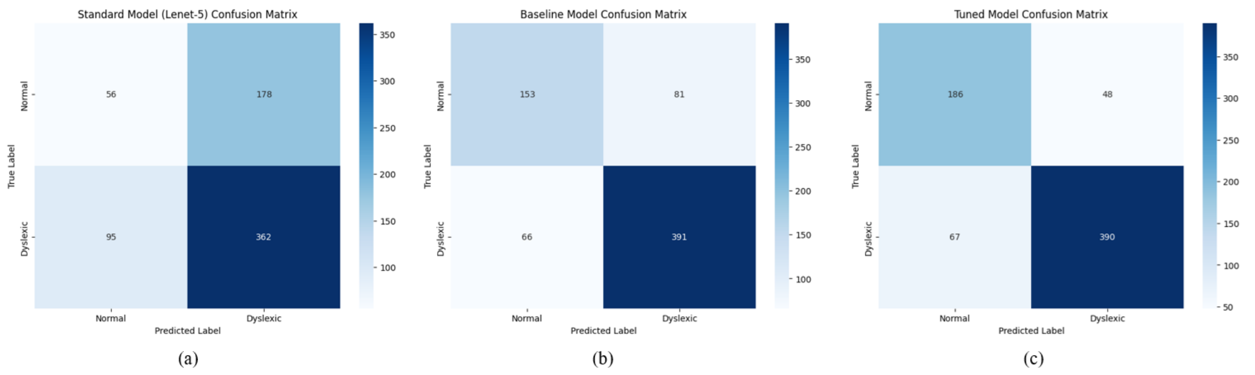


FIGURE 9. Confusion matrices of the three evaluated models: (a) LeNet-5 model, (b) baseline Custom LiteBinaryNet model, and (c) tuned custom LiteBinaryNet model

Figure 9 compares classification performance across models for dyslexia prediction. The LeNet-5 (a) exhibits poor separation between normal and dyslexic classes, with high misclassification of normal samples as dyslexic. The baseline Custom LiteBinaryNet (b) improves class balance with fewer false negatives, though misclassification of normal cases remains considerable. The tuned Custom LiteBinaryNet (c) achieves the most balanced performance, reducing both false positives and false negatives, thereby yielding the highest predictive reliability. From the confusion matrices, LeNet-5 misclassifies a substantial proportion of normal cases (178 out of 234) as dyslexic, leading to low specificity and an overall accuracy of 60.49%. The Baseline Custom LiteBinaryNet markedly reduces this misclassification, correctly identifying 153 normal cases and 391 dyslexic cases, resulting in an improved accuracy of 78.73%. The Tuned Custom LiteBinaryNet further enhances classification, correctly predicting 186 normal and 390 dyslexic samples with only minor misclassifications, thereby achieving the highest accuracy of 83.36%. These results confirm that the tuned model provides the best trade-off between sensitivity and specificity, while ensuring more reliable dyslexia prediction compared to both LeNet-5 and the baseline model.

TABLE 3. COMPARISON OF MODEL EVALUATION METRICS

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
LeNet-5	60.49	67.04	79.21	72.62
Baseline	78.73	82.84	85.56	84.18
Tuned	83.36	89.04	85.34	87.15

The evaluation results in Table 3 show that the LeNet-5 model achieved only 60.49% accuracy with an F1-score of 72.62%, highlighting its limitations in handling dyslexia classification tasks. The baseline Custom LiteBinaryNet demonstrated a substantial improvement, reaching 78.73% accuracy and an F1-score of 84.18%, which indicates that a lighter architecture with regularization provides better generalization. The tuned Custom LiteBinaryNet achieved the best performance with 83.36% accuracy and an F1-score of 87.15%, while its precision reached 89.04% and recall reached 85.34%, showing more balanced predictive capability. These results confirm that hyperparameter optimization plays a critical role in enhancing model performance, making the tuned Custom LiteBinaryNet more reliable for dyslexia prediction compared to the other two models.

In addition, the predictive reliability of the models was further assessed through ROC curves and their corresponding AUC values. These metrics provide a comprehensive evaluation by capturing the trade-off between sensitivity and specificity across varying thresholds. The ROC and AUC analysis highlights significant performance differences among the three models. The standard LeNet-5 model shows limited discriminative capability with an AUC of 0.56, indicating performance close to random guessing. The baseline Custom LiteBinaryNet substantially improves the classification with an AUC of 0.87, reflecting the effectiveness of adopting a lighter architecture with regularization. The tuned Custom LiteBinaryNet achieves the highest AUC of 0.91, confirming that hyperparameter optimization further strengthens the model's ability to separate positive and negative cases. These findings emphasize that the tuned Custom LiteBinaryNet not only achieves the best accuracy and loss performance but also delivers superior robustness in terms of overall discriminative power.

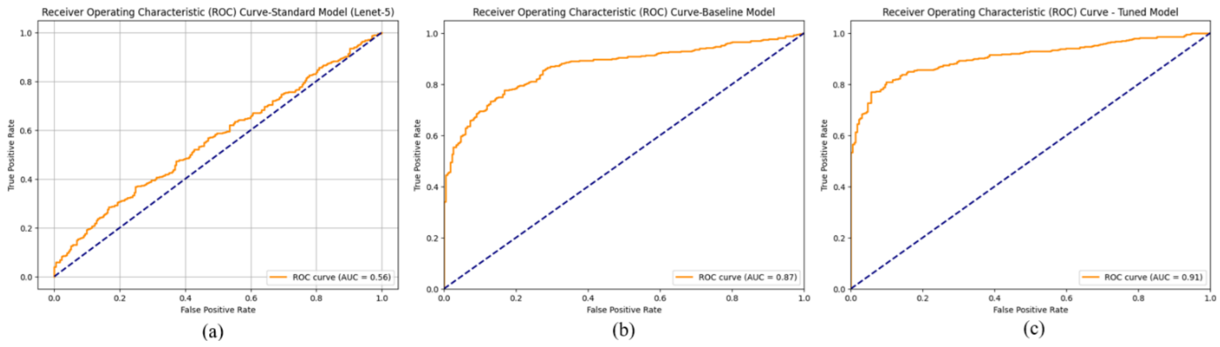


FIGURE 10. ROC curves of the three evaluated models: (a) LeNet-5 model, (b) baseline Custom LiteBinaryNet model, and (c) tuned custom LiteBinaryNet model

However, the superior performance of the tuned Custom LiteBinaryNet cannot be explained by numerical metrics alone, as these do not fully reflect the complexity of the model's learning process in handwriting recognition. To provide deeper insights, feature maps were visualized across convolution and pooling layers. As shown in Figure 11, the tuned model progressively extracts discriminative features from the image of the letter "K," starting with the global outline in the early layers and advancing to more abstract spatial representations in the deeper layers. This hierarchical process enables the network to refine simple shapes, such as lines and branches resembling "K" or "X," into complex patterns that capture distortions commonly found in dyslexic handwriting. These include disproportionate letter sizes, inconsistent strokes, unstable slanting, and letter reversals, where letters deviate from standard forms or appear with incorrect orientation. Such features, often observed in children with dyslexia, provide a crucial basis for distinguishing between typical and dyslexic handwriting [7], [31].

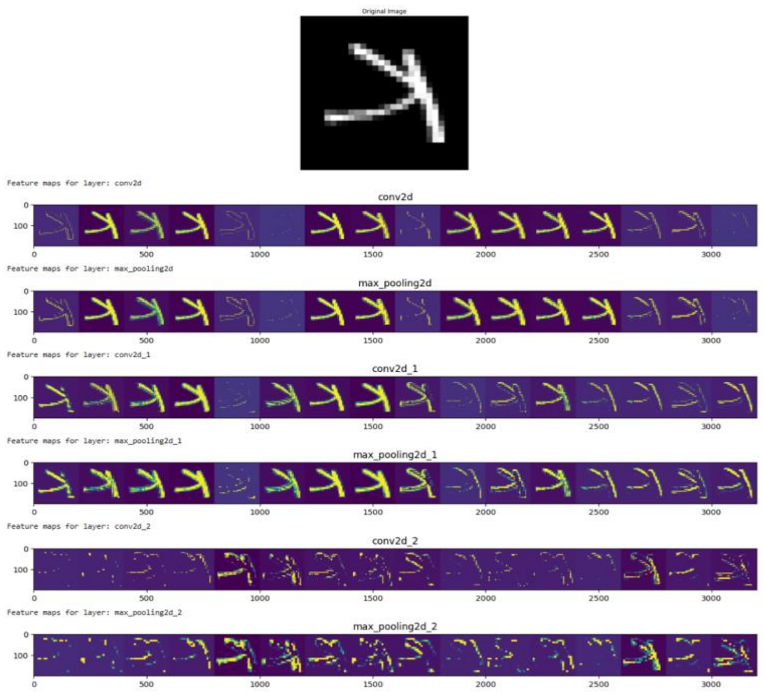

Original Image
Feature maps for layer: conv2d
conv2d
Feature maps for layer: max_pooling2d
max_pooling2d
Feature maps for layer: conv2d_1
conv2d_1
Feature maps for layer: max_pooling2d_1
max_pooling2d_1
Feature maps for layer: conv2d_2
conv2d_2
Feature maps for layer: max_pooling2d_2
max_pooling2d_2

FIGURE 11. Feature map visualization of CNN layers in the Custom LiteBinaryNet architecture for the letter "K"

In the context of dyslexia detection, the hierarchical mechanism is a pertinent factor, as it enables the model to recognize variations in letter shapes and distinctive handwriting patterns, such as disproportionate letter sizes, inconsistent strokes, unstable slanting, and distorted shapes. Such distortions manifest in the form of letters deviating from the standard shape, incorrect orientation, or letter reversal, as exemplified by the letter "K" written with reversal, as seen in the original image. These patterns are common characteristics of the handwriting of children with dyslexia and are an important basis for the model to distinguish normal handwriting from dyslexic handwriting.

3.4 Prediction Results of the Custom LiteBinaryNet Model

Among the three evaluated models, the tuned Custom LiteBinaryNet demonstrated the most balanced and consistent performance, making it the most reliable for further analysis. Focusing on this model ensures that the prediction results accurately reflect its ability to recognize handwriting patterns associated with dyslexia, while avoiding potential biases from less optimized architectures. A visual representation of classification outcomes is therefore provided to observe how the model performs at the individual sample level. As illustrated in Figure 12, the predictions are displayed alongside labels and probability scores, where values approaching 1 indicate strong confidence in identifying a sample as dyslexic, and lower values correspond to normal handwriting.

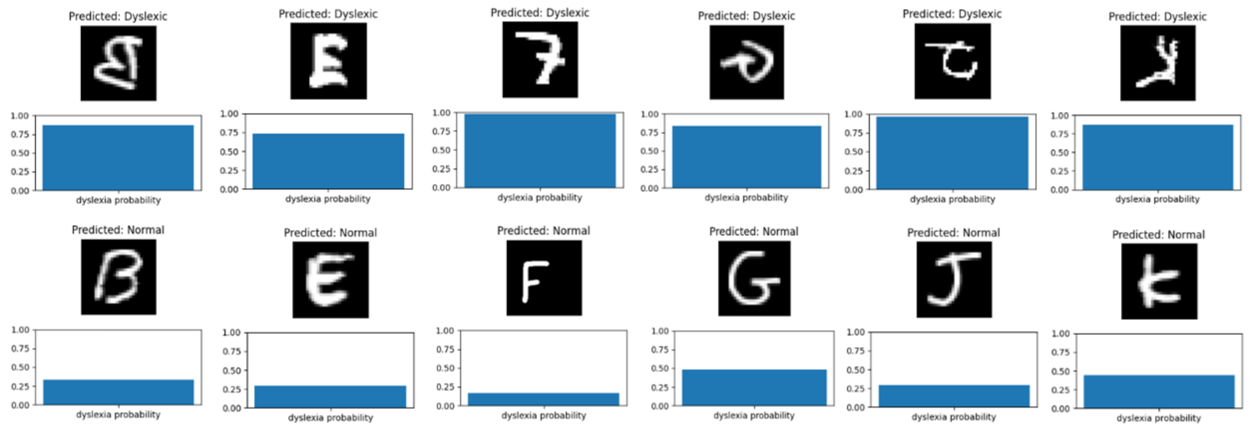
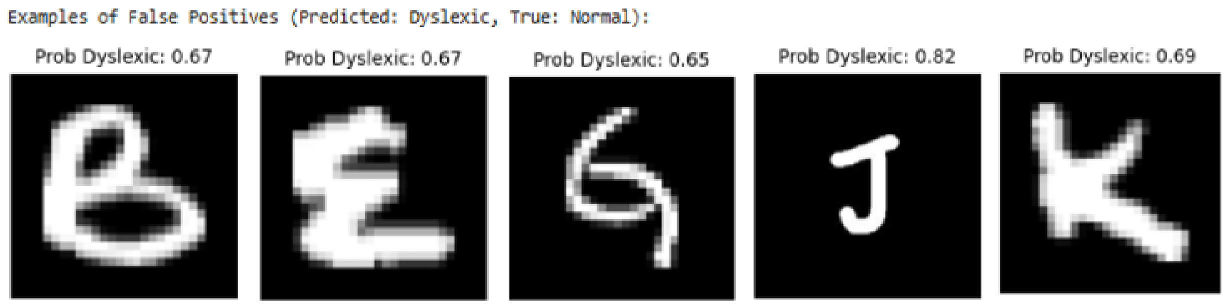
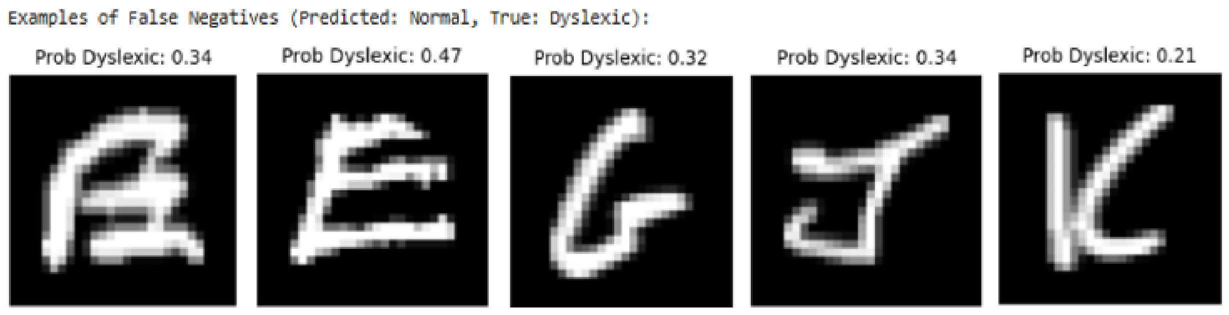


FIGURE 12. The tuned Custom LiteBinaryNet model prediction on handwritten samples

The tuned Custom LiteBinaryNet model, a machine learning algorithm designed to distinguish dyslexic and normal handwriting, utilizes distinctive letter shape features to achieve this distinction. In the context of dyslexic handwriting styles, observable characteristics include distorted and disproportionate letter shapes, as well as inconsistent stroke directionality. Examples of such errors include letters with excessively slanted, broken, or seemingly similar strokes that appear as other letters. Conversely, normal handwriting is characterized by stable, proportional letter shapes and consistent stroke directions, which leads to a lower classification probability within the dyslexia category. The model's classification is predicated on the assumption that the distinguishing factors include spatial characteristics, such as letter size, line straightness, curve consistency, and structural regularity. Following the presentation of true predictions on numerous handwriting samples, it is imperative to examine instances of misclassification that arise within the model. This analysis offers insight into the model's limitations by examining its misclassification and highlighting its inability to recognize particular handwriting patterns and identify features prone to ambiguity.



(a)



(b)

FIGURE 13. Examples of false positives and false negatives in the tuned Custom LiteBinaryNet model predictions

Despite the tuned Custom LiteBinaryNet model's promising performance in differentiating between normal and dyslexic handwriting, the visualization outcomes depicted in Figure 13 reveal instances of ambiguity that pose significant challenges for classification. The examples of false positives, where normal handwriting was incorrectly predicted as dyslexic, and false negatives, where dyslexic handwriting was classified as normal, highlight this limitation. These errors are particularly evident in letters with unusual shapes, distorted contours, or variations in strokes that visually resemble dyslexic handwriting, even when originating from normal samples. Conversely, some dyslexic letters displaying only mild distortions or subtle differences in slant are mistakenly identified as normal. Such patterns suggest that the model's predictions are influenced by overlapping discriminative features between normal and dyslexic handwriting, especially when irregularities are not pronounced enough to trigger a confident dyslexia prediction. To mitigate these errors, a comprehensive strategy is required, including hard example mining, data augmentation specifically tailored to dyslexic handwriting patterns, and the incorporation of supplementary features such as stroke direction and letter irregularity. By refining the feature space in this way, the model could move beyond general shape recognition and demonstrate heightened sensitivity to the subtle handwriting variations that often serve as critical indicators of dyslexia.

3.5 XAI-driven Custom LiteBinaryNet results

The performance of the Custom LiteBinaryNet model demonstrates the potential for favorable outcomes. Nevertheless, the exclusive reliance on quantitative metric evaluation may not provide a comprehensive understanding of the model's decision-making processes. In this regard, XAI-based analysis is employed to identify specific regions of the handwriting that are prioritized by the model. The visual explanation enables the evaluation of the concordance between the model's predictions and the observed patterns in the handwriting of dyslexic children. This approach offers intuitive explanations, thereby enhancing the interpretability of the classification results.

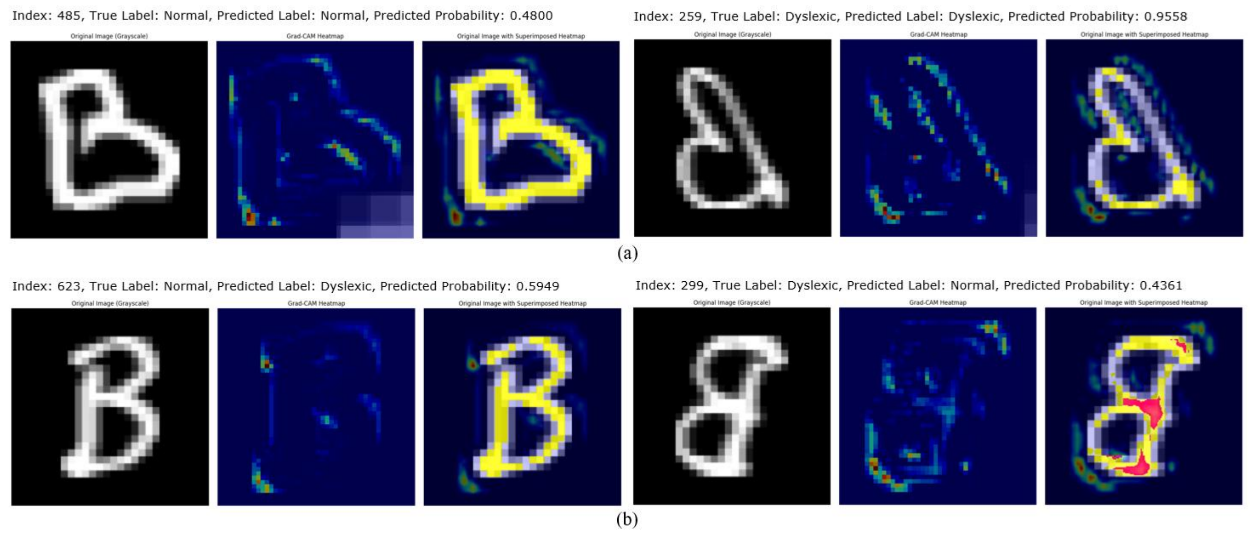


FIGURE 14. Grad-CAM visualization for dyslexia prediction through handwriting classification

The Grad-CAM visualization in Figure 14 illustrates how the tuned Custom LiteBinaryNet model directs its attention when classifying handwriting samples of dyslexic and normal children. For correctly classified cases, the heatmaps show strong activation along the primary structures of the letters, such as dominant curves and consistent horizontal strokes, which enables the model to capture meaningful patterns and produce accurate predictions. In contrast, misclassified cases reveal clear deviations in attention. False positives occur when the model emphasizes peripheral edges or additional visual artifacts that resemble dyslexic features, causing normal handwriting to be incorrectly labeled. False negatives appear when activation is concentrated on less informative fragments of the letter, which prevents the model from recognizing the distinctive distortions of dyslexic handwriting, such as irregular shapes, unstable strokes, or letter reversals. These findings suggest that although CNNs can extract discriminative handwriting features, they remain vulnerable to overlapping patterns between classes.

The Grad-CAM heatmaps not only provide interpretability by revealing the model's focal regions but also highlight attention biases that contribute to classification errors. Such insights are valuable for guiding improvements in architecture design and the application of targeted data augmentation strategies. Building on this interpretive analysis, the subsequent step involved occlusion sensitivity mapping to further evaluate the contribution of each image component to the model's classification decisions.

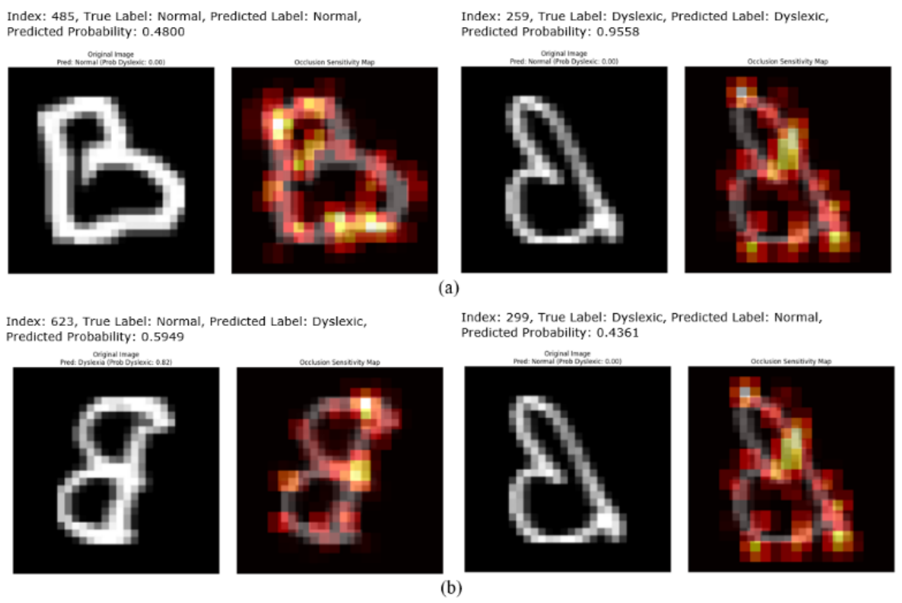

Index: 485, True Label: Normal, Predicted Label: Normal,
Predicted Probability: 0.4800
Original Image
Pred: Normal (Prob Dyslexic: 0.00)
Occlusion Sensitivity Map
Index: 259, True Label: Dyslexic, Predicted Label: Dyslexic,
Predicted Probability: 0.9558
Original Image
Pred: Normal (Prob Dyslexic: 0.00)
Occlusion Sensitivity Map
(a)
Index: 623, True Label: Normal, Predicted Label: Dyslexic,
Predicted Probability: 0.5949
Original Image
Pred: Dyslexia (Prob Dyslexic: 0.82)
Occlusion Sensitivity Map
Index: 299, True Label: Dyslexic, Predicted Label: Normal,
Predicted Probability: 0.4361
Original Image
Pred: Normal (Prob Dyslexic: 0.00)
Occlusion Sensitivity Map
(b)

FIGURE 15. Occlusion sensitivity map visualization in handwriting classification of dyslexic and normal children

Figure 15 displays the prediction results of a handwriting classification model using the Occlusion Sensitivity method. Each output shows the original image (left) and the occlusion sensitivity map (right). Figure 15(a) presents correct predictions, classified as true positive or true negative, while Figure 15(b) exhibits incorrect predictions, classified as false positive or false negative. The intensity of the red color on the sensitivity map indicates the areas of the letter that most influence the model's decision. The occlusion sensitivity map visualization illustrates how the CNN model utilizes specific components of the letter image to generate predictions. Moreover, in the handwriting example delineated in Figure 15(a), the model exhibits the capacity to furnish correct predictions for both normal and dyslexic handwriting. The sensitivity map provides a visual representation of the relevant features of the letter, including the main curves and horizontal lines. These features are identified as significant elements in the analysis and are considered discriminatory characteristics. In contrast, misclassification occurs in Fig. 16(b). A false positive prediction is characterized by an excessive allocation of model attention to additional areas or edges of the letter that bear resemblance to the predominant pattern characteristic of dyslexic handwriting. As a consequence, this leads to an incorrectly predicted normal handwriting. A false-negative prediction is characterized by the model's inability to discern specific characteristics of dyslexic handwriting, such as unstable shapes or additional strokes. Instead, the model's attention is fragmented on smaller, less relevant parts of the letter, resulting in a misclassification. Generally, the occlusion sensitivity map explains that the model's performance is contingent on its ability to prioritize informative regions of the letters, thereby highlighting its deficiencies in accommodating shape variations across classes.

The findings derived from the XAI-driven CNN with the Custom LiteBinaryNet model yield a visual representation that accentuates the handwriting elements that exert a pronounced influence on the classification process. Grad-CAM projects the model's focus on specific parts of the letter that are considered discriminatory, while Occlusion Sensitivity demonstrates the change in prediction when part of the image is occluded. The integration of these two approaches generates a coherent representation that the model assimilates, demonstrating its capacity to discern not only the global configuration of the letter but also the local characteristics, such as stroke direction, line proportions, and distortion patterns that are characteristic of the handwriting of children diagnosed with dyslexia.

3.6 Implications and Limitations

The findings of this study show that the proposed XAI-driven Custom LiteBinaryNet functions as a lightweight CNN architecture that effectively predicts handwriting patterns in children with and without dyslexia from a neurocognitive perspective. The model not only classifies handwriting with consistent accuracy but also reflects underlying neurocognitive mechanisms associated with dysgraphia related behaviors such as irregular letter formation, spacing inconsistencies, and motor instability. Its compact structure allows efficient computation even with limited datasets while maintaining high predictive performance. In addition, the use of XAI methods such as Grad CAM and Occlusion Sensitivity enhances model transparency by revealing spatial attention regions that correspond to neurocognitive handwriting traits. These visual explanations illustrate the difference in attention patterns between correct and incorrect predictions, contributing to a more interpretable understanding of how dyslexia affects handwriting.

From an applied perspective, this study carries important implications for education and healthcare, particularly in supporting early neurocognitive screening for dyslexia. XAI-driven Custom LiteBinaryNet provides both quantitative prediction results and visual explanations that can be interpreted by educators, psychologists, and clinicians. This strengthens its role as a practical diagnostic support tool that connects computational predictions with neurocognitive insights. Furthermore, because of its lightweight nature, the model can be implemented on devices with limited resources, such as tablets or school-based digital learning platforms, making it suitable for early screening applications in educational environments.

However, several limitations should be noted. First, the study relies on secondary data, which limits control over the accuracy of annotation and the neurocognitive validity of the labels. Second, the dataset lacks demographic diversity, which restricts the generalization of findings across broader populations. Third, the neurocognitive interpretations generated by the model have not yet been systematically validated by experts in cognitive psychology or neuroscience.

Future research is encouraged to develop more representative datasets that capture the neurocognitive variability of dyslexic children. Combining handwriting data with multimodal sources such as writing dynamics, eye movement, or neural activity recordings may further enhance model accuracy and neurocognitive interpretability. Exploring additional XAI techniques, such as Layer-wise Relevance Propagation or SHAP, could also strengthen the understanding of the relationship between brain processes and handwriting behavior. Finally, validation in real educational and clinical settings is essential to evaluate the ethical, practical, and diagnostic implications of AI-based neurocognitive prediction systems for dyslexia.

4. CONCLUSION

This study developed a Custom LiteBinaryNet-based CNN model capable of performing neurocognitive prediction of dyslexia through handwriting analysis. Compared with the conventional LeNet-5 benchmark, the proposed architecture demonstrated superior performance and computational efficiency, achieving an accuracy of 83.82%. The integration of XAI methods, including Grad-CAM and Occlusion Sensitivity, provided transparent neurocognitive interpretability by visualizing attention regions corresponding to letter structures and motor control patterns. Correct predictions reflected focused activation on stable features, while errors revealed dispersed attention consistent with dysgraphic irregularities. These results indicate that the XAI-driven Custom LiteBinaryNet not only enhances predictive performance but also bridges deep learning with neurocognitive understanding of handwriting formation. Consequently, this framework offers a practical, interpretable, and efficient approach to early dyslexia prediction, supporting educational and clinical decision-making with transparent, neurocognitively meaningful insights.

FUNDING STATEMENT

This research was funded by the Ministry of Higher Education, Science, and Technology of the Republic of Indonesia through the Regular Fundamental Research (PFR) grant No. 108/C3/DT.05.00/PL/2025 and supported by the Institute for Research and Community Service of Universitas Singaperbangsa Karawang (LPPM UNSIKA), under a derivative contract No. 256/UN64.10/PP/2025.

ACKNOWLEDGMENT

The authors thank the research assistant team, comprising Riyan Setiawan, Muhammad Immamul Caesar, and Muhammad Alif Shafar, for their valuable contributions in data collection, pre-processing, and administrative activities, which greatly assisted in the completion of this study. The authors express their sincere appreciation to Teruna Ria Kindergarten, Karawang, Indonesia, for their collaboration in collecting real-world data, which has served as a crucial basis for the subsequent phases of model validation and implementation. The authors are also grateful to the anonymous reviewers for their constructive comments, which helped improve the quality of the manuscript.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interests.

REFERENCES

[1] A.M. D'Mello, J.D.E. Gabrieli, Cognitive Neuroscience of Dyslexia, Lang. Speech Hear. Serv. Sch. 49 (2018), 798-809. https://doi.org/10.1044/2018_lshss-dyslc-18-0020.

[2] R. Kunwar, H.P. Sapkota, An Overview of Dyslexia: Some Key Issues and Its Effects on Learning Mathematics, Turk. Int. J. Spec. Educ. Guid. Couns. 11 (2022), 82-98.

[3] K.A. Latief, Disleksia dan Tantangan Bagi Pegiat Leterasi, in: Seminar Dukungan Kegiatan Duta Baca Provinsi Aceh, 2020. https://doi.org/10.13140/RG.2.2.16731.18720.

[4] International Dyslexia Association, Frequently Asked Questions, https://dyslexiaida.org/frequently-asked-questions-2.

[5] S. Kandel, S. Valdois, French and Spanish-Speaking Children Use Different Visual and Motor Units during Spelling Acquisition, Lang. Cogn. Process. 21 (2006), 531-561. https://doi.org/10.1080/01690960500095946.

[6] T. Asselborn, T. Gargot, Ł. Kidziński, W. Johal, et al., Automated Human-Level Diagnosis of Dysgraphia Using a Consumer Tablet, Npj Digit. Med. 1 (2018), 42. https://doi.org/10.1038/s41746-018-0049-x.

[7] R.I. Nicolson, A.J. Fawcett, Dyslexia, Dysgraphia, Procedural Learning and the Cerebellum, Cortex 47 (2011), 117-127. https://doi.org/10.1016/j.cortex.2009.08.016.

[8] G. Aldehim, M. Rashid, A.S. Alluhaidan, S. Sakri, S. Basheer, Deep Learning for Dyslexia Detection: A Comprehensive CNN Approach with Handwriting Analysis and Benchmark Comparisons, J. Disabil. Res. 3 (2024), 1-8. https://doi.org/10.57197/jdr-2024-0010.

[9] H.W. Liu, S. Wang, S.X. Tong, DysDiTect: Dyslexia Identification Using Cnn-Positional-Lstm-Attention Modeling with Chinese Dictation Task, Brain Sci. 14 (2024), 444. https://doi.org/10.3390/brainsci14050444.

[10] A. Ajit, K. Acharya, A. Samanta, A Review of Convolutional Neural Networks, in: 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), IEEE, 2020, pp. 1-5. https://doi.org/10.1109/ic-ETITE47903.2020.049.

[11] V. Pham, C. Pham, T. Dang, Road Damage Detection and Classification with Detectron2 and Faster R-Cnn, in: 2020 IEEE International Conference on Big Data (Big Data), IEEE, 2020, pp. 5592-5601. https://doi.org/10.1109/BigData50022.2020.9378027.

[12] P. Ce, B. Tie, An Analysis Method for Interpretability of CNN Text Classification Model, Futur. Internet 12 (2020), 228. https://doi.org/10.3390/fi12120228.

[13] A.K. Wijaya, H. Lucky, S. Arifin, Web Application for IHSG Prediction Using Machine Learning Algorithms, Indones. J. Appl. Math. Stat. 2 (2025), 29-40. https://doi.org/10.71385/idjams.v2i1.21.

[14] Q. Zhang, Y.N. Wu, S. Zhu, Interpretable Convolutional Neural Networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2018. https://doi.org/10.1109/CVPR.2018.00920.

[15] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, et al., A Survey of Methods for Explaining Black Box Models, ACM Comput. Surv. 51 (2018), 1-42. https://doi.org/10.1145/3236009.

[16] T. Szandała, Unlocking the Black Box of CNNs: Visualising the Decision-Making Process with PRISM, Inf. Sci. 642 (2023), 119162. https://doi.org/10.1016/j.ins.2023.119162.

[17] J. Wang, H. Qiu, Y. Rong, H. Ye, Q. Li, et al., BET: Black-Box Efficient Testing for Convolutional Neural Networks, in: Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis, ACM, New York, NY, USA, 2022, pp. 164-175. https://doi.org/10.1145/3533767.3534386.

[18] A. Rai, Explainable AI: From Black Box to Glass Box, J. Acad. Mark. Sci. 48 (2019), 137-141. https://doi.org/10.1007/s11747-019-00710-5.

[19] M. Robaa, M. Balat, R. Awaad, E. Omar, S.A. Aly, Explainable AI in Handwriting Detection for Dyslexia Using Transfer Learning, in: 2024 12th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC), IEEE, 2024, pp. 17-22. https://doi.org/10.1109/JAC-ECC64419.2024.11061192.

[20] A.R. Hakim, B. Warsito, M.R. Divangga, et al. Performance Convolutional Neural Network (CNN) and Support Vector Machine (SVM) on Tuberculosis Disease Classification Based On X-Ray Image, Commun. Math. Biol. Neurosci. 2025 (2025), 10. https://doi.org/10.28919/cmbn/9021.

[21] C. Fan, D. Yan, F. Xiao, A. Li, J. An, et al., Advanced Data Analytics for Enhancing Building Performances: From Data-Driven to Big Data-Driven Approaches, Build. Simul. 14 (2020), 3-24. https://doi.org/10.1007/s12273-020-0723-1.

[22] S. Panigrahi, D.R. Das Adhikary, B.K. Pattanayak, Integrating Interpolation Techniques with Deep Learning for Accurate Brain Tumor Classification, J. Comput. Math. Data Sci. 16 (2025), 100124. https://doi.org/10.1016/j.jcmds.2025.100124.

[23] A. Patil, M. Rane, Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition, in: T. Senjyu, P.N. Mahalle, T. Perumal, A. Joshi (Eds.), Information and Communication Technology for Intelligent Systems, Springer Singapore, Singapore, 2021: pp. 21–30. https://doi.org/10.1007/978-981-15-7078-0_3.

[24] X. Glorot, Y. Bengio, Understanding the Difficulty of Training Deep Feedforward Neural Networks, J. Mach. Learn. Res. 9 (2010), 249-256.

[25] P. Krishnaprasad, A. Ramanujan, Ramanujan Sums Based Image Kernels for Computer Vision, in: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), IEEE, 2016, pp. 835-839. https://doi.org/10.1109/ICEEOT.2016.7754803.

[26] T.O. Omotehinwa, M.O. Lawrence, D.O. Oyewola, E.G. Dada, Bayesian Optimization of One-Dimensional Convolutional Neural Networks (1D CNN) for Early Diagnosis of Autistic Spectrum Disorder, J. Comput. Math. Data Sci. 13 (2024), 100105. https://doi.org/10.1016/j.jcmds.2024.100105.

[27] A. Zafar, M. Aamir, N. Mohd Nawi, A. Arshad, S. Riaz, et al., A Comparison of Pooling Methods for Convolutional Neural Networks, Appl. Sci. 12 (2022), 8643. https://doi.org/10.3390/app12178643.

[28] A.B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, et al., Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges Toward Responsible AI, Inf. Fusion 58 (2020), 82-115. https://doi.org/10.1016/j.inffus.2019.12.012.

[29] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, et al., Grad-Cam: Visual Explanations from Deep Networks via Gradient-Based Localization, in: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE, 2017, pp. 618-626. https://doi.org/10.1109/ICCV.2017.74.

[30] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-Based Learning Applied to Document Recognition, Proc. IEEE 86 (1998), 2278-2324. https://doi.org/10.1109/5.726791.

[31] V.W. Berninger, T.L. Richards, Brain Literacy for Educators and Psychologists, Academic Press, 2002. https://doi.org/10.1016/b978-0-12-092871-2.x5000-1.