Hybrid Approach for Recognition of Isolated Handwritten Fraction Notations in Telugu Script

Vempati Lakshmi Sravani
School of Computer and Systems Sciences
Jawaharlal Nehru University (JNU)
New Delhi, India

Piyush Pratap Singh
School of Computer and Systems Sciences
Jawaharlal Nehru University (JNU)
New Delhi, India

ABSTRACT

Recognition of handwritten digits in regional Indian languages presents formidable challenges due to the vast array of scripts and variability in writing styles. Despite notable advancements in research techniques and databases for recognizing "0 to 9" digits in Indian scripts such as Bangla, Kannada, Devanagari, Oriya, and Telugu, there exists a conspicuous gap in research explicitly addressing the recognition of "fraction" notations unique to the Telugu script. Consequently, the proposed methodology integrates deep learning techniques, employing Convolutional Neural Networks (CNNs) for feature extraction and Support Vector Machines (SVMs) with various kernels for classification. This approach is explicitly tailored towards recognizing handwritten Telugu "fraction" notations, aiming to fill the existing research void in this domain. To facilitate model training, a comprehensive dataset is curated comprising 4000 handwritten Telugu fraction images covering eight distinct notation classes and enhanced dataset diversity through data augmentation techniques. Extensive experimental validation showcases the efficacy of the proposed hybrid CNN-SVM framework, achieving an impressive accuracy of 99.86% using the RBF kernel, outperforming the standalone CNN model (99.67% accuracy). These findings highlight the effectiveness of the proposed method in this underexplored field of recognizing handwritten Telugu fractions that can contribute to the digital preservation of ancient Telugu manuscripts and fostering linguistic diversity, paving the way for broader language technology applications.

General Terms

Handwriting Digit Recognition, Indian Regional Languages

Keywords

Telugu Fraction Notations, Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), Kernel Functions

1. INTRODUCTION

Numerous studies have delved into the intricacies of recognizing English numerals, showcasing significant strides in this domain. However, regarding the numerals of regional languages of India, the research landscape needs to be explored more. India, a mosaic of languages and cultures, boasts 22 major languages scribed in 13 diverse scripts. A staggering 460 million individuals in the country rely on these regional scripts for various written communications, whether completing bank forms or sending traditional postcards. Nevertheless, the numerals adorning these documents often pose a challenge for identification due to their script-specific characteristics. Thus, there is a pressing need for a robust recognition system tailored to decipher these regional digits accurately. With concerted research efforts and the availability of comprehensive datasets, the recognition of handwritten text in Indian regional languages can witness substantial growth, facilitating applications in areas such as document digitization, language localization, and accessibility for diverse linguistic communities.

Recognizing handwritten text in Indian regional languages presents unique challenges due to the diverse nature of scripts and variations in handwriting styles. Telugu, a prominent Dravidian language primarily spoken in Andhra Pradesh and Telangana states of India, features a distinctive script marked by its distinct characters and writing conventions. Telugu script consists of basic notations representing vowels, consonants, cardinal numbers, and fractions. Handwritten digit notations in Telugu may exhibit variations in stroke thickness, curvature, and overall structure, making it challenging to develop accurate recognition algorithms. Additionally, variations in handwriting styles further compound the difficulty, requiring robust algorithms capable of accommodating such variability.

Table 1. Telugu Cardinal Numbers

Digit	Telugu Figure	Name
0	౦	సున్నా **sunna**
1	౧	ఒకటి **okaṭi**
2	౨	రెండు **reṇḍu**
3	౩	మూడు **mūḍu**
4	౪	నాలుగు **nālugu**
5	౫	ఐదు **aidu**
6	౬	ఆరు **āru**
7	౭	ఏడు **ēḍu**
8	౮	ఎనిమిది **enimidi**
9	౯	తొమ్మిది **tommidi**

Table 2. Telugu Fractional Notations

Value	Telugu Figure	Name
0	౸	హళ్లి **haḷḷi**
¼	౷	కాలు **kālu**
¾ or ½	౹	అర **ara**
¾	౺	ముక్కాలు **mukkālu**
0	౦	సున్నా **sunna**
1/16	౼	వీసము **vīsamu**
2/16 or 1/8	౽	పరక **paraka**
3/16	౾	మువ్వీసము **muvvīsamu**

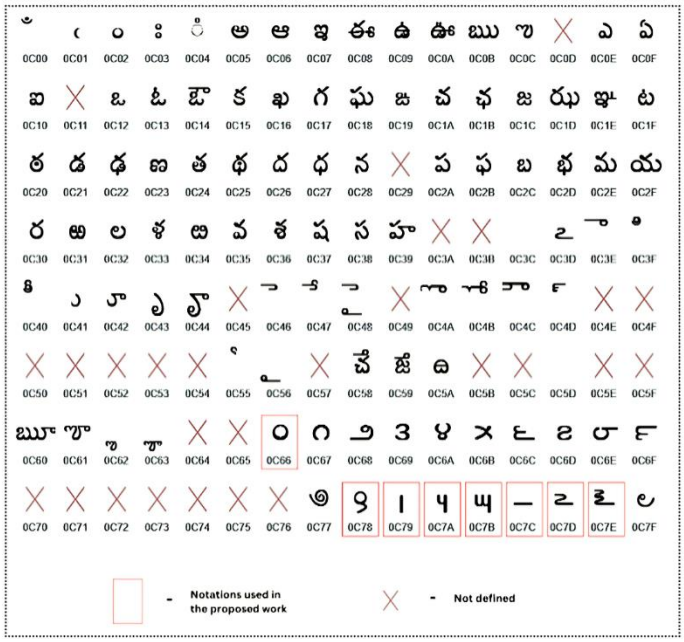

Fig 1: Telugu Unicode code chart - as of Unicode version 15.1

One of the critical challenges in handwritten digit recognition is the scarcity of datasets for Indian regional languages. Unlike English, where large labeled datasets such as MNIST are readily available, building labeled datasets for Indian regional languages requires significant effort due to the diversity of scripts and languages. This issue was addressed by crowdsourcing and collaborating with linguistic experts to create a high-quality dataset of Telugu Fraction notations.

The proposed approach is to integrate emerging technologies like deep learning and data augmentation to improve Telugu handwritten text recognition systems by extracting relevant features from handwritten digit images and classifying them into appropriate categories. Convolutional neural networks (CNNs) are utilized, which are particularly effective in capturing spatial dependencies in image data. By leveraging state-of-the-art techniques and building comprehensive datasets, handwritten digit recognition in Telugu can continue to evolve, contributing to the broader goal of advancing language technology and preserving cultural heritage.

2. BACKGROUND

Like many other languages, Telugu has its own script for writing numbers. However, specifically in the Telugu script, the numbers 0 through 9 are represented by unique notations distinct from those used for “fractions” [1]. Telugu uses notations to represent numbers, as shown in Table 1. These notations are used to write numerical values, similar to how people use them in English. In addition to these numerals, Telugu employs a separate set of notations dedicated explicitly to representing fractions, as in Table 2. These fraction notations are unique to Telugu script and are used to denote fractions without ambiguity [2].

In October 1991, the Telugu script was officially integrated into the Unicode Standard with the unveiling of version 1.0. This significant milestone marked the inclusion of Telugu characters into the global standard for character encoding, facilitating digital communication and representation of the Telugu language across various platforms and devices. Later, in 2008, Unicode version 5.1 introduced additional symbols (U+0C78 to U+0C7E), which are part of the "Telugu fractions" subblock within the "Telugu" block. The Telugu block resides within the Basic Multilingual Plane (BMP), encompassing Unicode's first 65,536 code points. This BMP serves as the foundation for accommodating a vast range of characters used in various writing systems across the globe. The Unicode Chart of Telugu, as per Unicode version 15.1, is provided in the Fig.1 [3].

The distinction between regular numerical and fraction notations is crucial for clarity and precision in Telugu writing. Using distinct notations ensures an unambiguous representation of both whole numbers and fractions within a numerical expression in Telugu script. The subdivision of unity is carried to a great extent in everyday practice among the Telugu people [4]. Similar to how decimal fractions in the decimal system descend by tens, Telugu fractions descend by fours. Ancient India used the number sixteen as a general divider (like 16 annas formed a rupee). Back then, four of these annas made a quarter, being doubled formed a half, and three quarters are 12 annas. From Table 2, it will be perceived that perpendicular lines symbolize the subdivision of a unit into fourths and sixteenths by horizontal lines. In either case, the number of these lines corresponds with the number of fractional parts to be represented.

Two different zeros were used, where the notation ౯ “haḷḷi” separates the integer and the fraction. This notation answers to the cypher in the section of annas, and a cypher in the section of paise is a “sunna,” as the Indian rupee was once divided into 16 annas and the anna into twelve paise. The second zero is the same as the U+0C66, Telugu Digit Zero, ౦. This system was widely used in India before decimalization when the rupee was the predominant currency [5]. It provided a convenient way to divide currency and measure transactions. Though the usage of these notations is not that regular in the current era, recognition and classification of these notations are necessary to accurately digitize many old books and ancient palm leaf manuscripts, preserving their original meaning and content [6].

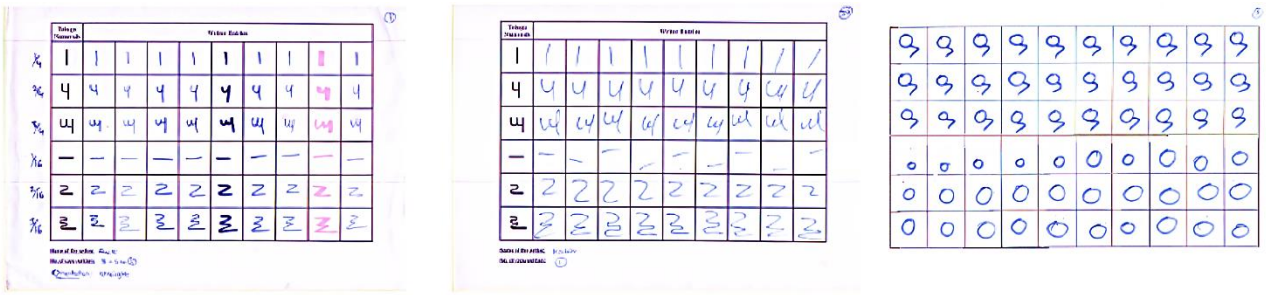

Fig 2: Sample sets of all Telugu Fraction Notations handwritten by different users

3. LITERATURE SURVEY

While significant advancements have been made in the field of recognizing handwritten numerals in languages such as English, Arabic, and Latin, there remains a notable gap in research when it comes to recognizing numerals in regional Indian languages. There has been a glaring absence of work focusing specifically on recognizing “fractions” in the Telugu script or developing datasets, despite having remarkable progress in recognizing “0 to 9” digits or their databases in scripts such as Bangla, Kannada, Devanagari, and Oriya, including Telugu.

Like in [7], the authors present two new handwritten numeral databases (Devnagari and Bangla scripts) and a multistage recognition system for these scripts. Their system achieves high accuracy based on wavelet representations and multilayer perceptron classifiers, exceeding 98.5% for isolated numerals and mixed numerals combining numerals from Devanagari, Bangla, and English scripts. Another approach [8] utilizes a modified quadratic classifier scheme that achieves an accuracy exceeding 98% recognition for handwritten numerals across all six popular Indian scripts, proving its effectiveness in handling the complexities of diverse writing systems.

One of the notable hybrid approaches [9] is where Principal Component Analysis (PCA) or Modular PCA (MPCA) is utilized for capturing statistical information alongside Quad-tree Longest-Run (QTLR) features to represent topological properties. This combined feature approach, evaluated with a Support Vector Machine (SVM) classifier, achieves high recognition accuracy, exceeding 97% on average for handwritten numerals in five major scripts (Arabic, Bangla, Devanagari, Latin, and Telugu).

For Odia script [10], an approach that extracts features from segmented numeral images using chain code histograms was suggested. These features are then fed into either a neural network or a quadratic classifier for recognition, achieving an accuracy of about 94%, and for the Devanagari script [11], the authors explored the suitability of SVMs for classification. Utilizing a linear SVM kernel, they achieve a high recognition accuracy of 99.48%.

In a related study [12], an approach leveraged global and local structural characteristics combined with a Probabilistic Neural Network (PNN) classifier for recognizing handwritten numerals in Kannada, Telugu, and Devanagari scripts. The findings indicated that the algorithm's performance exhibited improvement with the least radial value. In [13], a method utilizing a multilayer feed-forward back-propagation was tested across five scripts, Devanagari, English, Urdu, Tamil, and Telugu, achieving an average recognition rate of 96.53% with two hidden layers and 94% with one hidden layer.

In 2018, another hybrid approach [14] for recognizing handwritten Devanagari numerals was proposed, combining a Convolutional Neural Network (CNN) with a Genetic Algorithm (GA) and, in the same year, a zonal fractal dimension (ZFD) based feature extraction technique [15] was demonstrated that reported an accuracy of 99.23% for handwritten Telugu digit recognition.

4. DATASET

4.1 Data Collection & Labeling

A significant hurdle in Telugu handwritten digit recognition is the need for labeled datasets. Building comprehensive datasets of handwritten Telugu digit notations is crucial for training and evaluating recognition models. The initiative to collect and annotate the dataset has been done, which involved collaboration with native speakers and linguists to ensure the data was authentic and representative, as shown in Fig. 2. A total of 80 users contributed to the dataset with their respective handwriting.

4.2 Dataset Description

In the realm of handwritten numeral recognition, accommodating the diversity of individual handwriting styles is paramount. Therefore, this dataset encompasses various handwriting variations to ensure its adaptability. Initially, an all-inclusive dataset comprising 80 sets of handwritten numerals was gathered, totaling 4000 samples, each contributed by different individuals, showcasing their unique handwriting styles.

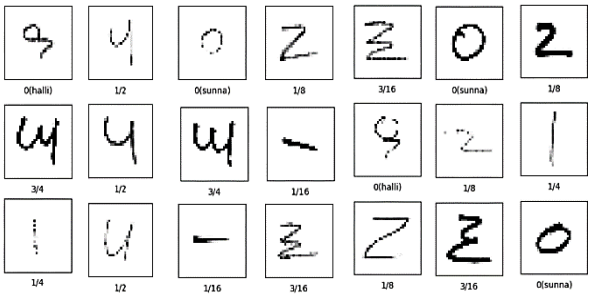

Fig 3: Samples from the dataset after preprocessing

The dataset compiled for this study is tailored to handwritten Telugu fraction numerals and to offer a well-balanced representation with 500 images allocated per class (per notation). This culminates in a comprehensive dataset comprising 4000 images. All images within the dataset are standardized to dimensions of 32*32 pixels and are presented in grayscale format. Binarization using Otsu's thresholding was performed to automatically threshold the image and separate the notation from the background [15].

4.3 Data Augmentation

However, despite its richness, the dataset's size remains relatively modest for developing a robust model. To address these limitations and enhance dataset robustness, data augmentation has been leveraged. The dataset is thus systematically expanded, introducing variations in orientation, size, and other parameters, thereby enriching the model's training data and improving its adaptability to diverse handwriting styles and layouts.

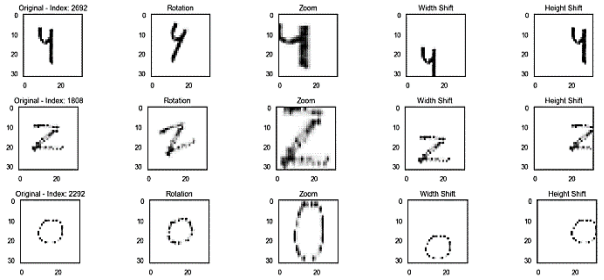

Fig 4: Sample of 3 images with their augmented images

Techniques such as rotation, scaling, and translation are employed to enhance the robustness of handwritten digit recognition models. As grayscale images are being used, there is no need for color augmentation or brightness enhancement. Data augmentation methodologies were implemented on the training dataset using the ImageDataGenerator class within TensorFlow's Keras API framework. These techniques included rotation, zooming, and horizontal and vertical shifts to reduce overfitting. By augmenting the training dataset with synthetically generated variations of handwritten digits, models can better generalize to unseen data and improve overall accuracy.

4.4 Data Manipulation

The dataset initially comprises images of varying dimensions, which are then standardized to a uniform size of 32x32 pixels. Subsequently, all images are converted to grayscale and organized into .csv files for easy handling. Each image consists of 1024 pixels, arranged in a grid of 32 pixels in height and 32 pixels in width. These pixels are represented by integer values ranging from 0 to 255, denoting their respective lightness or darkness levels, with higher values indicating darker shades.

Both datasets consist of grayscale images depicting hand-drawn Telugu fractions images. In the training dataset, denoted as train.csv, there are 1025 columns. The first column, "label," indicates the digit depicted in the associated image. The remaining columns contain the pixel values corresponding to each pixel in the image.

Each pixel column in the training set is named pixelx, where x varies from 0 to 1023, inclusive. To locate a specific pixel within the image, the index x is decomposed into i * 32 + j, where i and j range from 0 to 31, inclusive. Consequently, pixelx corresponds to the pixel positioned at row i and column j within the 32x32 matrix, with indexing starting from zero. Similarly, the test dataset, test.csv, is generated, following an 80:20 split from the original dataset.

5. METHODOLOGY

5.1 Proposed Scheme

The proposed workflow is centered around a hybrid classifier, which effectively addresses the limitations inherent in individual classifiers by synergistically leveraging their strengths. The figure below offers a comprehensive overview of its workflow. Initially, the digit images for training undergo preprocessing. Subsequently, essential features are extracted from the images by employing CNN. Ultimately, the classification process culminates with SVM, ensuring robust and accurate predictions.

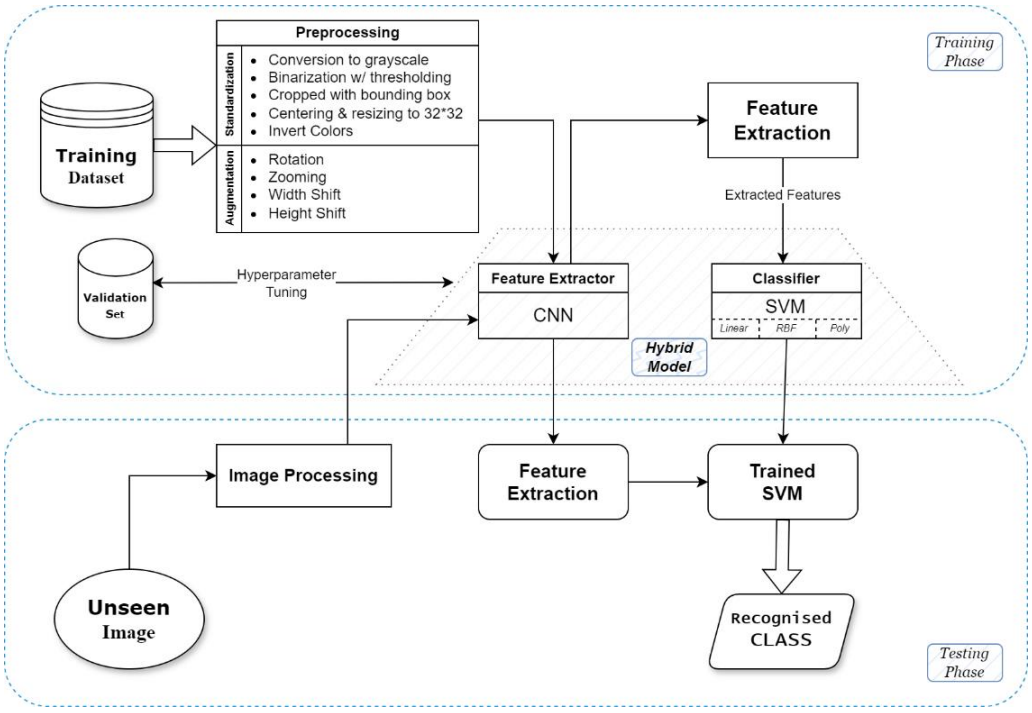

Fig 5: Proposed Workflow

5.2 Model Description

The architecture of the proposed CNN model is as follows:

- Input Layer - The input images used are sized 32x32 pixels and are grayscale, consisting of a single channel.
- Convolutional Layers – The architecture is designed with two sets of convolutional layers. Each set includes two convolutional layers, a max-pooling layer, and dropout regularization to mitigate overfitting. The initial set comprises two convolutional layers, each containing 32 filters sized 5x5. The subsequent set consists of two convolutional layers, each containing 64 filters sized 3x3.
- Flatten Layer - Following the last convolutional layer, the output is flattened to prepare for input into the fully connected layers.
- Fully Connected Layers - Two dense (fully connected) layers succeed the flatten layer. The first dense layer comprises 256 neurons activated by ReLU, followed by a dropout layer with a dropout rate of 0.5 to prevent overfitting further. The final dense layer contains eight neurons, corresponding to the number of classes in the dataset, and softmax activation is utilized for multi-class classification.
- Optimizer - The RMSprop optimizer is employed to compile the model with a learning rate of 0.001. The loss function, categorical cross-entropy is utilized for optimization and monitored accuracy as the evaluation metric.
- Learning Rate Annealing - To adaptively adjust the learning rate during training, a learning rate reduction strategy is incorporated which is based on the validation accuracy using the ReduceLROnPlateau callback.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        832

 conv2d_1 (Conv2D)           (None, 32, 32, 32)        25632

 max_pooling2d (MaxPooling2D  (None, 16, 16, 32)       0
 )

 dropout (Dropout)           (None, 16, 16, 32)        0

 conv2d_2 (Conv2D)           (None, 16, 16, 64)        18496

 conv2d_3 (Conv2D)           (None, 16, 16, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 8, 8, 64)         0
 2D)

 dropout_1 (Dropout)         (None, 8, 8, 64)          0

 flatten (Flatten)           (None, 4096)              0

 dense (Dense)               (None, 256)               1048832

 dropout_2 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 8)                 2056

=================================================================
Total params: 1,132,776
Trainable params: 1,132,776
Non-trainable params: 0
_________________________________________________________________
```

Fig 6: Proposed CNN Model Architecture

5.3 Feature Extraction & Classification

Furthermore, feature extraction using the CNN layers of the model is also explored. The features from the convolutional neural network (CNN) layers are extracted by excluding the final dense layer. Following this extraction, the Support Vector Machine (SVM) classifiers are leveraged, utilizing various kernels, such as linear, polynomial, and radial basis functions, for the classification task [16]. The validation accuracies of these SVM classifiers were evaluated, with the radial basis function kernel achieving the highest accuracy on the validation set. Hyperparameter tuning using grid search was performed to optimize the SVM classifier's parameters, resulting in selecting the best-performing model. Ensemble methods such as Bagging were also explored to enhance the classification performance further, demonstrating promising results in terms of validation accuracy.

6. RESULT ANALYSIS

6.1 Data Splitting

After loading the dataset of 4000 instances, there are a total of 3200 cases, where the training set is 2880, validation is 320, and testing instances are 800.

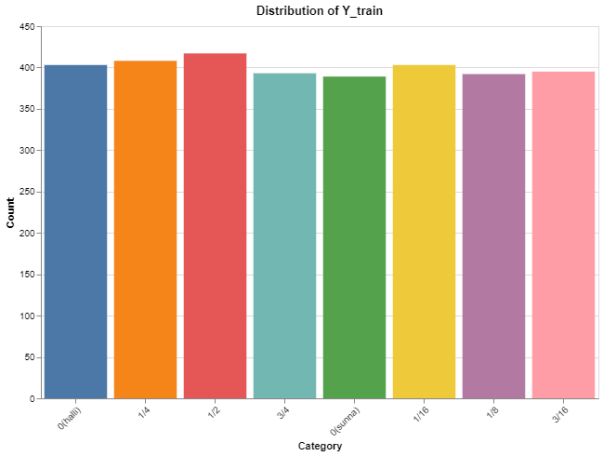

Fig 7: Frequency of labels in the training dataset

These numbers signify the frequency of each label in the dataset. In this case, the distribution is relatively balanced, as the counts of each label are comparable, indicating no significant class imbalance issue in the dataset.

6.2 Experimental Settings

The experimentation phase utilized an HP Pavilion 15-cc1xx Laptop running on an Intel Core i5-8250U processor clocked at 1.60GHz, paired with 8GB of RAM, operating within a Windows 10 environment. The experiments employed Jupyter Notebook, utilizing the dataset previously outlined. The handwritten sets were scanned using the HP LaserJet Pro M1136 Multifunction Printer. The batch cropping of each notation from the user sheet is done using XnConvert, a fast cross-platform batch image converter. The chosen file format was .bmp (Bitmap Image File) throughout the image preprocessing stages. This selection was made due to the inherent characteristics of BMP files, which offer a simple array structure of pixels. Each pixel's color data is directly stored within the file, making it particularly advantageous for handling high-resolution images or those demanding intricate detail. Any additional adjustments or improvements needed were conducted using the GNU Image Manipulation Program (GIMP), an open-source graphics editor renowned for its capabilities in image manipulation and conversion between various image file formats.

6.3 Discussion

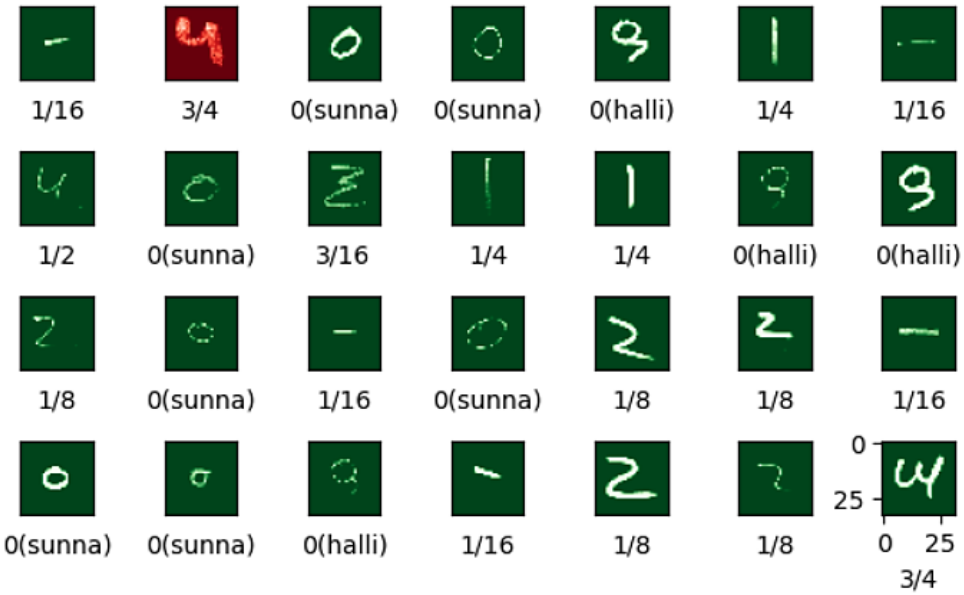

Fig 8: Part of the correctness of predictions is displayed in green, and the errors are in red

Then, the corresponding image with its predicted label is plotted by iterating through each prediction. The color map based on whether the predicted label matches the ground truth label is also determined. Green was used for correct predictions while red was used for incorrect ones, as in Fig. 8.

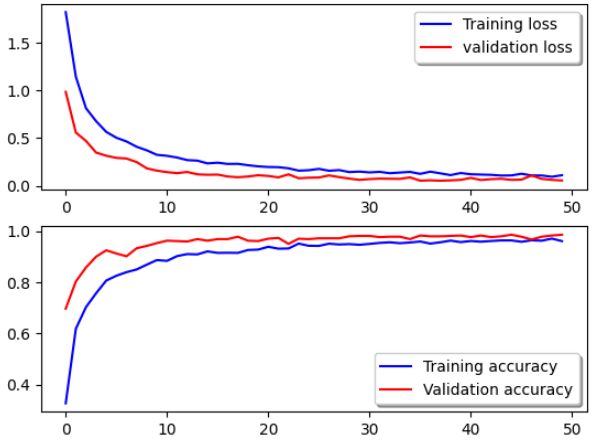

Fig 9: Training & Validation Loss and Accuracy Curves

Initially, both training and validation losses were high, indicating poor model performance. However, as training progressed, both losses decreased significantly, demonstrating the model's ability to learn effectively from the data. Notably, the validation loss remained close to the training loss throughout, signifying successful generalization and avoiding overfitting. This positive trend is further corroborated by the accuracy graphs, where both training and validation accuracy steadily increased. The close alignment between the curves on both graphs reinforces that the model is learning generalizable patterns. Finally, the high final training and validation accuracies solidify the model's performance. These graphs depict a well-trained model with excellent generalization capabilities, suggesting strong performance on unseen data.

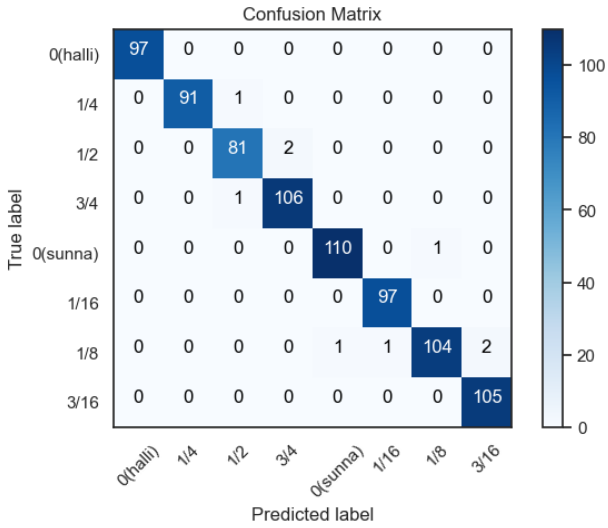

Fig 10: Confusion Matrix for Telugu Fraction Notations

The confusion matrix is a pivotal tool in assessing the efficacy of a classification model within the domain of multi-class classification challenges. Its structure delineates the actual classes via the rows and the predicted classes via the columns. Each diagonal entry signifies the count of accurately classified instances per class. Conversely, the off-diagonal entries depict misclassifications, signifying instances where the model's prediction diverged from the true class label.

The model performs well on specific classes, such as ¹⁄₁₆ and 0('haḷḷi'), where most instances are correctly classified. However, there are some classes where the model struggled, such as ¼ and ½. Class 0 ('sunna') has a relatively high number of correct predictions, but some instances are misclassified as ⅛ and ³⁄₁₆.

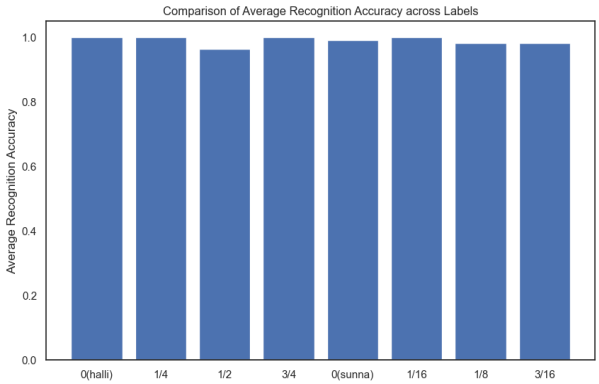

Fig 11: Comparison of Average Accuracy for all labels

From Fig.11, a consistent pattern of accuracy values, with all labels having approximately the same recognition accuracy rate can be observed. This indicates the model has learned robust and generalizable features without exhibiting significant biases or degradation for specific label categories. This could be attributed to good model architecture, effective training strategies, and a diverse dataset.

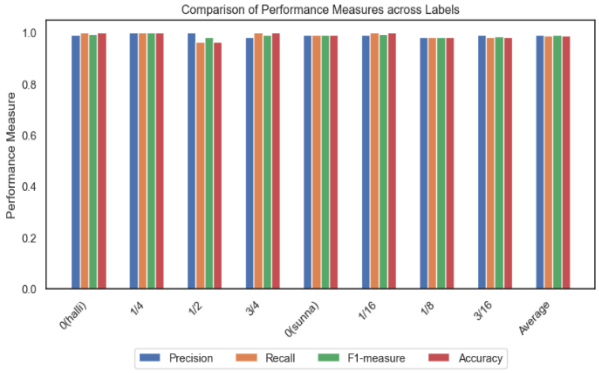

Fig 12: Comparison of Performance Measures for all Telugu Fraction Notations

Fig.12 compares multiple performance measures across labels, including precision, recall, F1-measure, and accuracy. Firstly, the consistent patterns within each performance measure across labels suggest that the model's behavior is relatively stable and does not exhibit significant fluctuations for specific label categories. The precision values appear relatively high across most labels, indicating that the model has a low false positive rate and performs well in minimizing incorrect positive predictions. Additionally, investigating the trade-offs between different performance measures for the intended use case could provide valuable insights into the model's suitability and areas for improvement.

The CNN model achieves an accuracy of 99.671% on the validation set after training for 50 epochs. Additionally, the CNN-SVM hybrid with a radial basis function (RBF) kernel achieved even higher accuracy than the pure CNN model, showcasing the potential of combining deep learning with SVM using different kernels. This paper is a primitive work to be proposed in the field of handwritten digit recognition, as no earlier work on Telugu fraction notations is present. So, in this study, the findings are presented by comparing the achieved results with the current state-of-the-art outcomes in recognizing digits "0 to 9" across various Indian scripts such as Bangla, Kannada, Devanagari, Oriya, and Telugu.

Table 3. Comparison of the overall accuracy of the proposed approach with previous approaches

Trail	Accuracy	State-of-art Results
CNN	99.67%	99.48 [11], 99.37 [8], 99.23 [17], 99.20 [9], 98.6 [13], 96.85 [18], 96.25 [14], 95.47 [19], 94.81 [10], 92.30 [20]
CNN-SVM (Linear)	96.88%	
CNN-SVM (RBF)	99.86%	
CNN-SVM (Poly)	94.19%	
CNN-SVM Bagging	98.75%	

7. CONCLUSION AND FUTURE WORK

This study has presented a robust hybrid approach for recognizing handwritten Telugu fraction notations by combining CNNs and SVMs. By creating a comprehensive dataset and strategic data augmentation, the challenges posed by limited labeled data are addressed and variability in handwriting styles. The tailored CNN architecture effectively learned discriminative features while integrating SVM classifiers with various kernels enhanced classification performance, achieving an accuracy of 99.86% with the RBF kernel. This achievement contributes to regional language text recognition by addressing a previously unexplored area of recognizing “fractions” within handwritten digit recognition of Telugu script.

While these results are promising, several avenues exist for future research. Although limited to Telugu fractions, expanding the dataset size could further improve generalization capabilities, so the dataset must be enlarged through continuous data collection efforts to solidify the model's robustness. Furthermore, exploring the recognition of fractions in conjunction with whole numbers within a sentence would be a valuable addition, which can raise the research on the recognition of mixed numerals and further extend the approach to multiple Indian scripts. Finally, integrating user feedback and usability testing could refine the recognition system for practical applications. This study lays the foundation for further advancements in recognizing handwritten notations in regional Indian languages that could advance language technologies and promote linguistic diversity in the digital age with potential applications in document digitization, cultural preservation, and accessibility for diverse linguistic communities.

8. ACKNOWLEDGMENTS

The authors extend their gratitude to the experts whose contributions were instrumental in developing the dataset and to family and friends for their unwavering support throughout this endeavor.

9. REFERENCES

[1] Charles Philip Brown, 1856 (2006), “The Grammar of the Telugu Language, 2nd Edition,” Chennai: Asian Educational Society, ISBN 81-206-0041-X.

[2] Alexander Duncan Campbell, 1849 (1991), “Grammar of the Teloogoo Language, 3rd Edition,” Chennai: Asian Educational Society, ISBN 81-206-0366-4.

[3] The Unicode Consortium. The Unicode Standard, Version 15.1.0, (South San Francisco, CA: The Unicode Consortium, 2023, ISBN 978-1-936213-33-7)

[4] Charles Philip Brown, Revised by M Venkata Ratnam, W H Campbell, Kandukuri Viresalingam. 1903 (2004), "Telugu-English Dictionary = Nighantuvu Telugu - Inglish, 2nd Edition," Chennai: Asian Educational Society, ISBN 81-206-0037-1.

[5] Puduru Seetarama Sastry, 1847 (1916), "Peddabālaśiksha," Cennapuri: Vavilla Ramaswami Sastrulu and Sons, Chennai.

[6] Nāgārjuna Venna. "Telugu Measures and Arithmetic Marks," JTC1/SC2/WG2 N3156, International Organization for Standardization.

[7] U. Bhattacharya and B. B. Chaudhuri, "Handwritten Numeral Databases of Indian Scripts and Multistage Recognition of Mixed Numerals," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 3, pp. 444–457, March 2009, doi: 10.1109/TPAMI.2008.88.

[8] U. Pal, N. Sharma, T. Wakabayashi and F. Kimura, "Handwritten Numeral Recognition of Six Popular Indian Scripts," Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brazil, 2007, pp. 749–753, doi: 10.1109/ICDAR.2007.4377015.

[9] Nibaran Das, Jagan Mohan Reddy, Ram Sarkar, Subhadip Basu, Mahantapas Kundu, Mita Nasipuri, Dipak Kumar Basu, "A statistical–topological feature combination for recognition of handwritten numerals," Applied Soft Computing, Volume 12, Issue 8, 2012, pp. 2486-2495, doi: 10.1016/j.asoc.2012.03.039.

[10] K. Roy, T. Pal, U. Pal, and F. Kimura, "Oriya handwritten numeral recognition system," Eighth International Conference on Document Analysis and Recognition (ICDAR'05), Seoul, Korea (South), 2005, pp. 770-774 Vol. 2, doi: 10.1109/ICDAR.2005.183.

[11] Shailedra Kumar Shrivastava, Sanjay S. Gharde, "Support Vector Machine for Handwritten Devanagari Numeral Recognition," International Journal of Computer Applications. 7, 11 (October 2010), 9–14, doi: 10.5120/1293-1769.

[12] B.V.Dhandra, R.G.Benne, Mallikarjun Hangarge, "Kannada, Telugu and Devanagari Handwritten Numeral Recognition with Probabilistic Neural Network: A Novel Approach," Recent Trends in Image Processing and Pattern Recognition. RTIPPR, 2 (None 2010), 83-88.

[13] Stuti Asthana, Farha Haneef, and Rakesh K. Bhujade, "Handwritten multiscript numeral recognition using artificial neural networks," International Journal of Soft Computing and Engineering, 1.1 (2011), 1–5.

[14] Adarsh Trivedi, Siddhant Srivastava, Apoorva Mishra, Anupam Shukla, and Ritu Tiwari, "Hybrid evolutionary approach for Devanagari handwritten numeral recognition using Convolutional Neural Network," Procedia Computer Science, 125 (2018): 525-532.

[15] Nobuyuki Otsu (1979), "A threshold selection method from gray-level histograms," IEEE Trans. Sys., Man., Cyber. 9 (1): 62–66, doi:10.1109/TSMC.1979.4310076.

[16] S. K. Manocha and P. Tewari, "Devanagari Handwritten Character Recognition using CNN as Feature Extractor," 2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Pune, India, 2021, pp. 1-5, doi: 10.1109/SMARTGENCON51891.2021.9645786.

[17] MSLB. Subrahmanyam, V. Vijaya Kumar, B. Eswara Reddy, " A Robust Zonal Fractal Dimension Method for the Recognition of Handwritten Telugu Digits," International Journal of Image, Graphics and Signal Processing (IJIGSP), Vol.10, No.9, pp. 42–55, 2018. doi: 10.5815/ijigsp.2018.09.06.

[18] S. V. Rajashekararadhya and V. P. Ranjan, "Zone-based hybrid feature extraction algorithm for handwritten numeral recognition of four Indian scripts," 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 2009, pp. 5145–5150, doi: 10.1109/ICSMC.2009.5346007.

[19] Jyothi, J., Manjusha, K., Anand Kumar, M., & Soman, K. P. (2015). Innovative Feature Sets for Machine Learning based Telugu Character Recognition. Indian Journal of Science and Technology, 8(24), doi: 10.17485/ijst/2015/v8i24/116917.

[20] R. S. Kunte and S. Samuel, "Script Independent Handwritten Numeral Recognition," IET International Conference on Visual Information Engineering, Bangalore, 2006, pp. 94–98.