JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 35 (2025) 166–174
Received: 21 January 2025
Accepted: 07 April 2025

Advancing Bangla typography: Machine learning and transfer learning based font detection and classification approach using the 'BanglaFont45' dataset

Udoskonalanie typografii bengalskiej: podejście do wykrywania i klasyfikacji czcionek w oparciu o uczenie maszynowe i uczenie transferowe przy użyciu zbioru danych "Bang-laFont45"

Kazi Samiul Islam[a], Gourab Roy[a], Nafiz Nahid[a], Sunjida Yeasmin Ripti[a], Md. Abu Naser Mojumder[a*], Md. Janibul Alam Soeb[b], Md. Fahad Jubayer[c*]

[a]*Department of Computer Science and Engineering, Sylhet Engineering College, Sylhet, Bangladesh*
[b]*Department of Farm Power and Machinery, Sylhet Agricultural University, Sylhet, Bangladesh*
[c]*Department of Food Engineering and Technology, Sylhet Agricultural University, Sylhet, Bangladesh*

Abstract

This paper presents a dataset for detecting and classifying Bangla fonts, consisting of 28,000 images across 45 classes, aimed at supporting font users and typography researchers. Four traditional machine learning models— Support Vector Classifier (SVC), Logistic Regression (LR), K-Nearest Neighbors (KNN), and Random Forest—achieved accuracies of 93.43%, 92.37%, 84.71%, and 81.48%, respectively, with SVC performing best. Six transfer learning models—VGG-16, VGG-19, ResNet-50, MobileNet-v3, Xception, and Inception—were trained, yielding accuracies of 87.74%, 80.00%, 87.26%, 80.55%, 82.30%, and 80.11%, respectively. The results highlight the effectiveness of both traditional and transfer learning models in font detection, with SVC and VGG-16 emerging as top performers.

Keywords: Typography; image recognition; Xception; Support Vector Classifier (SVC).

Streszczenie

W artykule przedstawiono zbiór danych do wykrywania i klasyfikacji czcionek Bangla, składający się z 28 000 obrazów w 45 klasach, mający na celu wsparcie użytkowników czcionek i badaczy typografii. Cztery tradycyjne modele uczenia maszynowego — klasyfikator wektorów nośnych (SVC), regresja logistyczna (LR), k-najbliższych sąsiadów i losowy las — osiągnęły dokładność odpowiednio 93,43%, 92,37%, 84,71% i 81,48% przy Najlepiej radzi sobie SVC. Przeszkolono sześć modeli uczenia się z transferem — VGG-16, VGG-19, ResNet-50, MobileNet-v3, Xception i Inception — uzyskując dokładność na poziomie 87,74%, 80,00%, 87,26%, 80,55%, 82,30% i Odpowiednio 80,11%. Wyniki podkreślają skuteczność zarówno tradycyjnych, jak i transferowych modeli uczenia się w wykrywaniu czcionek, przy czym SVC i VGG-16 okazały się najskuteczniejsze.

Słowa kluczowe: Typografia; rozpoznawanie obrazu; Xcepcja; Klasyfikator wektorów nośnych (SVC).

*Corresponding author(s)

Email address: abu.naser84@gmail.com (M.A.N. Mojumder), fahadbau21@hotmail.com (M.F. Jubayer)

1. Introduction

Typography is the art of arranging type to make written language legible, readable, and visually appealing when displayed. Typography plays a critical role in both the visual and functional aspects of written communication [1]. It not only shapes the aesthetic presentation of text but also influences readability, user experience, and accessibility. One of the primary functions of typography is to enhance communication by bridging verbal and visual messages. Effective typography can convey meaning and emotion, making the text more engaging and memorable for the audience [2]. For instance, the layout and design of typography can significantly impact the viewer's ability to process information, as a well-organized typographic structure facilitates comprehension and retention of the message [3]. Moreover, typography plays a significant role in user experience, particularly in digital interfaces. Research indicates that typographic properties, such as font size and line length, are crucial for legibility and overall user satisfaction [4]. In digital environments, where attention spans are limited, effective typography can guide users through content, enhancing their interaction with the interface and improving information retention [4]. The visual hierarchy established through typography helps users navigate digital content more efficiently, ultimately leading to a more satisfying user experience [5].

In a digital era where multilingual content is increasingly prevalent, ensuring that typographic designs are optimized for various scripts is of paramount importance. Variable Typography employs artificial intelligence (AI) to adapt typographic styles, improving information acquisition and reducing cognitive load during reading. This approach is particularly beneficial in digital environments where users engage with content in multiple languages [6]. Bangla, one of the most widely spoken languages globally, presents unique challenges and opportunities in the realm of typography. With over 245 million speakers, primarily in Bangladesh and the Indian state of West Bengal, Bangla is a significant language for digital communication, yet it remains underrepresented in typographic research and development compared to Latin-based scripts [7]. Despite being one of the world's most widely spoken languages, Bangla faces unique challenges in text recognition due to its complex script structure, including graphemes and diacritics [8]. Efforts to address these challenges include developing character-level recognition models for both printed and handwritten documents [8], as well as language and printing style detection systems [9]. Font recognition has emerged as a crucial area of study, with researchers employing transfer learning methods [10] and deep learning approaches [11] to achieve high accuracy in Bangla font detection.

The importance of typography extends beyond mere aesthetics; it has implications for legibility, cultural representation, and the effectiveness of communication [12]. In the context of Bangla, the script's intricate design, rich historical context, and diverse usage across various mediums necessitate specialized approaches in font detection and classification. This becomes especially pertinent as digital content in Bangla continues to grow, spurred by increased internet penetration and digital literacy in the Bangla-speaking regions. However, the development of tools and techniques for Bangla typography lags behind, posing challenges for designers, developers, and end-users alike [13]. The use of machine learning in typography is a burgeoning field, offering new possibilities for font detection, classification, and even creation. Traditional methods of font identification, which often rely on manual inspection or simple rule-based systems, are increasingly being supplemented or replaced by machine learning approaches [14]. Machine learning has revolutionized typography, offering new possibilities for font recognition, classification, and generation. Deep learning approaches have shown promise in font recognition and retrieval, outperforming traditional manual methods [15]. Neural networks can learn typographic styles from a small subset of letters, enabling both discrimination and generation tasks [16]. Recent advancements include using auto encoders to generate skeleton-based typography, allowing for the creation of diverse and experimental fonts [17]. These techniques offer improved control over style and the ability to interpolate between characters.

The motivation for this study arises from a critical gap in the existing technological landscape. On a daily basis, we encounter various types of banners, posters, and digital media that employ a range of fonts, particularly on roadsides and social platforms. Frequently, a distinctive font piques our interest, prompting us to explore its name or origin. Typically, this leads to taking a photograph, cropping the text, and utilizing font identification platforms like 'WhatFontIs.' However, these platforms are predominantly tailored to English and a few other languages, with none offering support for Bangla fonts—a glaring omission given the widespread use of Bangla in South Asia. To address this gap and driven by the curiosity to fill this void, we initiated an exploration aimed at developing a tool specifically designed for Bangla font detection. Our initial approach centered on word-based dataset creation, but it produced suboptimal results. In Bangladesh, a large majority of people are unaware of the diverse array of Bangla fonts available, primarily due to the lack of accessible tools for font recognition. This project, therefore, seeks to fill this gap by curating a comprehensive dataset and developing a tool that not only identifies Bangla fonts but also encourages the use and appreciation of a broader spectrum of Bangla typography. In this study, we explore several machine learning approaches for font detection and classification using the developed 'BanglaFont45' dataset. Following the dataset description, we present the machine learning models developed for font detection and classification. These models include traditional approaches, such as support vector machines (SVMs) and k-nearest neighbors (k-NN), as well as more advanced techniques like convolutional neural networks (CNNs) and transfer learning. Each model is evaluated on its ability to accurately detect and classify fonts from the 'BanglaFont45' dataset, with performance metrics including accuracy, precision, recall, and F1-score.

Our work goes beyond merely improving font recognition technology. It aspires to foster greater adoption of Bangla fonts by simplifying the process of font identification and making typographic resources more accessible to users. This effort contributes to the preservation and promotion of the richness inherent in Bangla script through both technological innovation and cultural awareness. In light of this unmet need, we propose a novel methodology utilizing computer vision techniques to automate the identification of Bangla fonts—encompassing typeface, weight, and slope—directly from images and graphical representations. We term this approach Bangla Font Detection and Classification (BFDC). Unlike the well-explored field of optical character recognition (OCR), BFDC specifically focuses on the visual attributes of Bangla script rather than character recognition alone, setting it apart from conventional document analysis paradigms. However, real-world Bangla text is often presented in varied fonts, styles, and contextual settings, necessitating an approach that accounts for these complexities. This research aims to bridge that gap by introducing innovative algorithms specifically designed to address the unique characteristics of Bangla script. Our objective is to empower both designers and enthusiasts to easily recognize and engage with the diverse landscape of Bangla typography.

2. Materials and methods

2.1. Dataset

The dataset was constructed in a three-step process aimed at capturing a wide variety of commonly used fonts in Bangla ANSI and Unicode. The initial step involved selecting fonts frequently used in banners, posters, roadside advertisements, and social media. From an initial pool of 50 font classes, 45 were chosen based on overall outputs to ensure diversity and relevance. Professional help was enlisted to create an Adobe Illustrator file, which served as the basis for generating the dataset. This file contained art boards for each character, including 11 vowels, 39 consonants, 10 numerals and basic symbols. Each character was exported as a separate PNG image. The font size of each character was adjusted based on its position to enhance identification. The styles considered were regular, bold, and italic. All styles of a single font were organized into the relative font folder.

2.2. Image generation and organization

Each folder held, on average, approximately 560 images, although the quantity fluctuated based on the images' accuracy. A total of 18,000 images were initially produced from the Illustrator file. Image augmentation techniques were employed to expand the dataset to 28,000 images, as illustrated in the class count bar chart in Figure 1. This augmentation was essential to increase unpredictability and enhance the efficacy of the font recognition models.

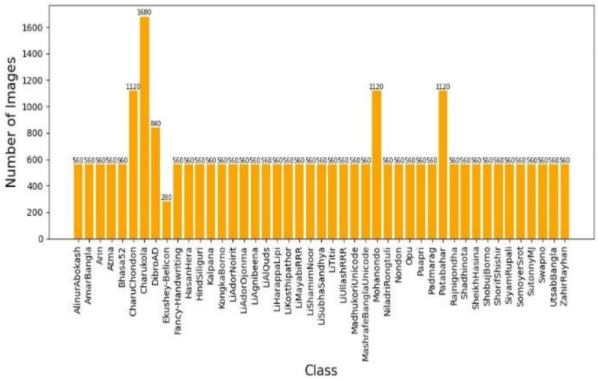

Figure 1: Class distribution of images in the BanglaFont45 dataset.

2.3. Image augmentation

Image augmentation techniques were carefully employed, primarily targeting underperforming font images and those deficient in specific styles. The employed augmentation strategies included rotation, width shifting, and the increase of width. Every folder was carefully labelled in accordance with its corresponding font. The dataset consists of 28,000 photos, obtained from an original collection of 18,000 by careful augmentation. This extensive dataset includes a diverse array of Bangla characters in multiple fonts and styles, rendering it an invaluable resource for the development of effective font recognition models. The meticulous stages and considerations involved in the dataset generation process emphasise the dedication to quality and diversity, which are crucial for attaining optimal performance in font recognition tasks. Figure 2 illustrates the visual data.

Figure 2: Visual representation of BanglaFont45 dataset fonts.

2.4. Implementation procedures

Figure 3 outlines the procedure for generating a dataset of Bangla font images, augmenting the dataset, and using it to train multiple machine-learning models for font classification. The process begins with the manual generation of 18,000 images. These images include Bangla characters (vowels, consonants, numerals, and symbols) in different fonts and styles (regular, bold, italic). To increase the dataset's size and variability, image augmentation techniques are applied. After augmentation, the dataset grows to 28,000 images, providing a more comprehensive set of examples for training ML models.

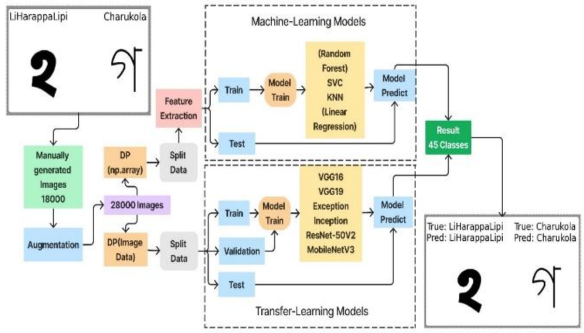

Figure 3: Workflow for BanglaFont45 dataset generation, augmentation, and machine learning model training.

2.5. Data preparation

The initial stage in the data preparation process for the ML model entails turning images into NumPy arrays. The method commences by loading each image from the dataset and scaling it to a designated target dimension (e.g., 224x224 pixels) with three color channels (RGB). The images are then transformed into arrays and normalized by dividing by 255.0 to scale the pixel values to a range between 0 and 1. During data preprocessing, we utilized Keras's `ImageDataGenerator` to supplement and prepare the training, validation, and testing datasets. We normalised the training data by rescaling pixel values to the [0, 1] range and employed augmentation techniques including random rotations, width and height shifts, shear transformations, zoom operations, and horizontal flips, filling vacant areas with nearest neighbor pixels. The validation and test datasets were solely rescaled, without any supplementary augmentation, to preserve data integrity throughout evaluation. Utilizing the `flow_from_directory` function, we import photos from designated folders, resize them to the appropriate dimensions, and batch them for processing, configuring the `class_mode` to 'categorical' for multi-class categorization. This guaranteed an uninterrupted supply of preprocessed and enriched data during training and evaluation.

2.6. Data splitting

The dataset, consisting of 28,000 images, is divided into training and testing subsets. 90% of the data (25,200 images) is designated for training the ML models, while the remaining 10% (2,800 images) is allotted for testing. The dataset is divided into three subsets: 70% (19,600 images) for training, 10% (4,200 images) for validation, and 10% (4,200 images) for testing. The training subset is employed to train deep learning (DL) models, the validation subset is utilized to optimize the models and mitigate over fitting, and the testing subset is used to assess the final performance of the trained models.

2.7. Model training

2.7.1. Machine learning classifiers

We explored different machine learning models for classification tasks, with four demonstrating superior performance: Random Forest, Support Vector Machines (SVMs), Logistic Regression, and K-Nearest Neighbours (KNN). Each model offers a distinct methodology for categorisation, enabling us to evaluate which one most effectively addresses the particular requirements of our dataset. The Random Forest model, a prevalent ensemble learning technique, functions by generating several decision trees during the training process. Each tree is constructed by random picks of characteristics and data subsets, and the individual forecasts from these trees are aggregated to provide a final prediction [18]. SVMs, an alternative model we evaluated, classify by identifying the best hyperplane that delineates distinct classes within the training data. SVMs seek to optimize the margin between the hyperplane and the nearest data points from each class, thereby establishing a resilient classification border [19]. In addition, we utilised Logistic Regression and K-Nearest Neighbours (KNN), each possessing distinct advantages. Logistic Regression, or the logit model, is extensively utilised in predictive analytics and classification endeavours [20]. K-Nearest Neighbours (KNN) provides a straightforward and efficient non-parametric method for classification. KNN predicts outcomes by examining the 'k' nearest data points in the training set, classifying new data according to its proximity to these neighbors, rather than constructing a model during training [21]. We adopted an image size of 224 × 224-pixels and before fitting the model, the MobileNet architecture was employed for feature extraction on the numpy train and test data, resulting in 4D arrays with dimensions of (25200, 5, 5, 1280) for the training set and (2800, 5, 5, 1280) for the test set. Here, 25200 and 2800 represent the number of samples. The feature maps contain 1280 channels for the training set and 1280 channels for the test set. Subsequently, these 4D arrays were reshaped into 2D arrays without altering the data, maintaining the number of samples as 25200 for the training set and 2800 for the test set. The reshaping process resulted in dimensions of (25200, 32000) for the training set and (2800, 32000) for the test set.

2.7.2. Transfer learning classifiers

The standout model, VGG-16, has a straightforward yet powerful design with 16 layers predominantly composed of convolutional layers, which aim to extract hierarchical patterns and textures from input images. Each block in VGG-16 includes 3x3 convolutional filters stacked on top of each other, using ReLU activation functions to introduce non-linearity, and max-pooling layers to progressively reduce spatial dimensions. The final layers are fully connected and serve to classify the extracted features, assigning each image to a specific category. With a total of approximately 15.3 million parameters, VGG-16's architecture is large but well-optimized for image recognition tasks. Most of its parameters are non-trainable, representing pre-trained weights from large datasets like ImageNet, which is crucial in transfer learning, as these weights enable the model to generalize effectively on new tasks with limited training data. In addition to VGG-16, other architectures were assessed, each possessing unique characteristics tailored to specific facets of image processing. VGG-19, an enhanced variant of VGG-16 including 19 layers, adheres to a comparable architecture, augmenting depth to apprehend more complex patterns. Xception employs depthwise separable convolutions in lieu of traditional convolutional layers, yielding a more economical model that diminishes computational expense while maintaining performance [22]. ResNet50 incorporates residual connections, or "skip connections," across layers, facilitating the training of significantly deeper networks by maintaining gradient flow during back propagation. The residual connections alleviate the vanishing gradient issue, rendering ResNet50 exceptionally efficient for deep networks that may face training challenges [23]. Inception v3 utilizes distinctive modules that capture multi-scale features through various filter sizes within a single module [24], whereas MobileNetV3, recognized for its lightweight design, incorporates inverted residual blocks with depthwise separable convolutions and squeeze-and-excitation modules, rendering it efficient and suitable for resource-limited settings [25]. A 224 × 224 pixel picture was used to train the network. When it comes to picture classification jobs, our optimizer of choice, "Adam," is a popular pick. Given the nature of our multi-class classification task, we set up the output layer using a "Softmax" activation function and utilized "Categorical Cross Entropy" as the loss function, following accepted industry standards. Table 1 presents the hyper parameters used for training these five models, including the number of epochs, learning rate, and batch size. These parameters were carefully selected to optimize the performance of each model during training.

Table 1: Training model parameters

Model	Epoch	Batch size	Learning rate
VGG-16	80	64	0.001
VGG-19	80	64	0.001
ResNet-50	30	64	0.001
MobileNet-v3	80	64	0.0001
Xecption	35	64	0.001
Inception	30	64	0.01

2.8. Evaluation metrics

The experimental results comprised four outcomes: true positive (TP), false positive (FP), true negative (TN), and false negative (FN) [26]. For evaluating the performance of a CNN in a classification task, several evaluation metrics are commonly used to gauge its accuracy and effectiveness, namely, precision, recall, accuracy, F1-score, confusion matrix, and categorical cross entropy loss. These evaluation metrics collectively provide a comprehensive assessment of CNN's classification performance, considering factors such as accuracy, precision, recall, and the ability to handle imbalanced datasets. Evaluating CNN models using these metrics helps in understanding their effectiveness in solving classification tasks across various domains. Prominent statistical Metrics are calculated using the following equations [27,28].

$$Precision = \frac{TP}{(TP+FP)} \quad (1)$$

$$Recall = \frac{TP}{(TP+TN)} \quad (2)$$

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (3)$$

$$F1 - score = \frac{2}{(\frac{1}{precision})+(\frac{1}{recall})} \quad (4)$$

Here, TP = true positive, FP = false positive, TN = true negative, and FN = false negative.

3. Experimental results and analysis

3.1. Experimental environment

The model was created in the research lab server of Sylhet Engineering College, Bangladesh, and is compatible with a Windows server 2019 system. Python 3.10.14, Tensorflow 2.10 and PyTorch 1.11 were employed for the development. The experiment was executed using a single GPU NVIDIA RTX A4000 with 16 GB of memory and CPU RAM 64 GB.

3.2. Machine learning classifier result

The performance of ML classifiers on the BanglaFont45 dataset demonstrates distinct differences in their ability to effectively classify fonts, as summarized in Table 2. Among the four models—SVC, LR, KNN, and Random Forest—SVC emerges as the most effective classifier, achieving an accuracy of 93.43%, along with high precision, recall, and F1-score values of 93%, 92%, and 93%, respectively. Sentiment analysis of Bengali news articles using supervised classifiers demonstrated high accuracy, with Random Forest achieving 99% [29]. The superior performance of SVC can be attributed to its ability to define an optimal hyperplane that maximizes the margin between classes, which is particularly beneficial for datasets with high-dimensional feature spaces.

Table 2: Performance metrics of machine learning classifiers on the BanglaFont45 dataset

Model	Accuracy	Precision	Recall	F1-score
SVC	93.43%	93%	92%	93%
LR	92.37%	92%	91%	92%
KNN	84.71%	85%	83%	84%
Random Forest	81.48%	82%	81%	81%

Logistic Regression closely follows SVC, with an accuracy of 92.37%, precision of 92%, recall of 91%, and F1-score of 92%. Despite its relative simplicity, Logistic Regression performs well due to its efficiency in handling linearly separable data, which aligns with the feature representations of the BanglaFont45 dataset. KNN and Random Forest, while still achieving acceptable results, lag behind SVC and LR. KNN records an accuracy of 84.71%, with a slight drop in precision (85%), recall (83%), and F1-score (84%). Its reliance on the local proximity of data points, coupled with the inherent diversity in Bangla fonts, may have contributed to this marginally lower performance. Random Forest, on the other hand, yields an accuracy of 81.48%, with consistent precision, recall, and F1-score values at 82%, 81%, and 81%, respectively. The ensemble nature of Random Forest enables it to handle complex decision boundaries; however, its performance is slightly hindered by the variability in font styles and augmentations present in the dataset. RF tends to be sensitive to small changes in training data and may overfit.

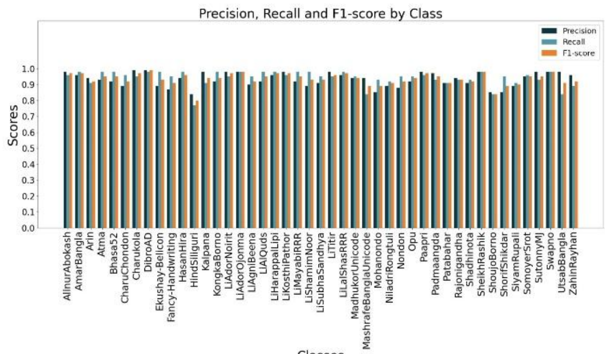

Figure 4: Precision, recall, and F1-score distribution across Bangla fonts.

Figure 4 provides a deeper insight into the model performance by visualizing the precision, recall, and F1-scores for individual font classes. The consistency of these scores across most classes underscores the robustness of the classifiers. However, certain font classes exhibit slightly lower scores, indicating potential areas for model improvement. These discrepancies could stem from variations in font characteristics, such as subtle stylistic differences or overlapping visual features between certain classes. The recognition of handwritten Bangla characters and numerals has been extensively studied using various machine learning algorithms. For Bangla numeral recognition, Support Vector Machines (SVM), Library for Large Linear Classification (LIBLINEAR), and Modified Quadratic Discriminant Function (MQDF) have demonstrated high accuracy [30]. In a recent study on Assamese word recognition, Gradient Boosting achieved the highest accuracy of 96.03%, closely followed by Logistic Regression and SVM with RBF kernel at 95.64% and 95.60%, respectively [31].

3.3. Transfer learning classifier result

The training and validation loss curves (Figure 5) and accuracy curves (Figure 6) highlight the learning behavior and generalization capabilities of each model. Among the models, VGG-16 demonstrates the best performance with the highest validation accuracy (0.8784) and low validation loss (0.3591), followed closely by ResNet-50 (validation accuracy: 0.8757, validation loss: 0.3765). Xception achieves moderate results (validation accuracy: 0.8395), while VGG-19 and Inception show reduced generalization with validation accuracies of 0.8022 and 0.8130, respectively. MobileNet-V3, although lightweight and efficient, lags behind with lower validation accuracy (0.8058) and the highest training loss (0.8792). These findings establish VGG-16 as the most effective model for Bangla font classification, with ResNet-50 as a strong alternative and MobileNet-V3 suitable for resource-constrained applications.

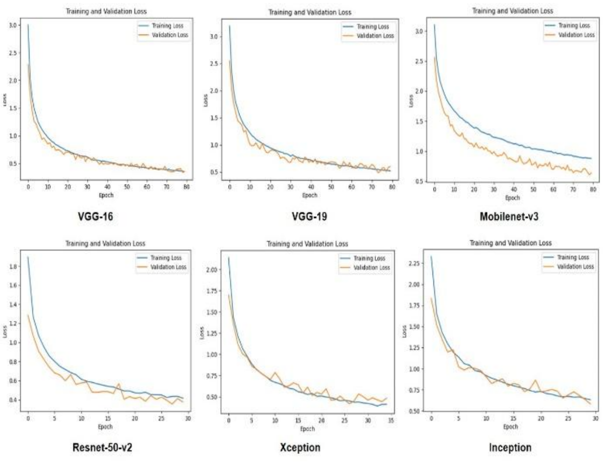

Figure 5: Training and validation loss for transfer learning models on the BanglaFont45 dataset.

VGG-16 emerges as the top-performing model, achieving the highest validation accuracy with steady convergence and minimal overfitting, highlighting its suitability for extracting relevant features. VGG-19 follows closely with its deeper architecture, while MobileNet-v3 achieves competitive accuracy with lower computational overhead, making it ideal for resource-constrained environments. Recent studies have explored deep learning approaches for Bangla font and character recognition. VGG-16 consistently emerges as a top-performing model, achieving high accuracy in both font detection (96.23%) and character recognition (98.65%) tasks [10]. VGG-19 and other architectures like Xception, Inception V3, and Vision Transformer also show promising results [10]. ResNet-50-v2 and Xception effectively handle the complex features of Bangla fonts due to their advanced architectures, such as residual connections and depthwise separable convolutions, although their higher computational requirements may limit practical applicability in some scenarios. Inception, while slightly less accurate, demonstrates robust multi-scale feature extraction. The consistent reduction in loss and the minimal gap between training and validation metrics across all models indicate strong generalization, attributed to the high quality and augmentation strategies of the BanglaFont45 dataset. These results affirm the potential of transfer learning in Bangla font classification, with VGG-16 as the most accurate model and MobileNet-v3 providing a computationally efficient alternative. Researchers have developed large datasets of Bangla fonts and characters to train these models, with some studies using data augmentation to expand their training sets [11].

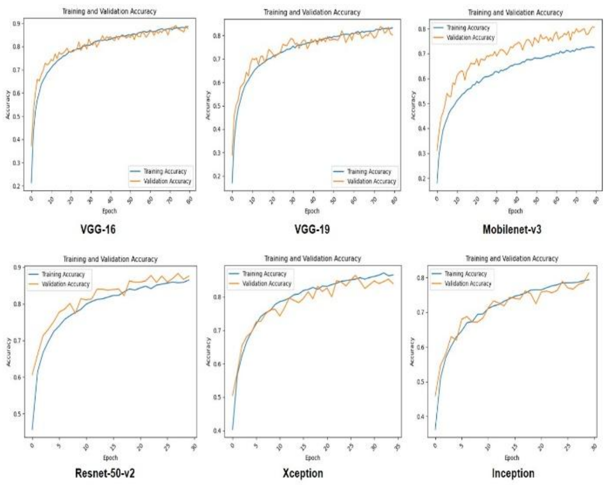

Figure 6: Training and validation accuracy for transfer learning models on the BanglaFont45 dataset.

3.4. Test results

Table 3 summarizes the performance of six transfer learning models in detecting and classifying Bangla fonts. These results offer critical insights into the models' ability to generalize to new instances, which is vital for evaluating their real-world applicability and robustness. VGG-16 leads with the highest accuracy (87.74%), precision (88%), recall (87%), and F1-score (86%), showcasing its robust feature extraction and generalization.

Similarly, ResNet-50 achieves comparable metrics, with an F1-score of 87%, confirming its efficiency in handling the dataset's complexity through residual connections. Xception also delivers a strong performance with an F1-score of 82%, benefiting from its use of depthwise separable convolutions. A CNN-based approach using global average pooling achieved 96% line-level accuracy on a dataset of 10 Bangla fonts [32]. Another study proposed a CNN approach with a space adjustment method using a stacked convolutional auto-encoder, achieving 98.73% accuracy on a dataset of 7 Bangla fonts [11]. While these models show comparable performance, the challenges unique to Bangla fonts such as the intricacies of ligatures and diacritics require further refinement in transfer learning-based approaches. When comparing our findings with research focused on other languages' font detection, several significant observations emerge. For instance, Ali et al. (2023) deployed a Sindhi-ligature font-diverse large-scale recognition system, achieving an accuracy of 92% [33]. Arafat et al. (2022) deployed detection and Urdu-text recognition at 89% accuracy [34]. These results suggest that models will need to be tuned to accommodate the distinctive problems posed by each script. Scripts derived from Latin are most likely to involve frequent character shapes, while the complexity of Bangla fonts presents an isolated challenge, perhaps partially explaining the observed range in model performance. The close lines and intricate curves of the Bangla fonts add one level of complexity that is not as common in Chinese, Sindhi, or Urdu script font configurations.

Table 3: Test results of transfer learning models

Models	Accuracy	Precision	Recall	F1-score
Vgg-16	87.74%	88%	87%	86%
Vgg-19	80.00%	82%	79%	78%
ResNet-50	87.26%	88%	87%	87%
MobileNet-v3	80.55%	80%	78%	78%
Xception	82.30%	85%	82%	82%
Inception	80.11%	81%	80%	80%

Models like VGG-19 and MobileNet-V3, however, show slightly reduced accuracy (80.00% and 80.55%, respectively) and precision-recall metrics, reflecting their limitations in extracting intricate font features. Inception, with an F1-score of 80%, demonstrates moderate capability but lags behind the top performers in precision and recall. The variability across individual font classes, illustrated in Figure 7, shows most classes achieving high precision, recall, and F1-scores, though a few classes underperform, likely due to overlapping visual characteristics or insufficient representation in the dataset. Transfer learning methods have also been applied, with VGG-16 achieving 96.23% accuracy on a dataset of five fonts [12]. Various models have been evaluated for Bangla Optical Character Recognition (OCR), including Inception V3, VGG16, and Vision Transformer, with VGG-16 achieving the highest accuracy of 98.65% on a large dataset of Bengali characters [35].

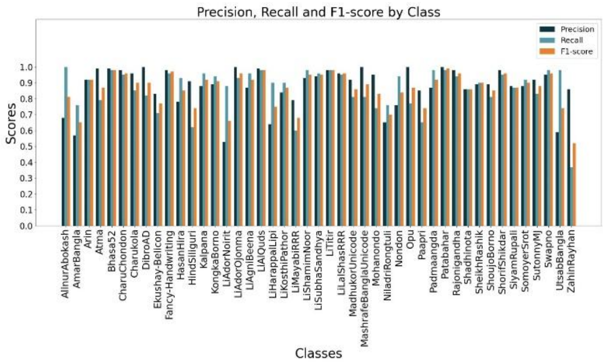

Figure 7: Precision, recall, and F1-Score across 45 BanglaFont classes using transfer learning models.

3.5. Model performance

The Support Vector Classifier (SVC) emerged as the top-performing model in both traditional machine learning and transfer learning approaches, achieving an accuracy of 93%. Figure 8 illustrates the SVC model's performance through its confusion matrix, which provides a detailed breakdown of classification outcomes across 45 font classes, including true positives, true negatives, false positives, and false negatives. The matrix highlights the model's high accuracy and effectiveness in handling the classification task, reinforcing its reliability for Bangla font recognition.

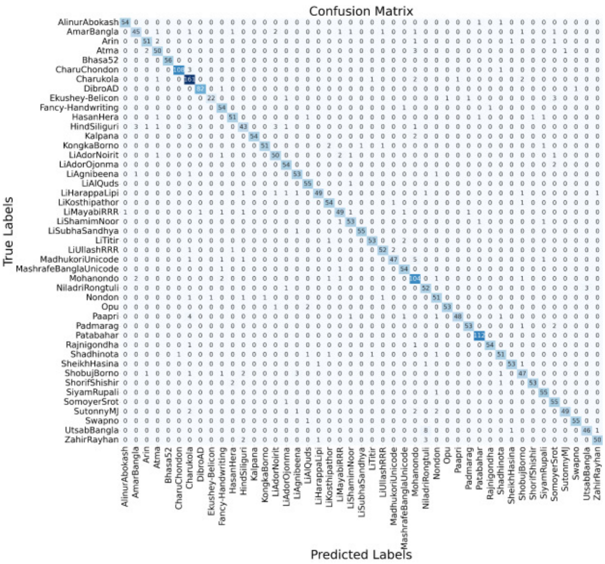

Figure 8: Confusion matrix for transfer learning models.

Figure 9 presents a visualization of the SVC model's predictions for randomly selected font classes. Each image is annotated with its true label and the corresponding predicted label, offering a clear depiction of the alignment between ground truth and the model's predictions. This visualization provides valuable insights into the model's performance across diverse classes, showcasing its ability to accurately classify Bangla fonts while also identifying areas for potential improvement. Such visual tools play a crucial role in assessing and refining machine learning models applied to complex datasets.

Figure 9: Sample true and predicted classifications of BanglaFonts.

4. Limitations

While this study presents significant advancements in Bangla font detection and classification, certain limitations should be acknowledged. Firstly, the dataset, although comprehensive, may not fully represent the vast diversity of Bangla fonts, especially those in niche or regional usage. The dataset's focus on specific fonts may limit the generalizability of the models to entirely new or custom fonts not included in the training set. Secondly, the study primarily addresses isolated font classification scenarios, whereas real-world applications often involve complex layouts, overlapping text, or degraded font samples, which could affect model accuracy.

5. Conclusions and future works

This study successfully demonstrates the feasibility of leveraging machine learning and transfer learning models for Bangla font detection and classification using the 'BanglaFont45' dataset. The results reveal that traditional machine learning models, particularly the Support Vector Classifier (SVC), and transfer learning models like VGG-16, deliver high accuracy and robustness in this domain. These findings contribute significantly to the field of Bangla typography and pave the way for the development of practical applications in font identification and digital design. Future work can address the identified limitations and extend the scope of this research. Expanding the dataset to include a broader variety of Bangla fonts, including custom and regional designs, would enhance the generalizability of the models. Incorporating real-world scenarios, such as mixed-font layouts and degraded text samples, could further improve the models' applicability. Additionally, exploring hybrid approaches that integrate contextual information, such as linguistic features or semantic cues, may provide a richer framework for font classification. Development of a specialized, lightweight Convolutional Neural Network (CNN) tailored to Bangla font classification could balance computational efficiency and accuracy, enabling deployment on mobile and embedded devices. Collaboration with typographers and software developers could lead to the creation of user-friendly tools for Bangla font recognition and recommendation. By addressing these areas, future research can continue to bridge the gap between technological advancements and practical needs in Bangla typography.

Funding

No funding received for this work.

Acknowledgement

Nothing to declare.

Data availability

Data will be made available upon reasonable request.

Declaration of competing interest

No potential conflict of interest was reported by the authors. All authors have read and agreed to the published version of the manuscript.

References

[1]. H. Ren, Y. Liu, G. Naren, J. Lu, The impact of multidirectional text typography on text readability in word clouds, Displays 83 (2024) 102724, https://doi.org/10.1016/j.displa.2024.102724.

[2]. Y.A. Hermanto, N.B. Ibrahim, J.A. Sudarmanto, F.W. Arizal, N.W. Firmansyah, Atomic design system for typography layout in student poster design, ANDHARUPA: Jurnal Desain Komunikasi Visual & Multimedia 9(3) (2023) 310-325.

[3]. R.A. Hurley, J. Fischer, A.D. Ouzts, J. Leininger, R. Thomas, A methodology for determining the proper point size for display type on packaging by means of the package's proportions, Packaging Technology and Science 27(12) (2014) 921-31, https://doi.org/10.1002/pts.2067.

[4]. M.E. Dick, M. Woloszyn, Influence of typographic properties on user experience in digital interfaces, Estudos em Design 31(2) (2023) 99-109, https://doi.org/10.35522/eed.v31i2.1711.

[5]. T. Zhu, Y. Yang, Research on mobile learning platform interface design based on college students' visual attention characteristics, Plos One 18(7) (2023) e0283778, https://doi.org/10.1371/journal.pone.0283778.

[6]. S. Shi, W. You, K. Han, J. Song, L. Sun, VariableTypography: Artificial Intelligence Augmented Reading Experience, In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers 87295 (2023) V002T02A088, https://doi.org/10.1115/DETC2023-114811.

[7]. M.S. Islam, Research on Bangla language processing in Bangladesh: progress and challenges, In 8th International Language & Development Conference (2009) 23–25.

[8]. A.K.M.S.A. Rabby, M.M. Islam, Z. Islam, N. Hasan, F. Rahman, Towards building a robust large-scale Bangla text recognition solution using a unique multiple-domain character-based document recognition approach, In 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE (2021) 1393–1399, https://doi.org/10.1109/ICMLA52953.2021.00225.

[9]. A.K.M.S.A. Rabby, M.M. Islam, N. Hasan, J. Nahar, F. Rahman, A novel deep learning character-level solution to

detect language and printing style from a bilingual scanned document, In 2020 IEEE International Conference on Big Data (Big Data), IEEE (2020) 5218–5226, https://doi.org/10.1109/BigData50022.2020.9378262.

[10]. S. Hasan, G. Rabbi, R. Islam, H.I. Bijoy, A. Hakim, Bangla font recognition using transfer learning method, In 2022 International Conference on Inventive Computation Technologies (ICICT), IEEE (2022) 57–62, https://doi.org/10.1109/ICICT54344.2022.9850765.

[11]. M.M. Islam, A.K.M.S.A. Rabby, N. Hasan, J. Nahar, F. Rahman, A novel Bangla font recognition approach using deep learning, Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Singapore: Springer Nature Singapore 2 (2021) 745–754, https://doi.org/10.1007/978-981-33-4367-2_71.

[12]. E. Grabska, I. Grabska-Gradzińska, T. Frodyma, The role of typography in visual design, Craft, Technology and Design (2022) 118-130.

[13]. A.K. Shamsuddin, M.B. Islam, Bengali typesetting for graphics design: Difficulties and way out, International Journal of Trends in Computer Science 2(11) (2013) 386–400.

[14]. A. Qaroush, A. Awad, M. Modallal, M. Ziq, Segmentation-based, omnifont printed Arabic character recognition without font identification, Journal of King Saud University-Computer and Information Sciences 34(6) (2022) 3025–3239, https://doi.org/10.1016/j.jksuci.2020.10.001.

[15]. Z. Wang, J. Yang, H. Jin, Z. Wang, E. Shechtman, A. Agarwala, J. Brandt, T.S. Huang, Deep learning for font recognition and retrieval, Applied Cloud Deep Semantic Recognition Auerbach Publications (2018) 109–130.

[16]. S. Baluja, Learning typographic style, arXiv preprint, arXiv:1603.04000 (2016), https://doi.org/10.48550/arXiv.1603.04000.

[17]. J. Parente, L. Gonçalo, T. Martins, J.M. Cunha, J. Bicker, P. Machado, Using autoencoders to generate skeleton-based typography, International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar), Cham: Springer Nature Switzerland (2023) 228–243, https://doi.org/10.1007/978-3-031-29956-8_15.

[18]. G. Biau, E. Scornet, A random forest guided tour, Test 25 (2016) 197–227, https://doi.org/10.1007/s11749-016-0481-7.

[19]. S. Salcedo-Sanz, J.L. Rojo-Álvarez, M. Martínez-Ramón, G. Camps-Valls, Support vector machines in engineering: an overview, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 4(3) (2014) 234–267, https://doi.org/10.1002/widm.1125.

[20]. J.C. Stoltzfus, Logistic regression: a brief primer, Academic Emergency Medicine 18(10) (2011) 1099–1104, https://doi.org/10.1111/j.1553-2712.2011.01185.x.

[21]. Z. Zhang, Introduction to machine learning: k-nearest neighbors, Annals of Translational Medicine 4(11) (2016) 218, https://doi.org/10.21037/atm.2016.03.37.

[22]. K. Shaheed, A. Mao, I. Qureshi, M. Kumar, S. Hussain, I. Ullah, X. Zhang, DS-CNN: A pre-trained Xception model based on depth-wise separable convolutional neural network for finger vein recognition, Expert Systems with Applications 191 (2022) 116288, https://doi.org/10.1016/j.eswa.2021.116288.

[23]. A. Shabbir, et al., Satellite and scene image classification based on transfer learning and fine-tuning of ResNet50, Mathematical Problems in Engineering 2021(1) (2021) 5843816. https://doi.org/10.1155/2021/5843816.

[24]. X. Xia, C. Xu, B. Nan, Inception-v3 for flower classification, In 2017 2nd International Conference on Image, Vision and Computing (ICIVC), IEEE (2017) 783–787, https://doi.org/10.1109/ICIVC.2017.7984661.

[25]. S. Wang, J. Ren, X. Guo, A high-accuracy lightweight network model for X-ray image diagnosis: A case study of COVID detection, PLOS One 19(6) (2024) e0303049, https://doi.org/10.1371/journal.pone.0303049.

[26]. M.F. Jubayer, F.M. Ruhad, M.S. Kayshar, Z. Rizve, M.J.A. Soeb, S. Izlal, M.I. Meftaul, Detection and identification of honey pollens by YOLOv7: A novel framework toward honey authenticity, ACS Agricultural Science & Technology 4(7) (2024) 747–758, https://doi.org/10.1021/acsagscitech.4c00220.

[27]. S. Rabby, F. Hossain, S. Das, I. Rahman, S. Das, J. Soeb, M.F. Jubayer, An automated approach for the kidney segmentation and detection of kidney stones on computed tomography using YOLO algorithms, Journal of Ideas in Health 6(4) (2023) 963–970, https://doi.org/10.47108/jidhealth.Vol6.Iss4.313.

[28]. M.J.A. Soeb, et al., Tea leaf disease detection and identification based on YOLOv7 (YOLO-T), Scientific Reports 13(1) (2023) 6078, https://doi.org/10.1038/s41598-023-33270-4.

[29]. S.J. Maisha, A.K. Masum, Supervised machine learning algorithms for sentiment analysis of Bangla newspaper, International Journal of Innovative Computing 11(2) (2021) 15–23, https://doi.org/10.11113/ijic.v11n2.321.

[30]. C. Halder, J. Paul, K. Roy, Comparison of the classifiers in Bangla handwritten numeral recognition, In 2012 International Conference on Radar, Communication and Computing (ICRCC), IEEE (2012) 272-276, https://doi.org/10.1109/ICRCC.2012.6450594.

[31]. N. Borah, U. Baruah, M.T. Ramakrishna, V.V. Kumar, D.R. Dorai, J.R. Annad, Efficient Assamese word recognition for societal empowerment: A comparative feature-based analysis, IEEE Access 11 (2023) 82302–82326, https://doi.org/10.1109/ACCESS.2023.3301564.

[32]. M.Z. Hasan, K.T. Rahman, R.I. Riya, K.Z. Hasan, N. Zahan, A CNN-based classification model for recognizing visual Bengali font, Proceedings of International Joint Conference on Computational Intelligence: IJCCI 2018, Springer Singapore (2020) 471–482. https://doi.org/10.1007/978-981-13-7564-4_40.

[33]. Z. Ali, S.A. Khan, M.K. Shahzad, H.S.M. Bilal, A Large-Scale Font-Diverse Sindhi Ligature Recognition System, In 2023 International Conference on Frontiers of Information Technology (FIT) IEEE (2023) 132-137, https://doi.org/10.1109/FIT60620.2023.00033.

[34]. S.Y. Arafat, N. Ashraf, M.J. Iqbal, I. Ahmad, S. Khan, J.J. Rodrigues, Urdu signboard detection and recognition using deep learning, Multimedia Tools and Applications (2022) 11965-11987, https://doi.org/10.1007/s11042-020-10175-2.

[35]. N.M. Dipu, S.A. Shohan, K.M. Salam, Bangla optical character recognition (OCR) using deep learning-based image classification algorithms, In 2021 24th International Conference on Computer and Information Technology (ICCIT), IEEE (2021) 1-5, https://doi.org/10.1109/ICCIT54785.2021.9689864.