"Unveiling Strikeouts in Handwritten Kannada Text: Harnessing the Power of Pre-trained Deep Learning Models"

Mr. Sangamesh S K
Assistant Professor
Department of Computer Science and Engineering
Rural Engineering College, Hulkoti

Mr. Pruthviraj Ambliyavar
Student
Department of Computer Science and Engineering
Rural Engineering College, Hulkoti

Mrs. Pavitra Gadhar
Assistant professor
Department of Computer Science and Engineering
K.L.E. Institute of Technology,Hubballi

Abstract

In this paper, we present a deep learning approach for detecting strikeouts in handwritten Kannada text. Strikeouts are common editing marks used to indicate deletion or correction of text in handwritten documents. Automated detection of strikeouts can aid in digital transcription and analysis of handwritten documents, particularly in contexts where efficient processing of large volumes of text is required.We propose a methodology that leverages pre-trained deep learning models, transfer learning, and image classification techniques to identify strikeouts in handwritten Kannada text. The process involves collecting a dataset of handwritten Kannada text images, preprocessing the data, fine-tuning a pre-trained convolutional neural network (CNN) on the dataset, and training the model to classify text regions with strikeouts.Our experimental results demonstrate the effectiveness of the proposed approach in accurately detecting strikeouts in handwritten Kannada text. We evaluate the model's performance using standard metrics such as accuracy, precision, recall, and F1-score, achieving promising results on a diverse test dataset.The developed system has potential applications in various domains, including digitization of historical documents, automatic transcription of handwritten notes, and analysis of handwritten forms and surveys. Future work may involve extending the approach to handle other languages and types of handwritten text, as well as improving the model's robustness to variations in writing styles and document conditions

Keywords

Strikeout, non-strikeout, Convolutional Neural Networks (CNN), Kannada handwritten Document

Introduction

Handwritten documents, with their historical significance and cultural value, often present unique challenges for digital processing and analysis. In particular, handwritten text may contain various editing marks, such as strikeouts, which denote deletions or corrections made during the writing process. Detecting and interpreting these strikeouts automatically can greatly facilitate the transcription and understanding of handwritten documents, especially in scenarios involving large volumes of text.This paper focuses on the task of detecting strikeouts in handwritten Kannada text, a language spoken predominantly in the southern Indian state of Karnataka. Kannada script, with its distinctive characters and ligatures, poses specific challenges for automated text analysis. Strikeout detection in Kannada text is particularly relevant for digitizing historical documents, transcribing handwritten notes, and analyzing handwritten forms and surveys.To address this challenge, we propose a deep learning-based approach that leverages pre-trained models and transfer learning. By utilizing pre-trained deep learning architectures, we aim to harness the learned features from large-scale datasets and adapt them to the task of strikeout detection in handwritten Kannada text. This approach offers several advantages, including reduced training time, improved accuracy, and generalization to diverse writing styles and document conditions.In this introduction, we provide an overview of the problem statement and its significance, outline the objectives of our study, and summarize the methodology employed. Subsequent sections will delve into the details of our approach, including data collection and preprocessing, model selection and fine-tuning, experimental setup, and results analysis. Finally, we discuss potential applications, limitations, and avenues for future research in strikeout detection and handwritten text analysis.

Related works

[2] Some works related to the present work are analyzed. It investigates and improves the performance of handwritten text recognition method Convolutional Recurrent Neural Network (CRNN) on handwritten lines containing struck out words. A model, trained on the IAM line database was tested on lines containing struck-out words.The model shows superior performance with respect to struck-out text detection [2].

In another study, the identification and processing of struck-out texts in unconstrained handwritten document images is introduced. The model uses a combined approach (i) pattern classification and (ii) graph-based method for identifying texts. In case of (i), Support Vector Machine (SVM) classifier is used to classification and detect moderate-sized struck-out components. In case of (ii), skeleton of the text component is considered as a graph. The strike-out stroke is identified using a constrained shortest path algorithm. Experimental analysis is carried out using 500 pages of documents. The model obtained an overall 91.56% F-Measure in English (Bengali) script for struck-out text detection [3].

[4] Deep learning techniques are used for Kannada handwritten character recognition. Here, convolutional neural networks (CNN) is adopted for the classification. It is a robust, dynamic and swift method to recognition handwritten character. Performance of the model is achieved approximately 93.2 % and 78.73 % for the two different datasets. Similarly, [5] both CNN and Tesseract tool is used for Kannada handwritten character recognition. Here, CNN model achieves 87% and Tesseract tool shows 86% classification accuracy.

Kannada language has many confusing characters which cause high difficulties in extraction. The dataset which is used for experimental analysis has ten classes. A novel method is introduced to identify using Random forest and Support Vector Machine classifiers. It achieves 78% classification accuracy[6]. Non-overlapping lines of Kannada characters are recognized using CNN deep learning model. It uses Chars74K dataset to build the model. The model achieves 98% classification accuracy [7]. Kannada numerals and handwritten characters recognition is performed with help of Artificial Neural Network. It also use wavelet transform for global feature extraction. The proposed method is experimented on 4800 images of handwritten Kannada characters and 1000 images of handwritten Kannada numerals. The model shows classification accuracy of 91.00%. and 97.60% for Kannada handwritten characters and Kannada numerals respectively [8].

Single Kannada andEnglish character recognition is performed based on zone features. The experiment is performed using 2800 Kannada consonants and 2300 lowercase alphabets. It uses 32 X 32 normalized images and classification is performed with the help of SVM classifier. Here, preprocessed image is divided into nonoverlapping 64 zones to generate respective features. This model demonstrates average recognition accuracy of 73.33% and 96.13% for Kannada consonants and English lowercase alphabets respectively [11].

In another approach, Devanagari (Hindi) handwritten character recognition is performed with the help of Histogram of Oriented Gradients (HOG). The model uses segmentation, pre-processing, feature extraction (It is performed by partitioning the image into six parts), classification and recognition steps. Artificial Neural Network is used for Individual characters classification. The model obtain classification accuracy of 97.06%[9]. Similarly, Urdu character recognition is performed using UNHD dataset. It has two steps such as character recognition using CNN for feature extraction and bi-directional Long-Short term memory technique for classification. It uses seven layers and followed by a B-LSTM layer. It achieves more than 83% accuracy[10]. Similarly, In another study, CNN is used to recognition of Devanagari script in India. Here, deep learning model is used as a feature extractor as well as a classifier for the recognition of 33 classes of basic characters of Devanagari ancient manuscripts. Dataset contains 5484 characters used for the experimental work. The model achieves 93.73% for Devanagari ancient character recognition [13]. Some of the popular deep learning algorithms are described as follows:

DenseNet121: Huang et al. [60] discussed DenseNet model to vanish gradient problem. Here, each layer is interconnected in a feed-forward manner. Features maps of all preceding layers are used as input to each layer, and their feature maps are used as inputs into all subsequent layers. This collective knowledge retained several advantages: an improved flow of information in the network alleviated the problem of relearning redundant features and decreased the number of learnable parameters. The DenseNet121 was the first model released from the DenseNet family with 121 convolutional layers. After that, researchers began experimenting with added convolutional layers and eventually released DenseNet169 with 169 convolutional layers. The DenseNet201 with 201 layers is the most recent advancement in the DenseNet models and outperforms all other deep CNN models

EfficientNetB0: EfficientNet [16] is introduced in 2019. It efficiently handle classification tasks and demonstrates higher accuracy when used ImageNet dataset. EfficientNet has several input sizes (e.g., [B0] 224 × 224 to [B7] 600 × 600 input size). It is mainly uses compound scaling and there are many various methods to scale a convolution network in terms of height, width or depth. Compound scaling improves the performance of the network[17].

MobileNet: MobileNet is a deep convolutional neural network model open-sourced by Google [15]. It is designed for mobile or embedded vision applications. It uses depth-wise separable convolution for lightweight computations. Depth-wise separable convolution consists of two factorized convolutions such as standard depth-wise convolution and pointwise convolution. The first phase is the depth-wise spatial convolution, where the convolution is done depth-wise for each input channel with a single filter. The following phase uses pointwise convolution, which applies 1 × 1 convolution to combine the output of depth-wise convolution, thereby reducing the computational cost. This method of factorizing the convolution process into two phases reduces the model's size. MobileNet uses 28 convolutional layers with 3 × 3 depth-wise separable convolutions. All layers in the architecture are followed by batch normalization and ReLU nonlinear activation.

InceptionResNetV2: Inception-ResNet-v2 is a convolutional neural network. It is trained on more than a million images from the ImageNet database [18] [19]. This network is 164 layers deep and equals to raw charge of the newly announced InceptionV4 model

In pattern recognition, identifying handwritten numerals are a complex knot. Pikes are used to identify the Kannada Numerals. Here, handwritten Kannada characters are captivated in document fashion. Consequentially, pre-processing steps like noise removal, binarization, normalization, skew amendment, and thinning are performed. Features are extracted by using strategies such as Drift Length Count, Direction related progression code and Curvelet Transfiguration Wrapping. Here, deep convolution neural network classifier is adopted and obtain 96% accuracy [14].

In this section, various studies on handwritten text recognition, with a focus on methods for detecting struck-out text and methods for recognizing characters in several different scripts, including Kannada, English, Devanagari (Hindi), and Urdu. The studies use various techniques, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), support vector machines (SVMs), and artificial neural networks (ANNs).

The studies show that CRNNs can achieve superior performance for detecting struck-out text, and that deep learning techniques can achieve high accuracy for character recognition in different scripts. For example, the studies report that a CNN-based model can achieve a classification accuracy of 93.2% for Kannada characters, a model using both CNN and Tesseract can achieve 87% accuracy for Kannada characters, a model using Random Forest and SVM classifiers can achieve 78% accuracy for Kannada characters, a CNN-based model can achieve 98% accuracy for non-overlapping lines of Kannada characters, and a model using ANN and wavelet transform can achieve 91% accuracy for Kannada characters and 97.6% accuracy for Kannada numerals.

The studies also mention some of the popular deep learning algorithms like DenseNet121, EfficientNetB0, and MobileNet that are used in the field of handwritten text recognition.

Research gaps from the article are not explicitly mentioned but from the above, it can be inferred that there is a scope for improvement in the recognition of Kannada characters as the accuracy reported is lower than that reported for other scripts such as English and Devanagari. Additionally, more research is needed to improve the recognition of struck-out text, as the performance of the CRNN-based model is not explicitly reported. Another gap could be that, the study mainly focuses on only few scripts and Handwritten text recognition in other languages could be an area of future research.

Methodology

The diagram of the methodology is shown in Fig.1. It includes dataset, pre-processing (binarisation), pre-trained deep learning models, and strike-out/non-strike-out outputs can be used to represent a workflow for image classification using deep learning.

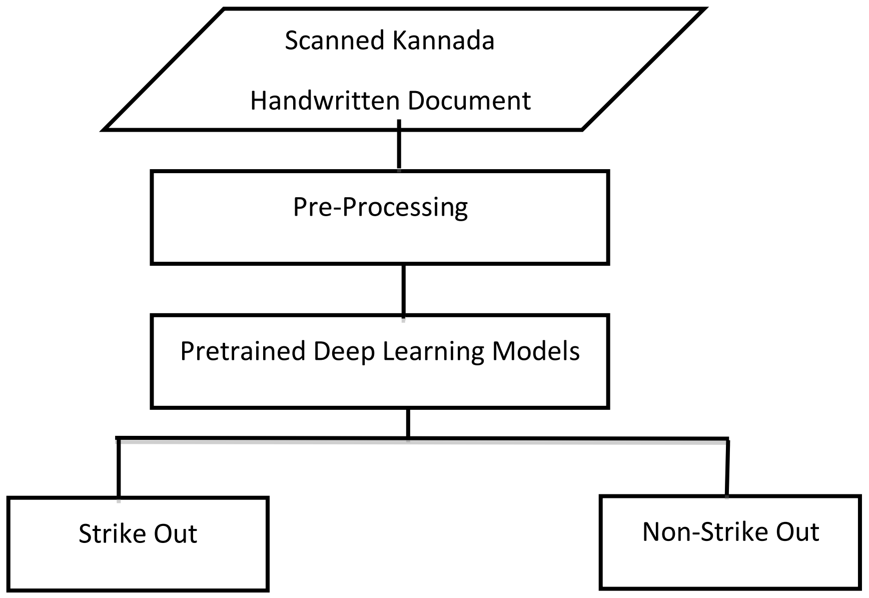
Scanned Kannada Handwritten Document
Pre-Processing
Pretrained Deep Learning Models
Strike Out
Non-Strike Out

Fig 1 Proposed Methodology

Data Collection

The standard dataset with respect to Kannada strikeout words are not available. Hence, creation of own dataset is carried out with the support of various professionals from colleges and universities. Collected dataset contains multiple paragraphs of Kannada words. Various experiments are conducted to normalized the image size and final fixed into 255X255 which demonstrate the optimum loss. The normalized dataset contains two classes i.e Strikeout and without Strikeout. Totally, 100 samples, each class has 50 samples. Sample input image is shown in Fig 2.

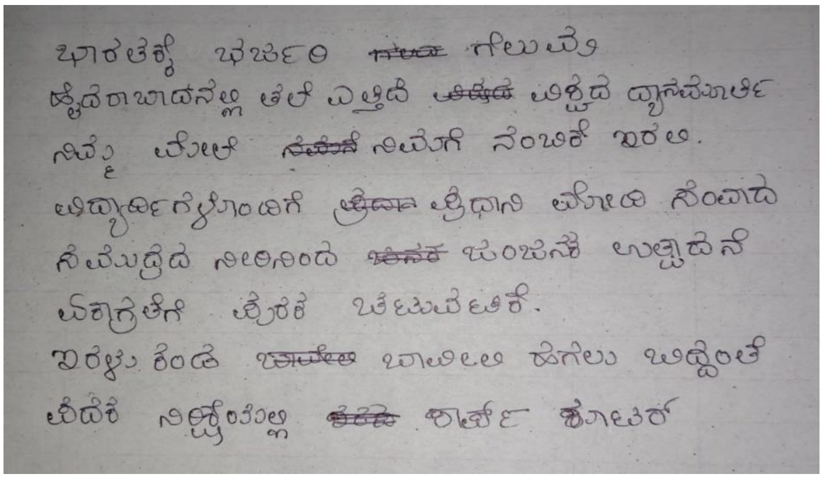

Fig 2; Sample Input Image

Pre-processing

It is the step where the images are prepared for use in the deep learning model. Binarization is the process of converting an image to black and white by thresholding the pixel intensities. It can be useful to eliminate noise and improve the contrast of an image.The sample of binarization of Kannada handwritten document is shown in Fig 3.

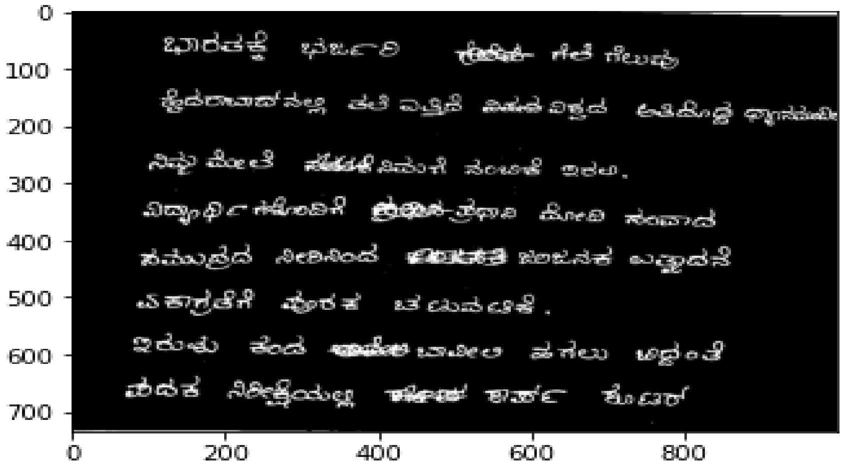

Fig 3: Binarization

Pre-trained deep learning models

Pre-trained deep learning models that have been trained on a large dataset, such as MobileNet, InceptionResNetV2, DenseNet121 and EfficientNetB0 and can be fine-tuned for a specific task, such as image classification.

Strike-out/non-strike-out

These are the outputs of final predictions of the deep learning model, indicating whether an input image is a strike-out or a non-strike-out .

Experimental Setup

The own dataset is created for the experiments. According to literature survey there are several deep CNN architectures achieve competing accuracies. To find the best architectures for the task at hand, the following models are analyzed and compared: MobileNet, InceptionResNetV2, DenseNet121 and EfficientNetB0. Each of the models above were pre-trained on created strikeout hand written datasets weights with the input image of size 255× 255 . All the architectures are trained for 10 epochs with Adam as the optimization function, Relu as activation function and the learning rate as 0.001 in Google Colab. Tensorflow2 and Keras2 are used to build and evaluate the models.

Algorithm:

1.**Input:** Create images of two categories {Stakeout, Non-Strikeout}

2.**Environment :** Use Google Colab and Install the required libraries

3.**Configuration:** Import the images, configure training and testing and validate the model

4.Configure batch_size=30

5.**Directories Configuration:** Create two directories(SD1, NonSD2), SD1 is for training/testing and NonSD2 for validation.

6.**Training and Testing:** Create the model using MobileNet or InceptionResNetV2, or DenseNet121 or EfficientNetB0 and dense layers with Relu activation function and output layer with a softmax activation function.

7. Compile the model using the ADAM optimizer with the learning rate as 0.001 and used loss function as Categorical_Crossentropy function

8. Ten epochs are used for Model fitting and to reduce learning rate ReduceLRonPlateau function

9. Model is saved for validation testing

10. Testing dataset is configured

Generate performance score values for each fold:

a. Classification report
b. Confusion matrix
c. AUC-ROC curve

Note: Following Steps are performed 10 times

11. Validation: Load the model (MobileNet or InceptionResNetV2, or DenseNet121 or EfficientNetB0)

12. Load validation datasets

13. Perform the validation

14. Generate performance score values

a. Confusion Matrix
b. AUC-ROC curve

Precision, Recall and F1-Score

Precision, recall, and F1 score are evaluation metrics used to measure the accuracy of a classifier.Precision measures the proportion of correct positive predictions among all positive predictions. It is defined as the number of true positive predictions divided by the sum of true positive and false positive predictions.Recall measures the proportion of correct positive predictions among all actual positive instances. It is defined as the number of true positive predictions divided by the sum of true positive and false negative predictions.F1 score is the harmonic mean of precision and recall, and represents the balance between precision and recall. It is defined as 2 times the product of precision and recall divided by the sum of precision and recall.

The formulas for precision, recall and F1 score are presented in equation 1,equation 2 and equation 3.

$$\text{Precision} = TP / (TP + FP) \qquad \ldots\ldots\ldots\ldots\ldots 1$$

$$\text{Recall} = TP / (TP + FN) \qquad \ldots\ldots\ldots\ldots\ldots 2$$

$$\text{F1-score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}) \qquad \ldots\ldots\ldots\ldots\ldots 3$$

where TP is the number of true positive predictions, FP is the number of false positive predictions, and FN is the number of false negative predictions.

Table 1 indicates Precision, Recall and F1-Score of four pre-trained deep learning models. For Strikeout class, EficientNetB0 model shows high F1-score (67%) value as compared to other state-of-art deep learning models for Strikeout class recognition. Similarly, InceptionResNetv2 shows higher F1-score (60%) value as compared to other pre-trained deep learning models for without strikeout class. Without strikeout class, InceptionResNetV2 indicates highest 70% F1-score as compared to other state-of art techniques.

Table 1 Precision, Recall and F1-Score for the DL models

	Strikeout			Without Strikeout		
Models	Precision	Recall	F1-score	Precision	Recall	F1-score
DenseNet121	0.55	0.55	0.55	0.55	0.60	0.57
EficientNetB0	0.50	1.00	0.67	0.00	0.00	0.00
MobileNet	0.38	0.30	0.33	0.42	0.50	0.45
InceptionResNetV2	0.60	0.60	0.60	0.60	0.60	0.60
Support vector machine	0.54	0.48	0.51	0.53	0.59	0.56

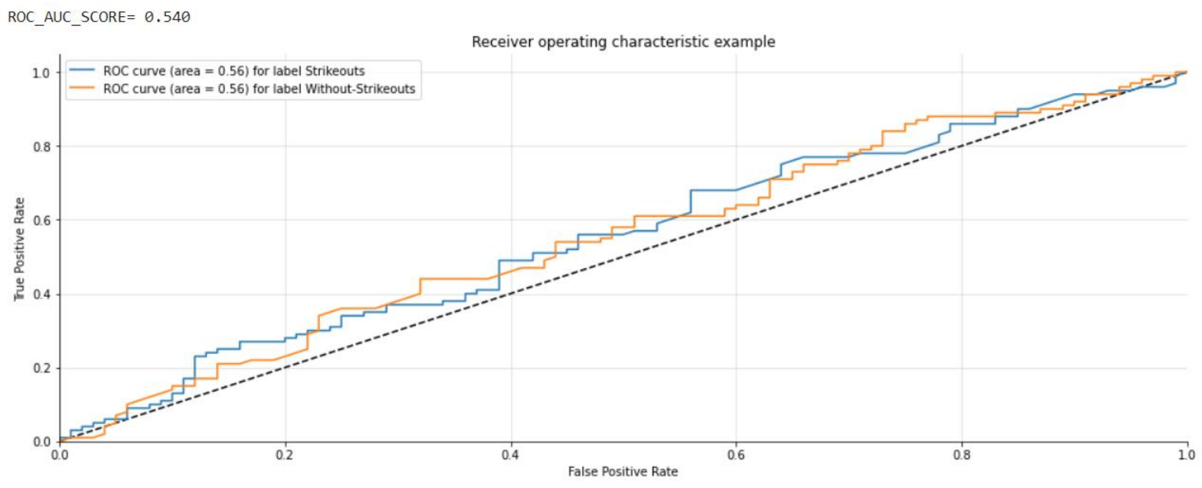

Fig. 4 ROC curve of Densenet121

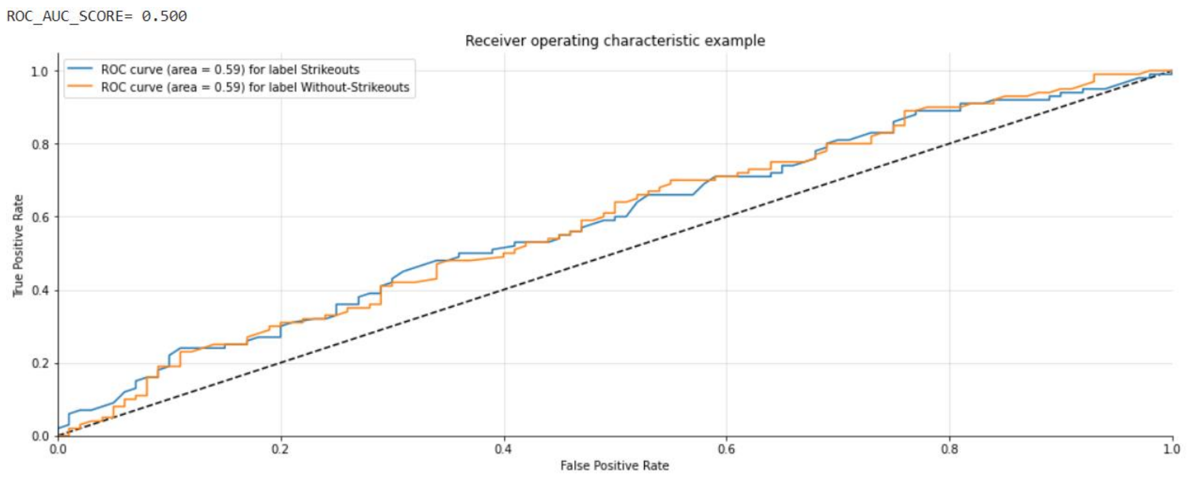

Fig. 5 ROC curve of EfficintNetB0

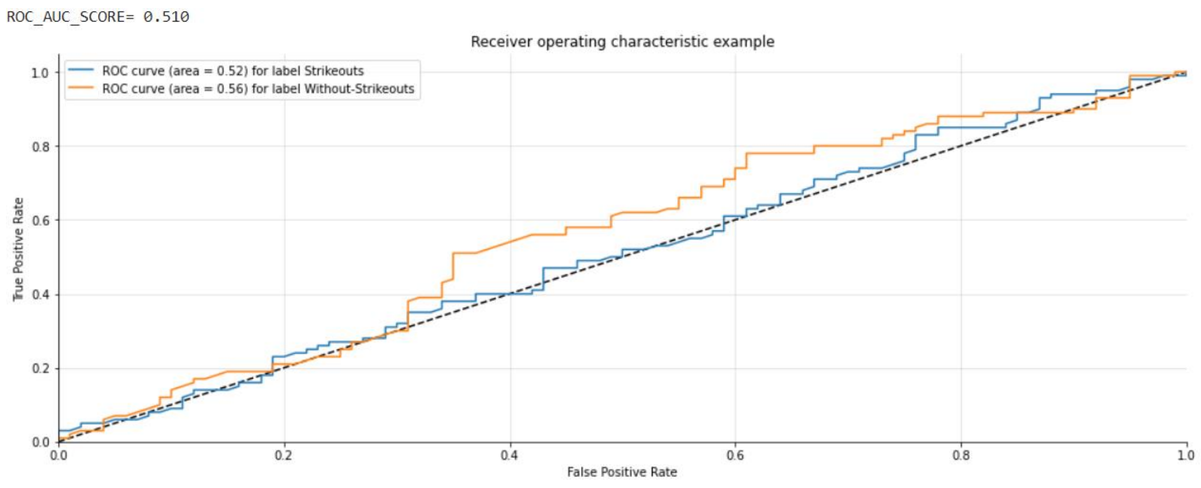

Fig.8 ROC curve of Support vector machine

Fig.4-Fig.8 shows Receiver operating characteristic (ROC) for the handwritten Kannada script recognition. The ROC curve describes the trade-off between true positive rate) and false positive rate. The curve closer to the 45-degree diagonal of the ROC space, the model is less accurate. InceptionResNetV2 model shows higher ROC score value as compared to all other pre-trained deep learning models.

Discussion

InceptionResNetV2 is another CNN model that has been trained on the large ImageNet dataset and has been shown to achieve high performance on various image classification tasks. It has a complex architecture that allows it to learn high-level features from the input images, which could be useful for detecting strikeout in handwritten documents.

In terms of performance evaluation, the F1-score is a commonly used metric for measuring the accuracy of a model in binary classification tasks like strikeout detection. The F1-score is the harmonic mean of precision and recall, and it gives a good indication of the trade-off between false positives and false negatives. A high F1-score indicates that the model is able to accurately detect strikeout in the majority of cases.

In summary,InceptionResNetV2 is a deep learning model that could potentially be used for detecting strikeout in Kannada handwriting and the F1-score can be used to evaluate the performance of the models.

Conclusion

In conclusion, the detection of strikeout in Kannada handwritten documents is a challenging problem due to the variability in handwriting styles and the complexity of the Kannada script. To address this problem, various approaches have been proposed, such as using computer vision and machine learning techniques, or Optical Character Recognition (OCR) with natural language processing, and using deep learning models like EfficientNetB0 and InceptionResNetV2. The present study is very useful for recognition of handwritten Kannada language with the support of pre-trained deep learning models. InceptionResNetV2 model shows efficient with respect to handwritten Kannada script recognition as compared to other state-of-models.

The performance of these models is evaluated using commonly used metrics like F1-score, which gives a good indication of the trade-off between false positives and false negatives.

In the future, more advanced techniques and models can be developed and implemented, such as using Attention-based models or Generative models and also more diverse dataset can be used to train these models. Additionally, research can be done to improve the robustness of these models to variations in handwriting styles and to improve the overall accuracy of strikeout detection in Kannada handwritten documents.

References

1. Asha K, Krishnappa HK (2018) Kannada Handwritten Document Recognition using Convolutional Neural Network. In: 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS). pp 299–301

2. A. F. Yaseen, A. Shaukat and M. Alam, "Emotion Recognition from Facial Images using Hybrid Deep Learning Models," 2022 2nd International Conference on Digital Futures and Transformative Technologies (ICoDT2), Rawalpindi, Pakistan, 2022, pp. 1-7 .

3. Chaudhuri BB, Adak C (2017) An approach for detecting and cleaning of struck-out handwritten text. Pattern Recognition 61:282–294 . https://doi.org/10.1016/j.patcog.2016.07.032

4. Fernandes R, Rodrigues AP (2019) Kannada Handwritten Script Recognition using Machine Learning Techniques. In: 2019 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER). pp 1–6

5. GururajMukarambi, B.V. Dhandra, MallikarjunHangarge, A Zone Based Character Recognition Engine for Kannada and English Scripts, Procedia Engineering, Volume 38, 2012, PP 3292-3299

6. Hassan S, Irfan A, Mirza A, Siddiqi I (2019) Cursive Handwritten Text Recognition using Bi-Directional LSTMs: A Case Study on Urdu Handwriting. In: 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML). pp 67–72

7. Hallur, V.C., HegadiR.S. Handwritten Kannada numerals recognition using deep learning convolution neural network (DCNN) classifier. CSIT 8, 295–309 (2020).

8. M. Naveenkumar, S. Srithar, B. Rajesh Kumar, S. Alagumuthukrishnan and P. Baskaran, "InceptionResNetV2 for Plant Leaf Disease Classification," 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2021, pp. 1161-1167

9. Nisa H, Thom JA, Ciesielski V, Tennakoon R (2019) A deep learning approach to handwritten text recognition in the presence of struck-out text. In: 2019 International Conference on Image and Vision Computing New Zealand (IVCNZ). pp 1–6

10. N SR, Nair B J B, M R A, M L P (2021) Kannada Confusing Character Recognition and Classification Using Random Forest and SVM. In: 2021 3rd International Conference on Signal Processing and Communication (ICPSC). pp 537–541

11. Narang, S.R., Kumar, M. & Jindal, M.K. DeepNetDevanagari: a deep learning model for Devanagari ancient character recognition. Multimed Tools Appl 80, 20671–20686 (2021)

12. Pasha S, Padma MC (2015) Handwritten Kannada character recognition using wavelet transform and structural features. In: 2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT). pp 346–351

13. Ramesh G, Kumar N. S, Champa HN (2020) Recognition of Kannada Handwritten Words using SVM Classifier with Convolutional Neural Network. In: 2020 IEEE Region 10 Symposium (TENSYMP). pp 1114–1117

14. Ramesh G, Sharma GN, Balaji JM, Champa HN (2019) Offline Kannada Handwritten Character Recognition Using Convolutional Neural Networks. In: 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE). pp 1–5

15. Reya Sharma, Baijnath Kaushik, Offline recognition of handwritten Indic scripts: A state-of-the-art survey and future perspectives, Computer Science Review, Volume 38, 2020

16. Singh N (2018) An Efficient Approach for Handwritten Devanagari Character Recognition based on Artificial Neural Network. In: 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN). pp 894–897

17. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556

18. Tan, M.; Le, Q.V. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 10691–10700