

Custom CNN for Gujarati Letter Recognition: Enhancing OCR Accuracy with Deep Learning

Meet Patel¹, Sachin Bhoite², Riddhi Panchal³ and Pravin Metkewar⁴

¹⁻⁴Department of Computer Science and Applications, MITWPU, Pune, Maharashtra
Email: sachintenjuly@gmail.com

Abstract— Detection as well as recognition of Gujarati letter (GL) forms is not only a crucial step in several projects, but also it can be considered to be an initial state of the languages processing pipeline. Optical Character Recognition (OCR) by Convolutional Neural Networks (CNN) has been seen to have promising outcomes which is the way by sending this method into automation. In this regard, we suggest a design of a custom CNN network comprising of the elements matching the local shape of the recognised GL. In the CNN model, a dataset of Gujarati characters is used as a domain for the hyperparameters to train, such as learning rate, batch size and dropout rate. The play through of the version under consideration is appraised using a classification an account, which will furnish information on the process, don't forget and F1-score for each magnificence. Last but not least, the metrics which encompass accuracy and loss is also computed to measure the general performance of the version being utilized. Through image processing automation that detects various types of Gujarati characters, this technology helps the process to save and evaluate the green data which is written in documents with Gujarati language as well. In addition, it enables the development of such factors as inclusive textbooks and tools for the community which is Gujarati-speaking. The proposed system of OCR device that exploits a specific CNN technique to resolve the demanding situations of Gujarati alphabet identification and classification shows how the deep learning algorithms could be useful in this process which can be invaluable both to language science and technological developments.

Index Terms— Gujarati script, letter detection, Convolutional Neural Network(CNN), machine learning, Optical Character Recognition (OCR).

I. INTRODUCTION

OCR technology has revolutionized the way we process and analyze textual information, enabling automated conversion of printed or handwritten text into machine-readable formats. However, the effectiveness of OCR systems largely depends on their ability to accurately recognize characters, especially in languages with complex scripts. Gujarati, an Indo-Aryan language primarily spoken in the Indian state of Gujarat, presents a unique challenge for OCR due to its intricate and diverse set of characters. Recognizing handwritten GL using OCR requires robust algorithms capable of understanding and distinguishing between the various shapes and forms of these characters. Traditional OCR methods often struggle with handwritten scripts due to variability in writing styles, strokes, and contextual factors. In recent years, the advent of DL techniques, particularly CNNs, has shown promise in improving the accuracy and efficiency of handwritten character recognition.

This research endeavors to explore the application of CNN-based OCR methods for detecting GL from handwritten documents. By harnessing the power of DL and CNNs, we aim to overcome the challenges posed by the intricate nature of Gujarati script and develop a system capable of accurately identifying individual characters.¹ The significance of this study lies in its potential to facilitate the digitization of handwritten Gujarati text, thereby enabling automated processing and analysis of documents in this language. Such advancements hold particular importance in fields such as linguistics, historical document analysis, and information retrieval, where access to digitized Gujarati text can unlock new avenues for research and development. Through this research, we seek to contribute to the growing body of knowledge in the field of OCR and deep learning, while also addressing the specific needs and challenges associated with Gujarati script recognition. By leveraging CNN algorithms and innovative methodologies, we aim to pave the way for more accurate, efficient, and accessible OCR solutions tailored to the complexities of handwritten Gujarati text. Top of Form.

A. Operating Concepts of the OCR System

The working principle of an OCR method involves a series of steps aimed at converting images containing text into machine-readable text. Here's an overview of the working principle of OCR:

Preprocessing

- The input image containing text undergoes pre-processing to enhance its quality and improve OCR accuracy.
- Pre-processing steps may include
 - Image binarization: Converting the image into binary format (black and white) to separate text from the background.
 - Noise removal: Removing any unwanted elements or artifacts from the image that may interfere with text recognition.
 - Skew correction: Adjusting the orientation of the text to make it horizontally aligned.
 - Contrast enhancement: Increasing the contrast between text and background to improve readability.

Text Localization

- The pre-processed image is analyzed to identify regions containing text (text localization).
- Techniques such as edge detection, connected component analysis, or neural networks may be used to locate text regions accurately.

Character Segmentation

- Once text regions are identified, individual characters are segmented from the text blocks.
- Segmentation involves separating each character to isolate them for recognition.
- Techniques like bounding box detection, contour analysis, or neural networks are employed for accurate character segmentation.

Feature Extraction (FE)

- Extract relevant features from segmented characters to represent them in a format suitable for recognition.
- Features may include shape descriptors, texture features, or other characteristics that help distinguish one character from another.
- FE aims to capture unique attributes of each character that aid in its identification.

Recognition

- The extracted features are fed into a recognition algorithm, which matches them against a predefined set of character templates or employs machine learning models.
- Traditional OCR methods may use pattern matching or template matching techniques to recognize characters based on their features.
- Modern OCR systems often leverage ML algorithms, such as ANN (including CNNs), RNNs, or DL models, to recognize characters based on learned patterns and features.

Post-processing

- Post-processing steps may be applied to refine OCR results and improve accuracy.
- Techniques such as spell-checking, language modeling, context analysis, or error correction algorithms are used to correct any inaccuracies in the recognized text.
- Post-processing aims to ensure the accuracy and coherence of the OCR output.

Output

- The final output of the OCR process is machine-readable text corresponding to the text content in the input image.
- The recognized text can be further processed, stored, analyzed, or used in various applications such as document digitization, text mining, indexing, or translation.

By following these steps, an OCR method effectively transforms images containing text into editable and searchable text data, enabling automation and digitization of textual information across various domains.

TABLE I: BACKGROUND AND LITERATURE SURVEY

Sr no.	Paper title and year	Methods used	Limitations
1	GHCR from Text Images (2020)	CNN and Multi-Layer Perceptron (MLP)	Less robustness for different handwritings
2	Gujarati Script Recognition (2023)	CNN based EfficientNet B3 and YOLO v4	YOLO has a limitation of the number of classes.
3	CNN-Based Optical Character Recognition for Isolated Printed GCN (2022)	CNN(both for handwritten and printed)	Less robust for different handwriting variations
4	Gujarati OCR: Compound CR Using Zernike Moment Feature Extractor (2020)	Zernike moments	Misclassification of similar looking characters (like 'dha')
5	Offline HGCR using Hybrid Classification Technique (2023)	SVM+HMM, XGBoost, RF	Difficult to identify compound words and sentences, Limited dataset
6	HGCR Using ML and DL (2023)	MobileNet, DenseNet, and NasNetMobile, XGBoost, RF, LR	Combination of Consonants and Vowels are no being identified
7	Hybrid Deep Resnet With Inception Model For OCR In GL (2022)	DL- Adaptive Weiner Filter, Contrast Limited Adaptive Histogram Equalization, Deep ResNet with Inception model (GoogleNet)	Less convolutional layers as compared to other existing technologies.
8	CNN Based Meitei Mayek HCR(2018)	CNN	Had to recreate the whole dataset as there was no dataset available for proper Meitei Mayek

II. METHODOLOGY

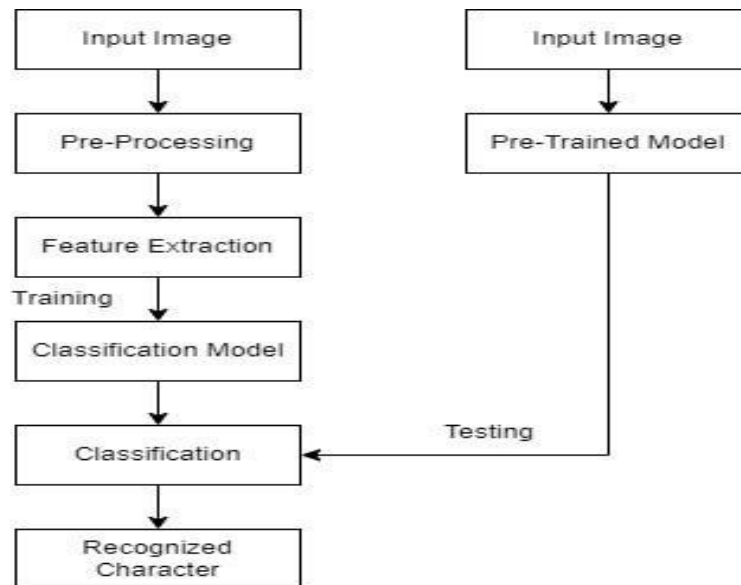


Figure 1. Block Diagram of the Methodology

A. Dataset Preparation

The dataset is taken from Kaggle open source. Which consists of 34 GCs, each with 11 different modifiers (Maatra) to produce a total of 385 characters. There are also 11 distinct vowels. To maintain uniformity, the image sizes are maintained at 32 by 32. The dataset is also subjected to picture augmentation (zooming, rotation, brightness, and horizontal-vertical shift) in order to create additional character variations.

Data Collection

- Utilize the dataset from Kaggle, as previously mentioned in your draft. This dataset comprises 34 Gujarati characters and 11 distinct vowels, along with modifiers leading to 385 character variations

Data Augmentation

- Apply image augmentation techniques to increase the diversity of the dataset, which helps in improving model robustness. Techniques include:
 - Rotation
 - Zooming
 - Brightness adjustment
 - Horizontal and vertical shifts

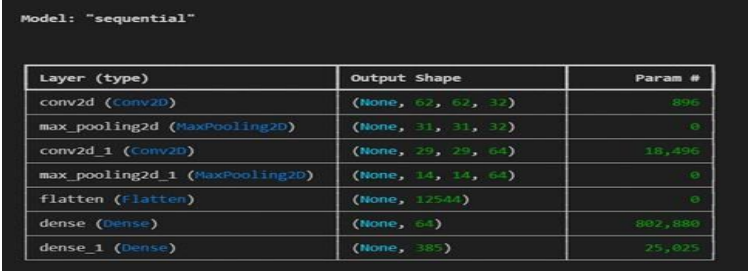
Pre Processing

- Normalize the images to have pixel values between 0 and 1.
- Resize images to 32x32 pixels as mentioned, maintaining uniformity across different images.

B. Model Design

CNN Architecture

- Design a custom CNN tailored for the intricacies of the GS.
- Incorporate convolutional layers that are effective in capturing spatial hierarchies in images. Use filters that progressively increase to capture more complex features at deeper layers.
- Implement pooling layers to reduce spatial dimensions and parameter count, which aids in controlling overfitting.
- Include dropout layers to prevent overfitting by randomly dropping units during the training process.
- Use activation functions such as ReLU for non-linear transformations within the network, enhancing learning capability.



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 29, 29, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
flatten (Flatten)	(None, 12544)	0
dense (Dense)	(None, 64)	802,800
dense_1 (Dense)	(None, 385)	25,025

Figure 2. Model Architecture

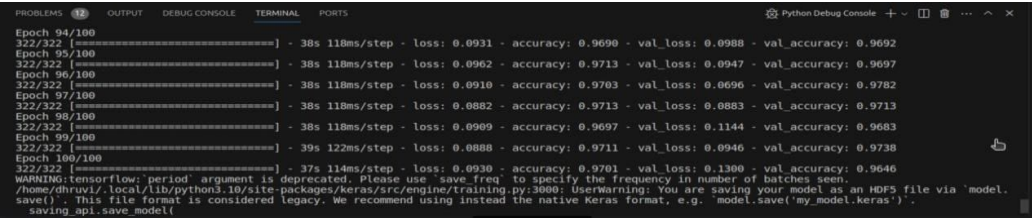
Hyperparameter Tuning

- Experiment with different values for learning rate, batch size, and dropout rate to find the optimal configuration for training.
- Utilize validation data split from the main dataset for tuning and early stopping to curb overtraining.

C. Training

Loss and Optimization

- Use categorical cross-entropy as the loss function, which is typical for multi-class classification problems.
- Opt for an optimizer such as Adam or SGD for effective backpropagation through the network, adjusting weights and reducing loss.



```
Epoch 94/100 - 38s 118ms/step - loss: 0.0931 - accuracy: 0.9690 - val_loss: 0.0988 - val_accuracy: 0.9692
Epoch 95/100 - 38s 118ms/step - loss: 0.0962 - accuracy: 0.9713 - val_loss: 0.0947 - val_accuracy: 0.9697
Epoch 96/100 - 38s 118ms/step - loss: 0.0910 - accuracy: 0.9703 - val_loss: 0.0696 - val_accuracy: 0.9782
Epoch 97/100 - 38s 118ms/step - loss: 0.0882 - accuracy: 0.9713 - val_loss: 0.0883 - val_accuracy: 0.9713
Epoch 98/100 - 38s 118ms/step - loss: 0.0909 - accuracy: 0.9697 - val_loss: 0.1144 - val_accuracy: 0.9683
Epoch 99/100 - 39s 122ms/step - loss: 0.0888 - accuracy: 0.9711 - val_loss: 0.0946 - val_accuracy: 0.9738
Epoch 100/100 - 37s 114ms/step - loss: 0.0930 - accuracy: 0.9701 - val_loss: 0.1300 - val_accuracy: 0.9646
WARNING:tensorflow: period argument is deprecated. Please use save_freq to specify the frequency in number of batches seen.
/home/dhruxy/.local/lib/python3.10/site-packages/keras/src/engine/training.py:3080: UserWarning: You are saving your model as an HDF5 file via 'model.save()'. This file format is considered legacy. We recommend using instead the native Keras format, e.g. 'model.save('my_model.keras')'.
  saving_utils.save_model(
```

Figure 3. Training

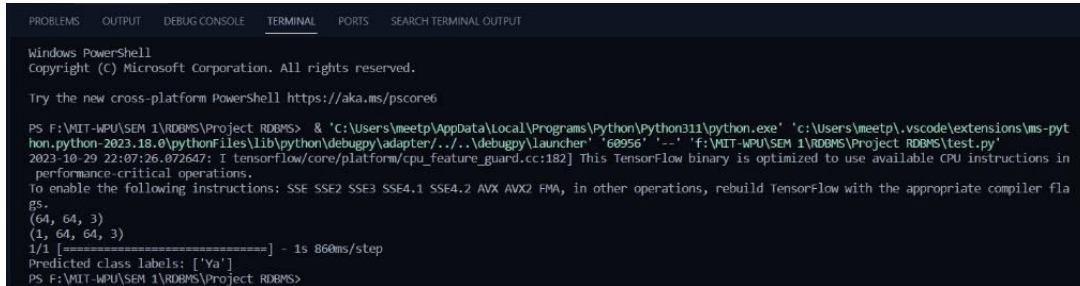
III. EVALUATION AND RESULT

A. Testing

- Set aside a portion of the dataset for final testing which is not used during the training phase.
- Evaluate model performance using the test data to ensure it generalizes well on new, unseen data.



Figure 4. Test Input Image



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS F:\MIT-WPU\SEM 1\ROBMS\Project ROBMS> & 'C:\Users\meetp\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\meetp\.vscode\extensions\ms-pyt
hon.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '60956' '--' 'f:\MIT-WPU\SEM 1\ROBMS\Project ROBMS\test.py'
2023-10-29 22:07:26.072647: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: SSE SSE2 SSE3 SSE4.1 SSE4.2 AVX AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler fla
gs.
(64, 64, 3)
(1, 64, 64, 3)
1/1 [=====] - 1s 860ms/step
Predicted class labels: ['Ya']
PS F:\MIT-WPU\SEM 1\ROBMS\Project ROBMS>
```

Figure 5. Output

B. Confusion Matrix

- Plot a confusion matrix to visualize the performance of the model across all classes, which helps in identifying specific classes where the model may be underperforming.

C. Performance Metrics Computation

- Compute detailed metrics including accuracy, loss across epochs, precision, recall, and F1-score for a comprehensive evaluation.

IV. CONCLUSIONS

In summary, the Convolutional Neural Network (CNN) model created for the recognition of GL proves to be a strong and promising approach to tackle the intricacies found in handwritten characters. Employing a structured approach that involves collecting datasets, preprocessing, designing architecture, training, and evaluation, the model showcases its effectiveness in understanding the subtleties of the Gujarati script. The project contributes to advancing character recognition technology for languages with complex scripts, paving the way for enhanced human-computer interaction and document understanding.

REFERENCES

- [1] S. S. Magare, Y. K. Gedam, D. S. Randhave, Prof. R. R. Deshmukh, 2014, Character Recognition of Gujarati and Devanagari Script: A Review, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 03, Issue 01 (January 2014).
- [2] M. J. Baheti and K. Kale, "Gujarati Numeral Recognition: Affine Invariant Moments Approach," International Journal of electronics, Communication & Soft Computing Science & Engineering, pp. 140-146, March 2012.
- [3] J. Memon, M. Sami, R. A. Khan and M. Uddin, "Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)," in IEEE Access, vol. 8, pp. 142642-142668, 2020, doi: 10.1109/ACCESS.2020.3012542.
- [4] Pareek, J., Singhanian, D., Kumari, R.R. and Purohit, S., 2020. Gujarati Handwritten Character Recognition from Text Images. Procedia Computer Science, 171, pp.514-523.
- [5] Audichya, Milind & Saini, Jatinderkumar. (2019). An Overview of Optical Character Recognition for Gujarati Typed and Handwritten Characters.
- [6] Ami Mehta, Ashish Gor "Multi font Multi size Gujarati OCR with Style Identification" International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS2017)
- [7] Solanki, P., & Bhatt, M. (2013). Printed Gujarati Script OCR using Hopfield Neural Network. International Journal of Computer Applications, 69(13), 33–37. <https://doi.org/10.5120/11905-7982>
- [8] Khopkar, M., 2013. OCR for Gujarati Numeral using Neural Network. IJSRD - International Journal for Scientific Research & Development, Vol. 1, Issue 3, 2013, pp.424–427.

- [9] Kumar, Anubhav & Sharma, Anuradha & Chawla, Monika & Prasad, T. (2009). Different Approaches in OCR of Indian Languages. 10.13140/RG.2.1.3243.5046.
- [10] Choksi, Amit & Thakkar, Shital. (2012). Recognition of Similar appearing Gujarati Characters using Fuzzy-KNN Algorithm.
- [11] S. J. Macwan and A. N. Vyas, "Classification of offline gujarati handwritten characters," 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2015, pp. 1535-1541, doi: 10.1109/ICACCI.2015.7275831.
- [12] Parida, Shantipriya & Dash, Satya & Bojar, Ondřej & Motlíček, Petr & Pattnaik, Priyanka & Mallick, Debasish. (2020). OdiEnCorp 2.0: OdiEnglish Parallel Corpus for Machine Translation.
- [13] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.
- [14] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., ... Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, 77, 354–377. doi:10.1016/j.patcog.2017.10.013.
- [15] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).
- [16] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "Mobilenetv2: Inverted residuals and linear bottlenecks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510-4520. 2018.
- [17] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." In *International conference on machine learning*, pp. 6105-6114. PMLR, 2019.
- [18] Tan, Mingxing, and Quoc Le. "Efficientnetv2: Smaller models and faster training." In *International Conference on Machine Learning*, pp. 10096-10106. PMLR, 2021.
- [19] Dongre, Vikas J., and Vijay H. Mankar. "Devnagari document segmentation using histogram approach." *arXiv preprint arXiv:1109.1247* (2011).
- [20] Likforman-Sulem, L., Zahour, A. & Taconet, B. Text line segmentation of historical documents: a survey. *IJDAR* 9, 123– 138 (2007). doi.org:10.1007/s10032-006-0023-z
- [21] Manivannan Arivazhagan, Harish Srinivasan, and Sargur Srihari "A statistical approach to line segmentation in handwritten documents", *Proc. SPIE 6500, Document Recognition and Retrieval XIV*, 65000T (29 January 2007); doi:10.1117/12.704538
- [22] Perruchet, Pierre, and Annie Vinter. "PARSER: A model for word segmentation." *Journal of memory and language* 39, no. 2 (1998): 246- 263.
- [23] T. M. Breuel, "Segmentation of handprinted letter strings using a dynamic programming algorithm," *Proceedings of Sixth International Conference on Document Analysis and Recognition*, 2001, npp. 821-826, doi: 10.1109/ICDAR.2001.953902.
- [24] Polesel, Andrea, Giovanni Ramponi, and V. John Mathews. "Image enhancement via adaptive unsharp masking." *IEEE transactions on image processing* 9, no. 3 (2000): 505-510.
- [25] Hummel, Robert. "Image enhancement by histogram transformation." Unknown (1975).
- [26] Munteanu, Cristian, and Agostinho Rosa. "Gray-scale image enhancement as an automatic process driven by evolution." *IEEE transactions on systems, man, and cybernetics, part B (cybernetics)* 34, no. 2 (2004): 1292-1298.
- [27] White, James M., and Gene D. Rohrer. "Image thresholding for optical character recognition and other applications requiring character image extraction." *IBM Journal of research and development* 27, no. 4 (1983): 400-411.
- [28] Nagy, George, Thomas A. Nartker, and Stephen V. Rice. "Optical character recognition: An illustrated guide to the frontier." In *Document recognition and retrieval VII*, vol. 3967, pp. 58-69. SPIE, 1999.
- [29] Singh, Sukhpreet. "Optical character recognition techniques: a survey." *Journal of emerging Trends in Computing and information Sciences* 4, no. 6 (2013): 545-550.
- [30] Lopresti, Daniel. "Optical character recognition errors and their effects on natural language processing." *International Journal on Document Analysis and Recognition (IJ DAR)* 12, no. 3 (2009): 141-151.