ISSN: 2584-0495 Vol. 1, Issue 7, pp. 460- 468

International Journal of Microsystems and IoT

ISSN:(Online) Journal homepage: https://www.ijmit.org

# Optical Character Recognition and Analysis of Tamil Characters

**Sujith M, Akash Raj ST, Yeshwanth B, Priyadharshini J, and Jayasudha M**

**Cite as:** Sujith, M., Akash Raj, S., Yeshwanth, B., Priyadharshini, J., & Jayasudha, M. (2023). Optical Character Recognition and Analysis of Tamil Characters. International Journal of Microsystems and IoT, 1(7), 460–468. https://doi.org/10.5281/zenodo.10609826



Published online: 18 December 2023.



**DOI:** https://doi.org/10.5281/zenodo.10609826

International Journal of Microsystems and IoT
Vol. 1, Issue 7, pp. 460- 468; DOI: https://doi.org/10.5281/zenodo.10609826

# Optical Character Recognition and Analysis of Tamil Characters

Sujith M, Akash Raj ST, Yeshwanth B, Priyadharshini J and Jayasudha M

School of Computer Science and Engineering, Vellore Institute of Technology Chennai, Chennai, India

**ABSTRACT**

Several tools are available for the analysis of the English language. However, not many tools are available for the analysis of Tamil. Even though most of the applications and programming languages across the internet have incorporated the use of foreign languages in them, the ease of usage is not up to the mark. Popular programming languages like Python and Java have the capability of processing Tamil, however, they do not follow the rules of the language and their respective language classes are not up to the mark. For e.g., கு is considered as 2 separate characters 'க' + 'ு'. This creates an unnecessary overhead in the processing of the language. We also find the absence of an open-source library that can process Tamil and perform its analysis similarly to the NLTK library. The aim of the project is to build an open-source and user-friendly Python library for the analysis of the Tamil corpus in the correct form. A separate class is created for the language to overcome the issues of the inbuilt Python class that handles Tamil alphabets. Several functions must be provided to the users for processing any given Tamil text. We perform the optical character recognition of printed Tamil characters and process them further to analyze the characters detected.

**KEYWORDS**

Character Analysis; Corpus Analysis; Library; OCR; Package; Python; Tamil Characters.

## 1. INTRODUCTION

Optical character recognition is the recognition of the characters that are printed on images. It follows the detection of the characters printed followed by the recognition of the character. This technique is widely used in various fields and applications. Google Lens is an application that does this job in an efficient way. Several tools and packages have been developed for this purpose. However, these kinds of tools are not that good with languages other than English. It is difficult to find tools that perform these jobs for regional languages like Tamil, Telugu, and Hindi. Python libraries are used for serving various functionalities for various reasons. OCR is also an important function that Python libraries can perform. An OCR library can be developed solely for the Tamil language, and it can also have functions to perform the corpus analysis of the language. OCR has several layers in its processes like line extraction, word extraction, and character extraction followed by recognition.

Tamil is a Dravidian language that is spoken in the state of Tamil Nadu in India. It is a very old language that is still in use. About 88.6 million people in the world speak and read the language. Out of this about 25 million and 12 million people read the top two newspapers that are printed in the state. There is a huge amount of Tamil-printed image data being generated every day. Currently, about 50 renowned Tamil newspapers are operating and generating news every day. The Tamil news corpus is so big and if trained properly, can provide an OCR model for the language. The development of a tool to process such a language has a very high scope.

A corpus analysis of a language simply means the analysis of the structure, syntax, and grammar of a language. In this project, we analyse the corpus on the character level. Once after the recognition of a character, we find the class of the character. We have four main classes in Tamil namely uyir, mei, uyir-mei and ayutha ezhuthu. This project also solves several other problems like the processing of Tamil characters in the Python programming language. Python does not process Tamil as it is supposed to. For eg: in Tamil, the letters 'க்', 'ங்' etc. are mei eluthukkal or consonants. However, python takes the letters 'க', 'ங' as the base and treats 'க்', 'ங்' as the combination of 'க' + '்' and 'ங' + '்' respectively while it is supposed to be the other way around. Similarly for all other compounded consonants. Due to this, the length of the characters is not counted properly. ['க்', 'கா', 'கி', 'கே'] are all considered as two characters while only 'க' is a character of length one. So, we override the length function in Python so that it can process Tamil words.

Transliteration is the process of changing words from one language to another such that they sound the same. It is different from translation as in translation the meaning of the sentences does not change. Transliterated words do not give any actual meaning, but they are the combination of letters of another language to produce the same sound. For e.g.: the word 'கடல்' is translated as 'sea' and transliterated as 'kadal' in English. In recent times, with the growth of social media, transliteration tends to play a major role in natural language processing. This is because most of the time, people tend to chat in their regional language by typing the letters in English. So, transliteration of the text would play a vital role before the actual processing could be done. We have included a function that can transliterate Tamil words into English words. The Tamil corpus that we use to identify the root words is in English.

## 2. RELATED WORKS

A paper uses various deep-learning techniques to recognize printed or handwritten Urdu text and produces an Urdu corpus that can be used for further research and development [1]. It has produced the largest-ever Urdu corpus available that consists of over 6,02,472 images, including text-line and word images in three prominent fonts. Three-level tokenization is performed with the text being the root and characters being the leaves. Works well for texts with mixed fonts. This concludes that the model is generalized for different fonts. The size of the corpus is still not sufficient to build a great model. The corpus size could be further expanded. The model is generalized for three different Urdu fonts, it could be further expanded. The background and foreground colors in the dataset are not different. So, the performance may be affected by different colors. We need to perform data augmentation to generate a large and sufficient database for the training and validation of the model. We need to ensure that the images in our dataset have a wide range of fonts and colors so that the model can perform well in various conditions. This paper proposes a conceptual learning approach for the recognition and classification of Chinese characters [2]. It is the first paper to propose a conceptual approach to Chinese character recognition. The method obtains better performance than state-of-the-art methods. The method does not overperform other methods for all given datasets. Only a specific kind of data set like the training set is suitable for detection. Conceptually even though it has many advantages, it is better to use deep learning algorithms to obtain good results across a wide variety of data. Another paper aims to create a new benchmark for optical character recognition in Tamil by generating a huge dataset and analyzing the use case of the database by applying several machine learning models [3]. The dataset was obtained from about 850 Tamil people which includes school students, homemakers, university students, and faculty. Many machine learning and deep learning models like CNN, KNN, SVM, random forest etc. are used to analyze the data. Only the basic models are used for analyzing the data which is insufficient and creates a need for resembling. We need to perform ensemble learning to combine the power of multiple machine learning and deep learning algorithms. The accuracy of our final model could be improved through the ensemble techniques. A paper introduces a framework for optical character recognition for mobile or embedded systems. It uses a two-step framework that treats character separation as language-independent and classification as language-dependent modules. The proposed method surpasses the algorithmic method implemented in Tesseract OCR. The method primarily aims at the segmentation part of OCR and not at the classification. The independent segmentation approach can be used along with a properly trained and efficient model to achieve a good result in the end [4]. This paper proposes a two-stage strategy for the recognition and classification of braille characters. It uses a lightweight CNN with a faster processing speed. Performs better than the state-of-the-art models with good accuracy and efficiency. The application also provides an accessible user interface. Works well only with fewer datasets. A high-quality image is required to produce good results. The dataset must include low-quality images with different extensions to produce better results [5]. Effective image denoising and recognition for noisy handwritten characters involves using advanced machine,

learning techniques, such as deep neural networks, to pre-process and analyze the image data, resulting in improved accuracy and reduced noise in the final output. Improved accuracy: By removing noise from handwritten character images and accurately recognizing the characters, the overall accuracy of automated systems that rely on character recognition can be significantly improved. Reduced errors: Image denoising can significantly reduce the number of errors caused by noise in handwritten character images resulting in more accurate results Increased efficiency: When the accuracy of recognition of characters is improved the efficiency is increased and this leads to increased throughput. Improved data quality: This can improve the quality of the data used in applications such as handwriting recognition. It helps to improve decision-making and gives better outcomes. Effective noise removal from images and recognition of noisy handwritten characters can lead to more accurate and reliable automated systems that can improve efficiency, data quality, and accessibility. Computational complexity: It can be computationally expensive, particularly when dealing with large datasets or complex image features. Takes a longer time to process the data Limited accuracy: there are still limitations to their accuracy, particularly when dealing with very noisy or low-quality images. Difficulty with certain character types: Certain types of characters, such as cursive or stylized handwriting, can be more difficult to recognize accurately, particularly when dealing with noisy or low-quality images. Sensitivity to changes in image quality: Image denoising and recognition techniques can be sensitive to changes in image quality, particularly changes in lighting, contrast, or resolution. This can impact the accuracy of recognition and require additional preprocessing steps. One solution is to use data augmentation techniques to increase the diversity of the training dataset. This can involve adding noise, varying the lighting conditions, or changing the orientation of the characters in the images. A combination of image denoising techniques and machine learning algorithms can also be used to improve recognition accuracy Preprocessing techniques such as binarization, thinning, and skeletonization can be employed to improve the quality of the images and reduce noise before applying the recognition algorithms [6]. A residual handwritten Chinese text recognition based on CNN is a state-of-the-art technique that utilizes residual connections and attention mechanisms to achieve high accuracy in recognizing offline handwritten Chinese text. High accuracy: The proposed model achieved state-of-the-art results in recognizing handwritten Chinese text, indicating that the residual attention mechanisms and FCN architecture are effective in improving recognition accuracy [7]. The literature study explores the field of Urdu optical character recognition (OCR), highlighting its newness and wide research potential. It covers OCR system components, emphasizing challenges in recognizing Urdu script, especially ligatures. The paper identifies future research problems, including the need for standardized datasets, labeling challenges, cursive text segmentation, and optimal classification algorithms. It suggests solutions such as automated labeling, segmentation-free systems, and experimentation with traditional and deep learning methods to enhance recognition in Urdu Nastalique calligraphic style [8]. Another paper proposes a handwritten character recognition system for printed Tamil images, implementing Optical Character Recognition (OCR) in the initial stage. The project addresses challenges like poor print quality and unknown font faces and recognizes handwritten text inaccurately. Utilizing a Convolutional Neural Network (CNN) model for handwritten digit recognition, the system exhibits the potential to robustly recognize Tamil characters without additional feature collection. Experimental results indicate comparable accuracy with existing methods, achieving 98.00% accuracy for handwritten Tamil character recognition on a large dataset, demonstrating the model's robustness [9]. This paper introduces a novel Text Character Recognition (TCR) approach involving two main processes: pre-processing and recognition. The pre-processing stage comprises RGB to grayscale conversion, binarization with thresholding, image complementation, morphological operations, and linearization. Recognition is performed using a Convolutional Neural Network (CNN) with a unique approach - the fully connected layer and weights are fine-tuned by an enhanced version of standard LA, called SALA. The proposed model (CNN + SALA) outperforms other state-of-the-art models, including RNN, LA, EHO-NN, and DCNN, with an accuracy of ~0.84 at TD = 50, demonstrating improvements of 12.7%, 3.4%, 7.2%, and 11.2%, respectively [10]. This paper presents a novel approach, the Hierarchical Graph Transformer, for large-scale multi-label text classification. The method involves modeling text as a graph structure to capture diverse semantics and connections. A multi-layer transformer structure, incorporating a multi-head attention mechanism at the word, sentence, and graph levels, is introduced to comprehensively capture textual features. Hierarchical tag relationships are explored, utilizing label hierarchy to generate representations, and designing a weighted loss function based on semantic label distance. Experimental results on three benchmark datasets show that the proposed model effectively captures text hierarchy, logic, and label relationships [11]. This paper introduces a convolutional feature fusion method for Urdu character recognition in natural scenes. The approach incorporates multi-scale feature aggregation and multi-level feature fusion networks. The multi-scale feature aggregation network combines low-level and mid-level convolutional features from different layers, followed by integration with high-level features in the multi-level feature fusion network. The fused features are then processed by a SoftMax classifier for predictions. The proposed network is evaluated on three datasets, including a novel Urdu character dataset for natural scene images. Real-time data augmentation enhances training set samples. Experimental results demonstrate superior performance on the new Urdu character dataset and Chars74K, with competitive results on ICDAR03. The paper outlines future directions, including extending the method to recognize cropped words and developing an end-to-end system for text recognition [12]-[16]. Another paper addresses the challenging task of Arabic offline OCR for printed text by developing a model based on a combination of the FKNN classifier and Genetic Algorithm (GA) [17]. The model initially uses a fourteen-feature dataset, which is reduced to six features through GA. The FKNN, known for its speed and simplicity, is then employed for classification. The model leverages the [18] fuzzification process of FKNN to handle the cursive nature of Arabic characters, achieving high recognition accuracy of 98.69% for various samples in a short time. The proposed model focuses on minimizing recognition errors, reducing running time, and maintaining a simple structure. Notably, it effectively addresses Arabic word overlapping using a fuzzy classifier [19]-[22]. Future work aims to utilize more complex Arabic fonts datasets, especially Diwani font, and address diacritics problems, while also extending the model to handle Arabic handwritten words [23].

## 3. PROPOSED WORK

We have created a Python library implementing the work of character count and classification to analyze the characteristics of the language. It counts the number of characters in a paragraph/ statement and segregates them into six different classes namely uyir (vowels) (அ, ஆ, இ, ஈ, உ, ஊ, எ, ஏ, ஐ, ஒ, ஓ, ஔ), ayutha ezhuthu(ஃ), mei(consonants) (க, ங, ச, ஞ, ட, ண, த, ந, ப, ம, ய, ர, ல, வ, ழ, ள, ற, ன), uyir-mei (a compounded form of vowels and consonants) (refer table 1.1). The mei(consonants) are further sub-classified into 3 types namely vallinam (க, ச, ட, த, ப, ற), mellinam(ங, ஞ, ண, ந, ம, ன) and idayinam(ய, ர, ல, வ, ழ, ள).

### 3.1 Techniques Used

The techniques used are pre-processing techniques to filter the noise in the image with the help of an open cv library in python. This[24]-[27] is a need to improve the accuracy of detecting the images. Line extraction breaks down a huge paragraph into separate sentences. Word tokenization to break down the sentences into words. Character extraction to break down words into their characters. These characters are then fed to the classifier model. A character recognition model is manually trained on the 124 classes of images to identify each character. Using the Convolution Neural Networks (CNN) [28] model the classification is done into 4 different classes namely, Uyir ezhuthukal, Mei ezhuthukal, Uyir-mei ezhuthukal and Ayutha ezhuthu. Transliteration is done to convert the Tamil words into English with the same pronunciation. Word pronunciation is also included to pronounce the detected words as audio.

**Table. 1** Compound form of Vowels and Consonants

| consonants | Vowels | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | அ | ஆ | இ | ஈ | உ | ஊ | எ | ஏ | ஐ | ஒ | ஓ | ஔ |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| க் | க | கா | கி | கீ | கு | கூ | கெ | கே | கை | கொ | கோ | கௌ |
| ங் | ங | ஙா | ஙி | ஙீ | ஙு | ஙூ | ஙெ | ஙே | ஙை | ஙொ | ஙோ | ஙௌ |
| ச் | ச | சா | சி | சீ | சு | சூ | செ | சே | சை | சொ | சோ | சௌ |
| ஞ் | ஞ | ஞா | ஞி | ஞீ | ஞு | ஞூ | ஞெ | ஞே | ஞை | ஞொ | ஞோ | ஞௌ |
| ட் | ட | டா | டி | டீ | டு | டூ | டெ | டே | டை | டொ | டோ | டௌ |
| ண் | ண | ணா | ணி | ணீ | ணு | ணூ | ணெ | ணே | ணை | ணொ | ணோ | ணௌ |

## 3.2 Flow Diagram

Figure 1 shows the basic flow of the functions in the Python library.

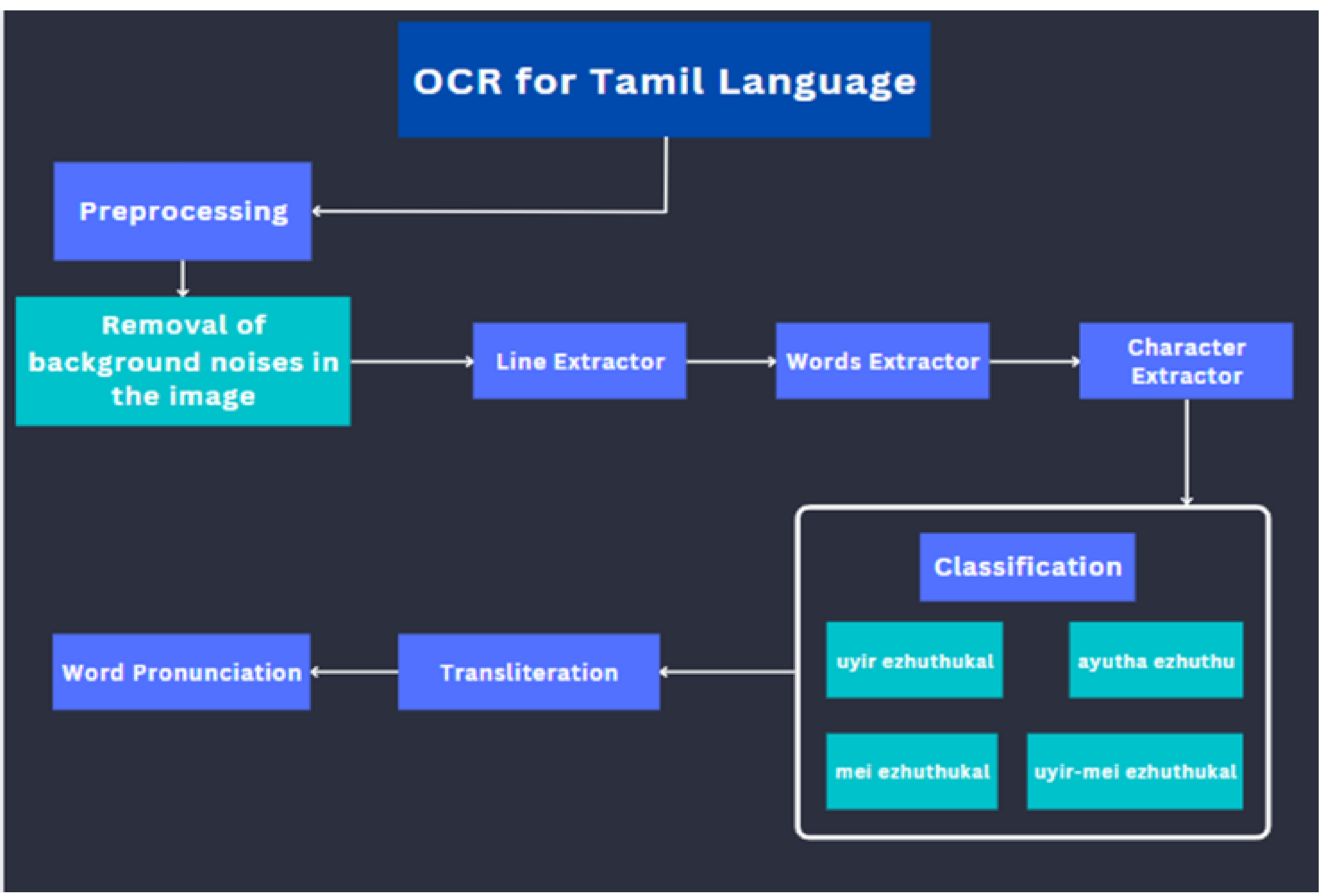


Fig. 1 Flow Diagram of the Proposed Work.

## 4. MODEL OUTPUT

An image (refer to figure 2) with a paragraph printed in Tamil was fed as input to the model to obtain the results.

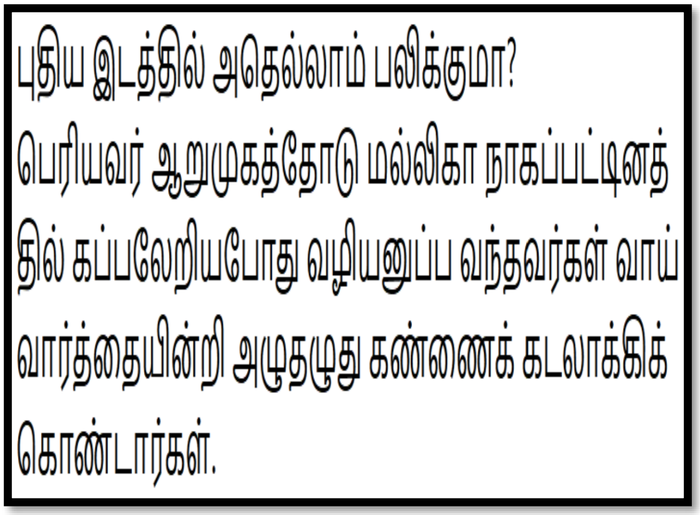

புதிய இடத்தில் அதெல்லாம் பலிக்குமா?
பெரியவர் ஆறுமுகத்தோடு மல்லிகா நாகப்பட்டினத்
தில் கப்பலேறியபோது வழியனுப்ப வந்தவர்கள் வாய்
வார்த்தையின்றி அழுதழுது கண்ணைக் கடலாக்கிக்
கொண்டார்கள்.

Fig. 2 Character Printed Image.

Figure 3 shows an example of the contour lines identified in a word extracted from the image.

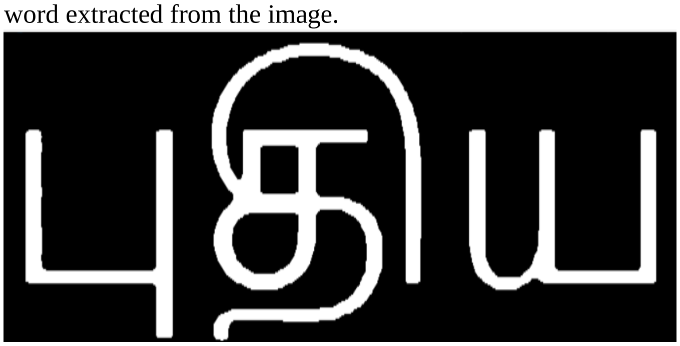



Fig. 3 Contours in a word image

Figure 4 shows us the Tamil characters that were detected and transliterates the same into English to provide the same sound while pronouncing the words.

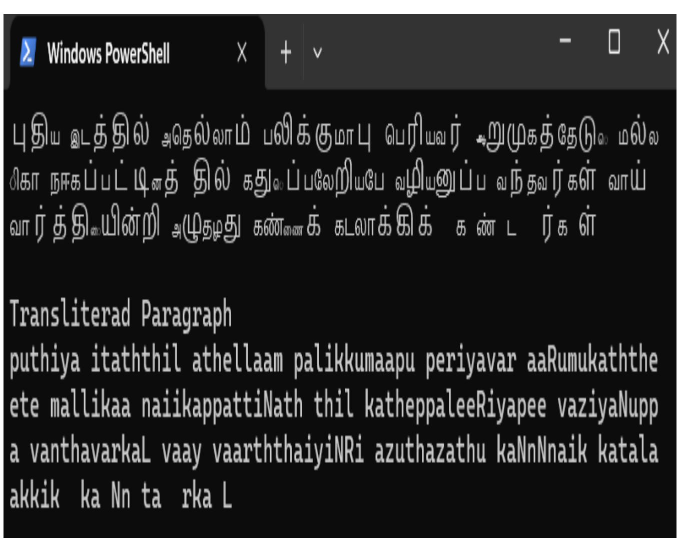
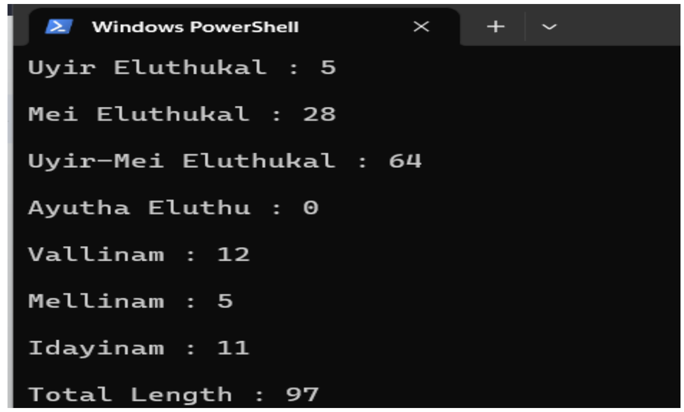



Fig. 4 Detected and Transliterated text

As shown in Figure 5 the characters recognized are classified into their respective classes and a count of the number of characters in each class is displayed.


Windows PowerShell

Uyir Eluthukal : 5
Mei Eluthukal : 28
Uyir-Mei Eluthukal : 64
Ayutha Eluthu : 0
Vallinam : 12
Mellinam : 5
Idayinam : 11
Total Length : 97

Fig. 5 Character classification and counts

## 5. RESULTS AND DISCUSSIONS

We have made a deep learning model that recognizes Tamil characters. The model is trained with a dataset consisting of 62496 images belonging to 124 classes.

The model is compiled with four convolutional and max pooling layers were added. The SoftMax activation function is used to obtain the probability for all the classes. The model was compiled and trained to run for 20 epochs in total. We have done software testing on the given dataset to clearly record the accuracy changes of the model based on the changes in the dataset.

The accuracy comparison with each epoch is shown in Figure 6 between the proposed and the existing models.

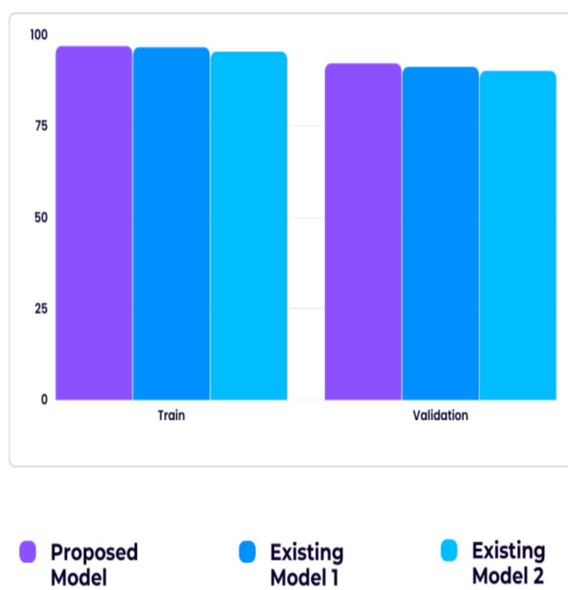


Fig. 6 Accuracy of model

Another important parameter that helps us in giving the efficiency of the model is the loss function. Unlike accuracy, losses of a model may not continuously increase or decrease, it could be unpredictable at times. Figure 7 shows the change in losses with increasing epochs. We can see that the loss gets reduced and then it again increases at a certain point.


Fig. 7 Loss of model

The training time taken for the tensor flow model is given in Table 2. As seen, it has taken about 4.75 hours for the training to be completed.

Table. 2 **Training** time of model

| **Epochs** | **Time (in seconds)** |
|---|---|
| **1** | 1300 |
| **4** | 4938 |
| **8** | 9502 |
| **12** | 13878 |
| **15** | 17129 |

Execution time of the functions in the library is indeed quite high. We would have to pass each function one after the other to achieve a higher result. The execution time of running several image files as input is given in Table 3.

Table. 3 Execution time of model

| **File Size (in lines)** | **Execution Time (in seconds)** |
|---|---|
| 3 | 6 |
| 5 | 9 |
| 8 | 13 |
| 12 | 17 |

Numerous Python libraries are used for several purposes, however, not many functions are provided in a single library for a specific purpose. We do provide many other extra functions as well apart from the OCR [29]-[31] detection like transliteration, pronunciation using an inbuilt library and length validation. Here we have a comparison of the average number of functions provided by several tools shown in Figure 8.

We can see that it is usually the online libraries that provide more functions. So, we have about seven functions in our library at present and it can be further expanded to perform further analysis works like morphology, POS tagging etc.

Fig 8 gives us a comparison of the average number of functions used in different types of applications and the number of functions in our model. The number of functions in our library has already surpassed the average numbers in websites and applications. This can be further expanded to achieve the average number of functions in a library.

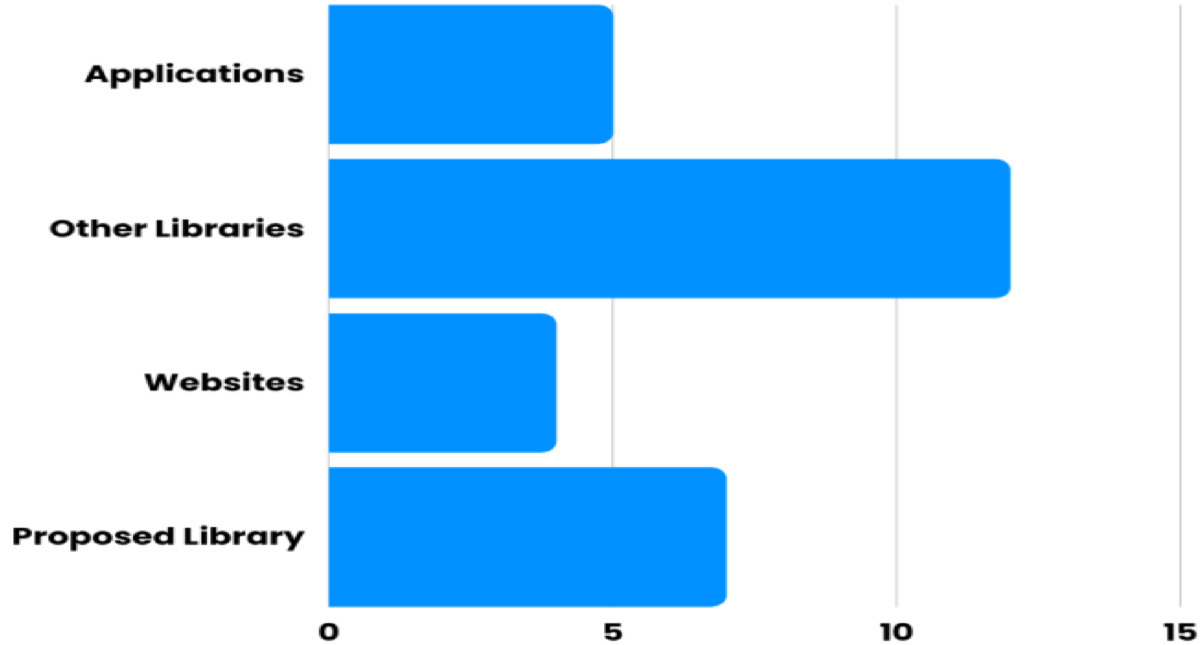



Fig. 8 Average number of functions

The lines, word and character extractor functions are the most important functions that occur even before the actual recognition takes place. The accuracy and execution time for those functions are given in Table 4.

Table. 4 Accuracy and Execution time of 3 functions

| Function | Accuracy | Execution Time |
|---|---|---|
| Line Extractor | 92 | 0.534 |
| Word Extractor | 79 | 0.832 |
| Character Extractor | 68 | 1.327 |

The dataset we selected is rich in various foreground and background colours that increase the efficiency and accuracy of detecting the characters in all possible scenarios. As we make use of the open-cv library, we are also able to detect the contours of different fonts up to a decent level of accuracy. Figure 9 shows the comparison of our model with existing models in terms of the average number of colours and fonts identified.

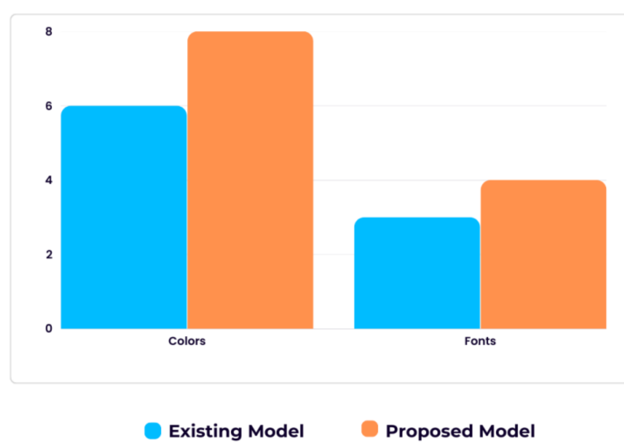



Fig. 9 Number of colours and fonts identified.

CRR is a metric that measures insertion, deletion and substitution errors that tend to occur at the character and word levels. It is a standard evaluation method that is used for assessing the accuracy of an OCR model. WRR equals the total number of words that were correctly identified divided by the number of words. Table 5 shows the values for the same for our model compared with some existing models.

**Table. 5** CRR, WRR and Execution time of 3 Models

| Model | WRR | CRR | Execution Time (seconds) |
|---|---|---|---|
| RNN LSTM | 95.37 | 98 | 128.63 |
| RNN GRU | 95.28 | 98 | 118 |
| CNN | 98 | 99 | 120 |

## 6. CONCLUSION

We have successfully created a Python library that can detect the Tamil characters in an image and perform the respective detection, recognition, and analysis of characters. We have achieved an accuracy of 94.38% with our recognition model along with precision and recall of 100 and 99.83 per cent respectively. Even though various libraries exist for the OCR detection of characters, this is the first library to be created solely for Tamil which is the novelty of this research paper. Several functions have been integrated into this library and in future the scope of expanding this library will be high. We can still make improvements to the accuracy of this model and make it better and can also add more corpus analysis functions to make it a better and complete package that can handle all the works of Tamil with a few lines of code. This would highly benefit the users planning to perform corpus analysis like morphological analysis, summarization etc. with Tamil text. The fact that they can give the images with printed text as input is an added advantage.

## REFERENCES


1. T. Nasir, M. K. Malik and K. Shahzad. (2021). MMU-OCR-21: Towards End-to-End Urdu Text Recognition Using Deep Learning. IEEE Access, vol. 9, pp. 124945-124962. http://doi.org//10.1109/ACCESS.2021.3110787

2. L. Xu, Y. Wang, X. Li and M. Pan. (2019). Recognition of Handwritten Chinese Characters Based on Concept Learning. IEEE Access, vol.7, pp. 102039-102053. http://doi.org//10.1109/ACCESS.2019.2930799

3. N. Shaffi and F. Hajamohideen. (2021). uTHCD: A New Benchmarking for Tamil Handwritten OCR. IEEE Access, vol. 9, pp. 101469-101493. http://doi.org//10.1109/ACCESS.2021.3096823

4. Y. S. Chernyshova, A. V. Sheshkus and V. V. Arlazarov. (2020). Two-Step CNN Framework for Text Line Recognition in Camera-Captured Images. IEEE Access, vol. 8, pp. 32587-32600. http://doi.org/10.1109/ACCESS.2020.2974051

5. T. Kausar, S. Manzoor, A. Kausar, Y. Lu, M. Wasif and M. A. Ashraf. (2021). Deep Learning Strategy for Braille Character Recognition. IEEE Access, vol. 9, pp. 169357-169371. http://doi.org//10.1109/ACCESS.2021.3138240

6. S. Gonwirat and O. Surinta. (2022). DeblurGAN-CNN: Effective Image Denoising and Recognition for Noisy Handwritten Characters. IEEE Access, vol. 10, pp. 90133-90148. http://doi.org//10.1109/ACCESS.2022.3201560

7. Y. Wang, Y. Yang, W. Ding and S. Li. (2021). A Residual-Attention Offline Handwritten Chinese Text Recognition Based on Fully Convolutional Neural Networks. IEEE Access, vol. 9, pp. 132301-132310. http://doi.org//10.1109/ACCESS.2021.3115606

8. N. H. Khan and A. Adnan. (2018). Urdu Optical Character Recognition Systems: Present Contributions and Future Directions. IEEE Access, vol. 6, pp. 46019-46046. http://doi.org//10.1109/ACCESS.2018.2865532

9. M. Chithambarathanu and D. Ganesh. (2021). Character Recognition Tamil Language in Printed Images using Convolutional Neural Network (CNN) analysis. 4th International Conference on Computing and Communications Technologies (ICCCT), Chennai, India, 2021, pp. 1-6. http://doi.org//10.1109/ICCCT53315.2021.9711893

10. R.B. Lincy, R. Gayathri. (2021). Optimally configured convolutional neural network for Tamil Handwritten Character Recognition by improved lion optimization model. Multimed Tools Appl 80, 5917–5943. https://doi.org//10.1007/s11042-020-09771-z

11. J. Gong. (2020). Hierarchical Graph Transformer-Based Deep Learning Model for Large-Scale Multi-Label Text Classification. IEEE Access, vol. 8, pp. 30885-30896. http://doi.org//10.1109/ACCESS.2020.2972751

12. A. A. Chandio, M. Asikuzzaman and M. R. Pickering. (2020). Cursive Character Recognition in Natural Scene Images Using a Multilevel Convolutional Neural Network Fusion. IEEE Access, vol. 8, pp. 109054-109070. http://doi.org//10.1109/ACCESS.2020.3001605

13. A. M. P. Ligsay, J. B. Rivera and J. F. Villaverde. (2022). Optical Character Recognition of Baybayin Writing System using YOLOv3 Algorithm. IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), Kota Kinabalu, Malaysia, 2022, pp. 1-5. https://doi.org/10.1109/IICAIET55139.2022.9936792

14. S. M. Darwish and K. O. Elzoghaly. (2020). An Enhanced Offline Printed Arabic OCR Model Based on Bio-Inspired Fuzzy Classifier. IEEE Access, vol. 8, pp. 117770-117781. http://doi.org//10.1109/ACCESS.2020.3004286

15. I. Anuradha, C. Liyanage, H. Wijayawardhana and R. Weerasinghe. (2020). Deep Learning Based Sinhala Optical Character Recognition (OCR). 20th International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, Sri Lanka, 2020, pp. 298-299. https://doi.org/10.1109/ICTer51097.2020.9325428

16. R. Pramanik, and S. Bag. (2020). Segmentation-based recognition system for handwritten bangla and devanagari words using conventional classification and transfer learning. IET Image Processing, 14(5), 959-972. https://doi.org/10.1049/iet-ipr.2019.0208

17. S. Ukil, S. Ghosh, S.M. Obaidullah. (2020). Improved word-level handwritten Indic script identification by integrating small convolutional neural networks. Neural Comput & Applic 32, 2829–2844. https://doi.org/10.1007/s00521-019-04111-1

18. S. Kundu, S. Paul, P.K. Singh. (2020). Understanding NFC-Net: a deep learning approach to word-level handwritten Indic script recognition. Neural Comput & Applic 32, 7879–7895. https://doi.org/10.1007/s00521-019-04235-4

19. A. Jindal, R. Ghosh. (2023). A hybrid deep learning model to recognize handwritten characters in ancient documents in Devanagari and Maithili scripts. Multimed Tools Appl. https://doi.org/10.1007/s11042-023-15826-8

20. R. Ghosh. (2023). Newspaper text recognition in Bengali script using support vector machine. Multimed Tools Appl. https://doi.org/10.1007/s11042-023-16862-0

21. V. Jayanthi, S. Thenmalar. (2023). Recognition of Tamil handwritten text from a digital writing pad using MWDCNN. Multimed Tools Appl. https://doi.org/10.1007/s11042-023-16878-6

22. N. Pughazendi, M. HariKrishnan, R. Khilar. (2023). Optical handwritten character recognition for Tamil language using CNN-VGG-16 model with RF classifier. Opt Quant Electron 55, 976. https://doi.org/10.1007/s11082-023-05211-y

23. K. Limbachiya, A. Sharma, P. Thakkar. (2022). Identification of handwritten Gujarati alphanumeric script by integrating transfer learning and convolutional neural networks. Sādhanā 47, 102. https://doi.org/10.1007/s12046-022-01864-9

24. D. Sudarsan, D. Sankar. (2023). Development of an effective character segmentation and efficient feature extraction technique for malayalam character recognition from palm leaf manuscripts. Sādhanā 48, 156. https://doi.org/10.1007/s12046-023-02181-5

25. H.T. Basavaraju, V.N.M. Aradhya, M.S. Pavithra. (2021). Arbitrary oriented multilingual text detection and segmentation using level set and Gaussian mixture model. Evol. Intel. 14, 881–894. https://doi.org/10.1007/s12065-020-00472-y

26. V.C. Hallur, R.S. Hegadi. (2020). Handwritten Kannada numerals recognition using deep learning convolution neural network (DCNN) classifier. CSIT 8, 295–309. https://doi.org/10.1007/s40012-020-00273-9

27. F.J.J. Joseph. (2020). Effect of supervised learning methodologies in offline handwritten Thai character recognition. Int. j. inf. tecnol. 12, 57–64. https://doi.org/10.1007/s41870-019-00366-y

28. A. Jindal, R. Ghosh. (2023). An optimized CNN system to recognize handwritten characters in ancient documents in Grantha script. Int. j. inf. tecnol. 15, 1975–1983. https://doi.org/10.1007/s41870-023-01247-1

29. C. S. Lwin and W. Xiangqian. (2020). Myanmar Handwritten Character Recognition from Similar Character Groups Using K-Means and Convolutional Neural Network. IEEE 3rd International Conference on Electronics and Communication Engineering (ICECE), Xi'An, China, 2020, pp. 172-176. https://doi.org/10.1109/ICECE51594.2020.9353028

30. P. Latha, V. Sumitra, T. Snehitha Reddy, S. Debbarma, V. Nath and V. Kannagi. (2023). Optical Fiber Alignment Aid with Image Processing on FPGA-based System, IETE Journal of Research, DOI: 10.1080/03772063.2023.2277868

31. W. Ni, H. Li, L. Lu and K. Qian. (2023). An optical character recognition algorithm for documents with cascading structures. 4th International Conference on Computer Vision, Image, and Deep Learning (CVIDL), Zhuhai, China, 2023, pp. 199-202. https://doi.org/10.1109/CVIDL58838.2023.10167350


## AUTHORS


**Sujith M** is currently a student at Vellore Institute of Technology Chennai, Chennai, India pursuing a BTech degree in computer science. His research interests are in the field of artificial intelligence and machine learning.

Corresponding author Email: sujith.m@vitstudent.ac.in

**Akash Raj ST** is currently a student at Vellore Institute of Technology Chennai, Chennai, India pursuing a BTech degree in computer science. His research interests are in the field of artificial intelligence and machine learning.

Email: akashraj.st2020@vitstudent.ac.in

**Yeshwanth B** is currently a student at Vellore Institute of Technology Chennai, Chennai, India pursuing a BTech degree in computer science. His research interests are in the field of artificial intelligence and machine learning.

Email: yeshwanth.b2020@vitstudent.ac.in

**Priyadarshini J** received her BE degree from Anna University, Chennai, India in 2006 and master's degree in computer science from Anna University, Chennai, India in 2008. She also completed her PhD in the field of information and communication at Anna University, Chennai in 2013. She is currently a professor in the school of computer science and engineering at Vellore Institute of Technology Chennai, Chennai, India. Her areas of interest are information and communication.

Email: priyadarshini.j@vit.ac.in

**Jayasudha M** received her BTech degree from Mailam Engineering College, Anna University, Chennai, India in 2005 and master's degree in computer and communication from SSN College of Engineering, Anna University, Chennai, India in 2008. She also completed her PhD in the field of cloud security at Vellore Institute of Technology Chennai, Chennai in 2022. She is currently an associate professor in the school of computer science and engineering at Vellore Institute of Technology Chennai, Chennai, India. Her areas of interest are network and cloud security.

Email: jayasudha.m@vit.ac.in