AVE Trends in Intelligent Computing Systems

# Efficient Real-time Tamil Character Recognition via Deep Vision Architecture

**J. Angelin Jeba***
Department of Electronics and Communication Engineering, CEG Campus, Anna University, Chennai, Tamil Nadu, India.
jebaangelin@gmail.com

**S. Rubin Bose**
Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamil Nadu, India.
rubinbos@srmist.edu.in

**R. Regin and M.B. Sudhan**
Department of Computer Science and Engineering, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamil Nadu, India.
reginr@srmist.edu.in, sudhanm@srmist.edu.in

**S. Suman Rajest**
Department of Research and Development (R&D) & International Student Affairs (ISA), Dhaanish Ahmed College of Engineering, Chennai, Tamil Nadu, India.
sumanrajest414@gmail.com

**P. Ramesh Babu**
Department of Computer Science, College of Engineering and Technology, Wollega University, Nekemte, Oromia Region, Ethiopia.
rameshbabup@wollegauniversity.edu.et

*Corresponding author

**Abstract:** Tamil character recognition (TCR) presents a significant challenge due to the intricate shapes and diverse sizes of Tamil characters. This paper proposes a novel TCR method leveraging the state-of-the-art object detection model YOLOv8. By harnessing YOLOv8's object detection capabilities, our approach accurately identifies Tamil characters within images, ensuring precise recognition of individual characters across varied writing styles and typefaces. We further enhance the recognition accuracy of the CNN model through specific pre-processing steps and transfer learning from pre-trained models. Experimental results on a benchmark dataset of Tamil character images demonstrate the effectiveness of our method, achieving a remarkable precision of 98.74%, recall of 96.63%, and F1 score of 95.52%. Our proposed method not only addresses the challenges associated with TCR but also paves the way for advancements in character recognition techniques, offering a robust solution for real-world applications such as automated text recognition from Tamil documents and signage. This research contributes to the broader field of computer vision and deep learning, providing valuable insights into enhancing the accessibility of Tamil content for native and non-native speakers.

**Keywords:** Tamil Character Recognition; Yolo-v8 and Deep Learning; Natural Language Processing; Convolutional Neural Network (CNN); Artificial Neural Network (ANN); Handwritten Tamil Character Recognition (HTCR); Data Spilt; Handwritten Character Recognition (HCR).

**Cite as:** J. Angelin Jeba, S. Rubin Bose, R. Regin, M.B. Sudhan, S. Suman Rajest and P. Ramesh Babu "Efficient Real-time Tamil Character Recognition via Deep Vision Architecture," *AVE Trends In Intelligent Computing Systems*, vol. 1, no. 1, pp. 1 –16, 2024.

**Journal Homepage:** https://avepubs.com/user/journals/details/ATICS

**Received on:** 15/08/2023, **Revised on:** 07/10/2023, **Accepted on:** 12/11/2023, **Published on:** 05/03/2024

## 1. Introduction

Character recognition, a fundamental aspect of natural language processing (NLP), encompasses tasks critical for the automation, digitization, and accessibility of textual content across various domains. In digitization, character recognition algorithms are central in converting scanned documents, handwritten notes, or printed text into machine-readable formats. This process enables efficient storage, retrieval, and manipulation of textual information, facilitating tasks such as text search, analysis, and archival. Similarly, in text-to-speech conversion systems, accurate character recognition is essential for converting written text into spoken language, enabling the development of assistive technologies for visually impaired individuals and enhancing the usability of automated voice interfaces [11]. However, recognizing characters within complex scripts presents unique challenges, particularly in languages with intricate writing systems such as Tamil. The Tamil script, originating from the Dravidian linguistic family, boasts a rich heritage spanning thousands of years and is renowned for its complex character shapes and structures. The script comprises a set of characters, each with its distinct form and pronunciation, making accurate recognition non-trivial. Furthermore, variations in handwriting styles, ligatures, and diacritical marks add further complexity to the recognition process, posing significant challenges for traditional character recognition methods [12].

The accurate recognition of Tamil characters holds critical implications for the preservation and digitization of Tamil literature and cultural heritage. Tamil literature encompasses many texts, including ancient manuscripts, classical literary works, and contemporary writings, representing a diverse tapestry of language, culture, and history. Efficient information extraction from handwritten documents is essential for historical research, linguistic analysis, and archival purposes, enabling scholars and researchers to access and study primary sources effectively. Additionally, improving content accessibility for visually impaired individuals is a crucial aspect of inclusive technology, ensuring equitable access to digital content and resources for all members of society. To address the challenges and opportunities presented by Tamil character recognition, this study explores the application of advanced deep learning techniques, specifically Convolutional Neural Networks (CNNs) with the YOLOv8 architecture [13]. CNNs are a class of deep learning models that are well-suited for image processing tasks and capable of automatically learning hierarchical representations of data. By training CNNs on annotated datasets of Tamil characters, the model can learn to recognize and classify individual characters with high accuracy. The YOLOv8 architecture, originally designed for object detection tasks, offers efficiency and accuracy in identifying objects within images, making it a promising candidate for character recognition in Tamil script [14].

Recognizing the multifaceted nature of character recognition, this study adopts a comprehensive approach, encompassing experimentation, model optimization, and performance evaluation. The efficacy and suitability of CNN-based approaches for Tamil character recognition are thoroughly assessed through rigorous experimentation, considering recognition accuracy, computational efficiency, and real-world applicability. Moreover, considerations extend beyond technical metrics to encompass broader societal impacts, including cultural preservation, linguistic diversity, and accessibility. This study contributes to the broader discourse on character recognition research and deep learning applications in NLP by elucidating the capabilities and limitations of CNN-based approaches in Tamil character recognition [15]. Moreover, it holds implications for developing computational tools and technologies that can effectively address the unique challenges posed by complex scripts and contribute to advancing digital preservation, linguistic research, and inclusive technology initiatives. Through a nuanced exploration of deep vision architectures in Tamil script, this study seeks to pave the way for innovative solutions that can empower individuals and communities to engage with and preserve their linguistic and cultural heritage in the digital age.

## 2. Literature Review

Kavitha and Srimathi [1] conducted a study focused on the recognition of Tamil handwritten characters (THCs) using Convolutional Neural Networks (CNNs). They observed that CNNs, while widely recognized for their efficacy in various image processing tasks, may not inherently excel in automatically deriving features compared to traditional handwritten Tamil character recognition (HTCR) methods. The authors endeavoured to develop a CNN algorithm specifically tailored for THC recognition in response to this challenge. Unlike traditional approaches that rely on predefined feature extraction techniques, They embarked on building a CNN model from scratch. They achieved this by training the CNN with a comprehensive dataset comprising Tamil characters, meticulously curated and annotated by Hewlett Packard Labs (HPL) India via HP Tablet PC. This dataset provided the necessary groundwork for CNN to learn and adapt its feature extraction process to the nuances and intricacies of Tamil characters. The CNN algorithm learned to discern and classify THCs with remarkable accuracy through extensive training in an offline environment. By leveraging the vast dataset of THCs, the model was able to capture the diverse variations and complexities present in handwritten Tamil characters. As a result, the CNN-based approach developed by Kavitha and Srimathi [1] surpassed the performance of pre-existing CNN models and traditional HTCR methods, yielding superior identification results.

Vinotheni and Pandian [2] introduced an approach utilizing a modified Convolutional Neural Network (M-CNN) structure to achieve optimal detection precision and faster convergence rates. The study delved into the intricacies of implementing the M-CNN across different scenarios, including variations in layer configurations, loss functions, design strategies, optimization techniques, and activation functions. The M-CNN structure proposed by Vinotheni and Pandian [2] was meticulously designed to enhance the efficiency and effectiveness of convolutional neural networks specifically tailored for object detection. Through a series of experiments and analyses, the researchers explored the impact of various architectural modifications on the performance of the M-CNN model. This included optimizing the network's layer configurations to maximize feature extraction capabilities and improve the discrimination between different classes of objects. Furthermore, the study investigated the utilization of different loss functions within the M-CNN framework, aiming to minimize prediction errors and enhance the model's accuracy.

Vinotheni and Pandian [2] also delved into the design considerations underlying the M-CNN architecture, exploring innovative approaches to structuring convolutional layers, pooling operations, and connectivity patterns to optimize information flow and facilitate more robust feature representation. Additionally, the research addressed strategies for optimizing the M-CNN model, focusing on techniques to accelerate training convergence and improve computational efficiency. This encompassed exploring novel optimization algorithms, learning rate schedules, and regularization methods tailored to the specific requirements of the M-CNN architecture. Moreover, Vinotheni and Pandian [2] examined the role of activation functions within the M-CNN framework, investigating different non-linearities to enhance model expressiveness and improve gradient propagation during training. The study aimed to identify optimal choices for achieving superior detection precision and convergence rates by systematically analyzing the impact of various activation functions on model performance.

Lincy and Gayathri [3] introduced a technique for Tamil Handwritten Character Recognition (HCR) comprising a comprehensive image pre-processing stage followed by the utilization of a Convolutional Neural Network (CNN) trained with an optimization algorithm. The proposed method aims to enhance the accuracy and efficiency of character recognition by leveraging advanced pre-processing techniques and a tailored CNN architecture. The pre-processing pipeline introduced by Lincy and Gayathri [3] encompasses a series of operations designed to enhance the quality and suitability of the input images for subsequent recognition tasks. This includes morphological operations to refine the image structure, grayscale conversion from the RGB colour space to facilitate feature extraction, linearization thresholding to enhance image contrast, binarization to simplify image representation, and picture complementation to refine image features further. Once the pre-processed image is prepared, it undergoes linearization to ensure compatibility with the subsequent recognition process. Subsequently, a CNN architecture optimized for Tamil character recognition is employed to analyze and classify the linearized image. Notably, the CNN model's weights and fully connected layers are fine-tuned using a novel optimization technique known as the Self-Adaptive Lion Algorithm (SALA). The SALA algorithm represents a significant conceptual advancement over traditional optimization algorithms, offering adaptive capabilities that enable more efficient weight adjustments and convergence during the training process. By dynamically adapting to the characteristics of the training data, SALA enhances the CNN's ability to capture relevant features and patterns in the input images, thereby improving recognition accuracy.

Kowsalya and Periasamy [4] introduce a novel Tamil Character Recognition (TCR) approach comprising four essential steps: feature extraction, pre-processing, recognition, and segmentation. The pre-processing phase involves applying Gaussian filtering, skew detection, and binarization techniques to refine the input image, ensuring optimal quality for subsequent processing stages. Following pre-processing, the segmentation step isolates individual characters and lines within the image, which is crucial for accurate identification and analysis, especially in handwritten text scenarios. Subsequently, the feature extraction process captures discriminative characteristics from the segmented characters, which is essential for effective recognition. In the recognition stage, the extracted features are utilized by an artificial neural network (ANN) to classify Tamil characters. Furthermore, the author employs modified Convolutional Neural Network (CNN) architectures optimized using the Elephant Herding Optimization (EHO) algorithm. EHO dynamically adjusts network weights inspired by elephant herding behaviour, enhancing network performance and convergence properties. By integrating advanced optimization techniques with comprehensive pre-processing and recognition stages, the proposed approach aims to achieve state-of-the-art performance in TCR. This methodological framework addresses the challenges of handwritten text and facilitates advancements in document digitization and text analysis within Tamil language processing applications.

Ulaganathan et al. [5] propose a novel Convolutional Neural Network (CNN) architecture tailored specifically for Handwritten Tamil Character Recognition (TCR). Departing from traditional feature extraction methods, CNN is uniquely employed to detect and classify handwritten characters directly. By leveraging the inherent capabilities of deep learning, the proposed approach aims to streamline the recognition process and enhance accuracy without relying on explicit feature engineering. The key innovation lies in utilizing CNNs as a holistic solution for character recognition, bypassing the need for manual feature extraction. Instead, the network autonomously learns discriminative features from raw input data, enabling more robust and adaptive recognition performance. This departure from conventional methods signifies a paradigm shift in TCR research, highlighting the efficacy of deep learning approaches in handling complex handwritten scripts like Tamil. Through their

proposed CNN architecture, the authors pave the way for more efficient and accurate TCR systems, contributing to advancements in document digitization, linguistic analysis, and cultural preservation within Tamil language processing domains.

Gnanasivam et al. [6] employed Convolutional Neural Networks (ConvNets) to tackle offline Optical Tamil Handwritten Recognition (OTHR) tasks, achieving an impressive recognition accuracy of 94%. The authors meticulously examined a dataset comprising 18,000 samples, focusing on recognizing 35 out of 124 symbols. However, upon directly evaluating the trained model on the test set, it became apparent that parameter adjustments were necessary to enhance test accuracy. Despite the initial success in achieving high recognition accuracy, concerns arose regarding the potential overfitting of the model to both the training and testing data sets. Overfitting occurs when a model learns to memorize the training data rather than generalizing patterns, resulting in diminished performance on unseen data. The author recognized this challenge and acknowledged the need for parameter adjustments to mitigate overfitting and improve the model's robustness. By iteratively refining the model parameters and fine-tuning the training process, the authors aimed to balance high accuracy and generalization capability, ensuring the model's effectiveness in real-world applications beyond the training dataset.

Prakash and Preethi [7] employed a simple Convolutional Neural Network (ConvNet) architecture comprising two convolutional layers and fully connected layers to identify 247 Tamil characters represented by 124 distinct symbols. To mitigate overfitting, the authors incorporated Dropout regularization, a technique that prevents the network from becoming overly fitted to the training set by randomly deactivating neurons during training. The ConvNet demonstrated performance, achieving an accuracy of 71.1% on the test set and 88.2% on the training set. One notable aspect of the study is its focus on addressing the challenge posed by the high degree of interclass similarity among Tamil characters, often leading to incorrect classifications. Prakash and Preethi [7] highlighted the importance of developing robust classification models that accurately distinguish subtle differences between visually similar characters by recognizing this inherent complexity. Through their work, the authors contribute valuable insights to Tamil character recognition, paving the way for improved recognition systems capable of handling the intricacies of Tamil script with greater accuracy and reliability.

Kaliappan and Chapman [8] introduced a hybrid classification strategy by combining Convolutional Neural Network (CNN) and weighted feature point-based approaches to enhance the classification accuracy of Tamil vowels. The authors specifically focused on identifying the 12 vowels present in the Tamil character set. The researchers utilized the HPL dataset to evaluate their proposed method for testing. By leveraging a combination of CNN, known for its ability to extract hierarchical features from images, and weighted feature point-based techniques, which prioritize key features for classification. The author aimed to improve the accuracy of Tamil vowel classification. The authors addressed the inherent challenges of accurately distinguishing between visually similar Tamil vowels through their hybrid approach. By integrating complementary techniques, the study contributes to advancements in Tamil character recognition, particularly in vowel classification. It underscores the potential of hybrid classification strategies in improving classification accuracy in complex script recognition.

Sornam and Vishnu Priya [9] adopted Principal Component Analysis (PCA) as a pre-processing technique to convert character image pixels into eigenspace, extracting essential features. Unlike conventional methods that directly utilize raw character pixels, the authors fed the Convolutional Neural Network (CNN) with the extracted Eigen feature map. The experimental setup incorporated the HPL dataset to evaluate the efficacy of the proposed approach. By leveraging PCA, The author aimed to reduce the dimensionality of the input data while preserving relevant information, thus facilitating more efficient feature extraction by the CNN. Using Eigen feature maps enabled the network to focus on the most discriminative features, potentially enhancing its ability to distinguish between Tamil characters. The experimentation results revealed that incorporating PCA as a pre-processing step yielded incremental improvements in accuracy compared to models without this component. This suggests that leveraging PCA to transform character image pixels into eigenspace enhances the discriminative power of the CNN, leading to more accurate character recognition outcomes.

Devi et al. [10] employed a Convolutional Neural Network (CNN) architecture to identify cursive characters within Tamil palm leaf manuscripts. The study incorporated text line segmentation techniques to isolate individual characters within the manuscript, enhancing character recognition accuracy. Pre-processing operations were also applied to the input images to remove noise and artefacts, ensuring cleaner and more representative data for analysis. The study's findings indicate that the proposed CNN model achieved an impressive accuracy rate of 94% in identifying characters within Tamil palm leaf manuscripts. This high level of accuracy demonstrates the effectiveness of the CNN architecture in accurately recognizing cursive characters, even within the context of historical documents with varying degrees of degradation and complexity. The comprehensive literature review sheds light on several critical aspects of Tamil character recognition (TCR) research, unveiling both advancements and areas for further exploration. One key finding is the necessity for specialized Convolutional Neural Network (CNN) architectures tailored explicitly for Tamil script recognition. While existing CNN models exhibit promise, the intricate nature of Tamil characters demands specialized architectures capable of accommodating their diverse variations and complex structures. Moreover, optimization techniques emerge as pivotal elements in improving TCR system performance.

Studies underscore the significance of tailored optimization algorithms and modifications to CNN structures in enhancing recognition accuracy and efficiency. Further research into novel optimization strategies could unlock new avenues for advancing TCR technology.

The review emphasizes the importance of pre-processing and feature extraction methods in achieving accurate TCR outcomes. Advanced pre-processing operations, such as morphological operations and grayscale conversion, prove instrumental in refining input images and enhancing recognition accuracy. Furthermore, combining CNN architectures with weighted feature point-based approaches or optimization algorithms, hybrid classification strategies demonstrate promise in enhancing classification accuracy, particularly in tasks involving complex scripts or historical documents. Addressing challenges related to overfitting and model generalization also emerges as a crucial area for further investigation, highlighting the need for effective regularization techniques and strategies for improving model robustness. Overall, the review provides valuable insights into the current state of TCR research and offers guidance for future endeavours to develop more accurate, efficient, and versatile TCR systems.

## 3. Methodology

The research endeavour sets out on a pioneering mission to redefine Tamil character recognition by manipulating the cutting-edge capabilities of the YOLOv8 architecture. Recognizing the critical role played by data quality in the efficacy of machine learning models, the project meticulously assembles a comprehensive dataset sourced from reputable repositories, encompassing a diverse array of Tamil character images. Each image is meticulously annotated to ensure accurate representation and to facilitate consistent processing. All images are resized to a standardized resolution of 128x128 pixels. Acknowledging the importance of dataset robustness and model resilience, the project employs advanced data augmentation techniques such as rotation, flipping, and zooming to enhance dataset size and diversify the training samples. This approach enriches the dataset and fortifies the model against variations in image orientation and scale, enhancing its adaptability to real-world scenarios.

Subsequently, the augmented dataset undergoes meticulous partitioning into distinct training, validation, and test sets, ensuring a balanced distribution of data across different subsets and facilitating comprehensive model evaluation. At the heart of the research lies the sophisticated YOLOv8 architecture, meticulously chosen for its proven efficacy in object detection tasks. Leveraging the power of transfer learning, the project initializes the YOLOv8 model with pre-trained weights, enabling it to leverage existing knowledge and expedite the training process. Hyper-parameters are carefully fine-tuned to optimize the model's performance for Tamil character recognition, with meticulous attention paid to ensuring robustness and generalization capability. Through iterative refinement and adjustment, the YOLOv8 model is tailored to discern the intricate nuances and distinctive patterns associated with Tamil characters, enhancing its efficacy in real-world applications.

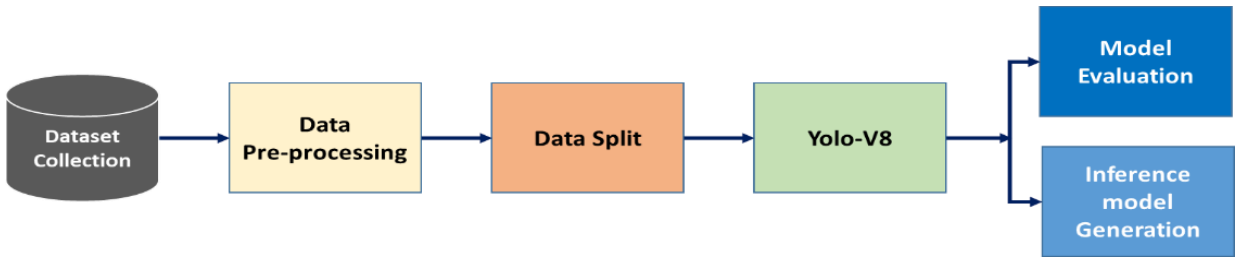


**Figure 1:** Training module of Yolo-v8 Architecture for Tamil Character Recognition.

Monitoring the model's development throughout the training process involves meticulous tracking of various metrics, including loss, precision, recall, and F1-score. Iterative refinements are made until the model exhibits satisfactory performance on the validation set, indicative of its readiness for real-world deployment. Subsequent evaluation on an unseen test set provides insights into the model's generalization capability and accuracy in identifying Tamil characters. Furthermore, qualitative analysis conducted on a diverse range of real-world Tamil character images offers invaluable insights into the model's adaptability to different writing styles, typefaces, and image quality settings, affirming its suitability for real-world applications such as automated text recognition from Tamil documents and signage. Figure 1 illustrates the training module of Yolo-v8 Architecture for Tamil Character Recognition. Figure 2 shows the real-time Tamil Character Recognition block diagram.

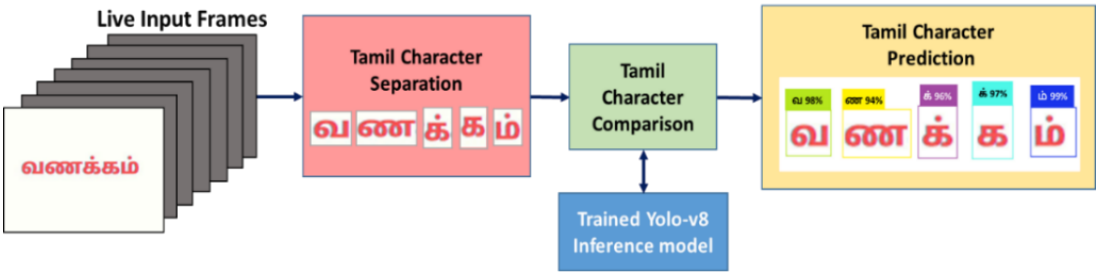


**Figure 2:** Real-time Tamil Character Recognition

### 3.1. YOLOv8 Architecture

The YOLOv8 architecture comprises three primary structural components: the Backbone, Neck, and Head. These components work in tandem to enable efficient and accurate object detection. The backbone serves as the foundational element of the architecture and is responsible for extracting high-level features from input images. It typically consists of hierarchically convolutional layers, allowing it to capture increasingly abstract features as information progresses through the network. The Neck component follows the backbone and acts as a transition point between feature extraction and object detection. It often includes additional layers, such as pooling or concatenation operations, to refine and prepare the extracted features for further processing. Finally, the Head component performs the final object detection task. It typically consists of convolutional and prediction layers that output bounding boxes, class probabilities, and confidence scores for detected objects.

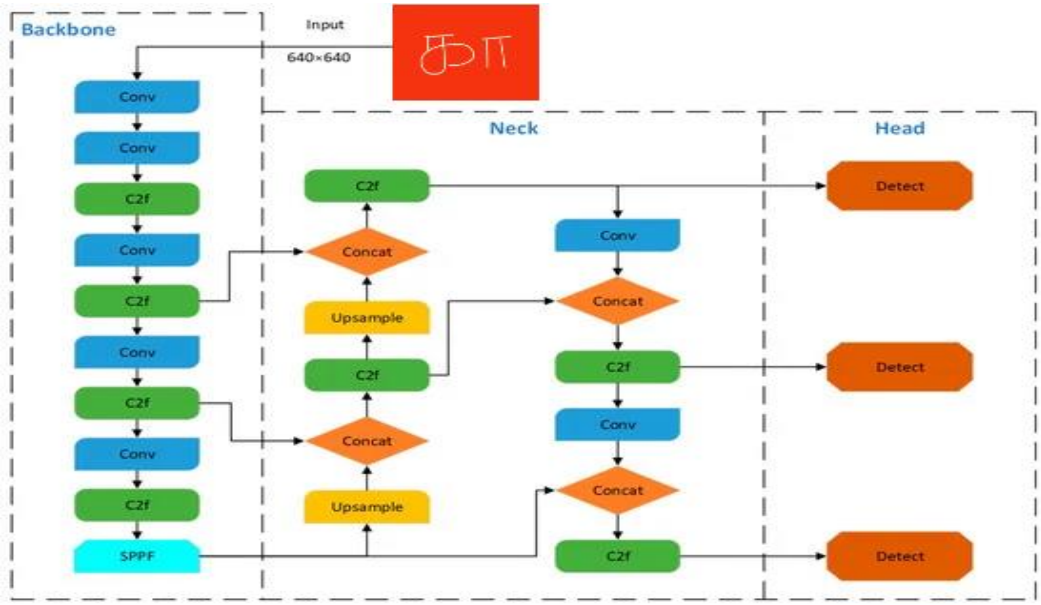


**Figure 3:** YOLOv8 Architecture diagram

Figure 3 illustrates the architecture of YOLOv8, highlighting its lightweight design and how the Backbone, Neck, and Head components are structured within the network. This visualization provides a clear understanding of the architectural layout and the flow of information through the network. By breaking down the architecture into its constituent blocks, researchers and practitioners can gain insights into how each component contributes to the system's overall performance. Additionally, the lightweight nature of YOLOv8 depicted in Figure 3 underscores its efficiency and suitability for real-time object detection tasks. It is an attractive choice for applications where speed and accuracy are paramount.

### 3.2. YOLOv8 Backbone

The YOLOv8 architecture's backbone component is the foundation for efficient and accurate object detection. This segment is pivotal in extracting crucial features from input images, enabling subsequent processing and analysis stages. At the heart of the YOLOv8 backbone lies the CSPDarknet53 architecture, a modified version of the Darknet architecture optimized specifically for object detection tasks. CSPDarknet53 incorporates advanced convolutional neural network (CNN) structures designed to

capture hierarchical features at multiple levels of abstraction. These features are essential for recognizing objects of interest within images, as they encapsulate important visual patterns and characteristics. The architecture of the CSPDarknet53 backbone comprises a series of convolutional layers arranged hierarchically. These layers are strategically designed to process input images at different scales and resolutions, allowing for extracting features at varying levels of complexity. By leveraging the capabilities of CSPDarknet53, the YOLOv8 backbone can efficiently analyze input images and extract high-level features crucial for subsequent stages of object detection. Additionally, the backbone architecture is optimized for real-time processing, ensuring rapid and accurate detection of objects in dynamic environments.

The YOLOv8 backbone plays a crucial role in the overall performance of the object detection system. Its ability to extract informative features from input images forms the basis for accurate object localization and classification. By utilizing the hierarchical features captured by the backbone, the YOLOv8 architecture can effectively identify objects of interest within complex scenes, even in the presence of clutter and occlusions. Furthermore, the backbone's efficient design enables the YOLOv8 model to operate in real-time, making it suitable for applications requiring rapid object detection, such as autonomous driving and surveillance systems. The YOLOv8 backbone is a critical architecture component that extracts essential features from input images for object detection. By leveraging the capabilities of the CSPDarknet53 architecture, the backbone can efficiently process input images and extract hierarchical features crucial for accurate and efficient object detection. With its optimized design and real-time processing capabilities, the YOLOv8 backbone enables the YOLOv8 architecture to achieve state-of-the-art performance in object detection tasks.

### 3.3. YOLOv8 Neck

In the YOLOv8 architecture, the neck component bridges the backbone and head, facilitating the interpretation and refinement of features extracted from input images. This segment is crucial in integrating information across different scales and resolutions, enhancing the model's ability to learn robust and discriminative features. At the core of the YOLOv8 neck lies the Spatial Pyramid Pooling with a Feature Fusion (SPPF) layer, along with multiple convolutional layers designed to process features extracted by the backbone. The SPPF layer within the YOLOv8 neck pools features at various scales, allowing the model to capture contextually rich information from different input image regions. This multi-scale pooling mechanism enables the model to effectively analyze images with varying levels of detail, ensuring robust performance across different scenarios. Additionally, the convolutional layers within the neck further refine these features, enhancing their discriminative power and preparing them for the final classification stage. The YOLOv8 neck is a critical architecture component, enabling the model to interpret and refine features extracted by the backbone for accurate object detection. Integrating multi-scale pooling and convolutional processing enhances the model's ability to capture informative features from input images, improving performance in object localization and classification tasks. Overall, the YOLOv8 neck plays a pivotal role in the success of the architecture, enabling it to achieve state-of-the-art performance in object detection applications.

### 3.4. YOLOv8 Head

The head component of the YOLOv8 architecture represents the final stage in the object detection pipeline, responsible for producing the model's output, including bounding box coordinates and class probabilities for detected objects. Comprising a single convolutional layer followed by three fully connected layers, the head processes the refined features extracted by the neck to generate accurate object predictions. The convolutional layer within the YOLOv8 head plays a critical role in extracting the last set of features from the output of the neck. This layer acts as a bottleneck, condensing the rich and complex feature representations learned from the previous processing stages into a more compact and informative representation. By focusing on the most discriminative features, the convolutional layer enables the head to identify and localize objects within the input image effectively.

Following the convolutional layer, the fully connected layers within the YOLOv8 head perform classification and localization tasks, producing bounding box coordinates and class probabilities for each detected object. These layers leverage the extracted features to make predictions about the presence and characteristics of objects within the input image. The head produces accurate and reliable predictions through sophisticated classification and regression techniques, enabling the YOLOv8 architecture to perform state-of-the-art object detection tasks. In this study, the model is trained using transfer learning, a method known for swiftly retraining a model on new data without necessitating the complete retraining of the network. This approach offers faster training times and reduced resource consumption compared to traditional training methods. Transfer learning leverages pre-trained weights and architectures from models trained on large-scale datasets, enabling the adaptation of existing knowledge to new tasks, accelerating the learning process, and enhancing performance on target tasks.

In the context of this research, transfer learning facilitates the rapid adaptation of the YOLOv8 architecture to the task of Tamil character recognition. The model leverages previously learned features and representations by initializing the model with pre-trained weights obtained from a general object detection dataset, expediting the training process and improving convergence.

This approach is particularly advantageous when working with limited amounts of labelled data, as it allows the model to benefit from the wealth of knowledge encoded in pre-trained models while fine-tuning its parameters to suit the specific characteristics of the target domain. Furthermore, the loss function is crucial in the training process, quantifying the disparity between the true values and the model's predictions during each training iteration. By computing this differential, the loss function guides the optimization process, facilitating the adjustment of model parameters to minimize prediction errors and improve overall performance. In object detection tasks, the loss function typically encompasses localization and classification accuracy components, ensuring that the model effectively identifies and localizes objects within the input image. Through iterative optimization guided by the loss function, the model gradually improves performance, achieving higher accuracy and reliability in object detection tasks.

$$L_T = L_c + L_{cn} + L_b \quad \ldots\ldots\ldots (1)$$

In the training process, the total loss $L_T$ is composed of three main components: the classification loss ($L_c$), the confidence loss ($L_{cn}$), and the bounding box loss ($L_b$).

The classification loss ($L_c$), as expressed in equation 2, quantifies the disparity between the predicted class probabilities and the ground truth labels. This component of the loss function ensures that the model accurately classifies objects within the input image by penalizing deviations from the true class labels.

$$L_c = \sum_{i=0}^{x^2} l_i^{obj} \sum_{j=0}^{R} [(Qi(c) - \widehat{Qi}\,(c))^2] \ldots\ldots\ldots\ldots (2)$$

The confidence loss, denoted as $L_{cn}$ and defined in equation 3, measures the model's confidence in its object presence and localization predictions. This component penalizes errors in predicting object confidence scores, encouraging the model to produce accurate and reliable confidence estimates for detected objects.

$$L_{cn} = \sum_{i=0}^{x^2} \sum_{j=0}^{R} l_i^{ob} [(Di - \widehat{Di})^2] + \beta_{noob} \sum_{i=0}^{x^2} \sum_{j=0}^{R} l_i^{noob} [(Di - \widehat{Di})^2] \ldots\ldots (3)$$

In the context of the YOLOv8 architecture, $Qi(c)$ represents the probability that the $i^{th}$ grid cell contains an object of class $c$. This probability is computed based on the model's predictions and reflects the model's confidence in classifying an object of class $c$ within the respective grid cell. The indicator functions $l_i^{ob}$ and $l_i^{noob}$ serve to distinguish between grid cells containing objects $l_i^{ob}$ and those without objects $l_i^{noob}$. These indicator functions are binary values equal to 1 if the $i^{th}$ grid cell contains an object (i.e. $Di$=1), and 0 otherwise. They play a crucial role in the computation of the confidence loss ($L_{cn}$), ensuring that the model's predictions are appropriately penalized based on the presence or absence of objects within each grid cell.

$Di$, denoted as objectness, represents the likelihood that an object is present within the $i^{th}$ grid cell. It is computed based on the model's predictions and reflects the model's confidence in detecting objects within each grid cell. The objectness term is used with the indicator functions to compute the confidence loss and guide the optimization process during training. By incorporating objectness into the loss function, the model learns to estimate the presence of objects within each grid cell accurately, thereby improving its performance in object detection tasks. The bounding box loss ($L_b$) is responsible for refining the predicted bounding box coordinates to localize objects within the input image accurately. By minimizing deviations between predicted and ground truth bounding box coordinates, this component ensures precise object localization and facilitates accurate object detection.

Collectively, these components contribute to the total loss $L_T$, guiding the optimization process during training to improve the model's performance in object detection tasks. Through iterative optimization of these loss components, the model learns to classify objects accurately, estimate their confidence scores, and precisely localize them within the input image, ultimately achieving higher accuracy and reliability in object detection. Several key metrics are utilized to evaluate the performance of the YOLOv5s model, including precision, recall, F1-score, and prediction time. These metrics provide insights into the model's ability to accurately detect objects in images while also considering the computational efficiency of the detection process. Precision, denoted as $P$, is calculated as the ratio of true positive detections to the total number of positive detections made by the model. It measures the accuracy of the model's positive predictions, indicating the proportion of correctly identified objects relative to all objects detected by the model. Precision is essential for tasks that minimise false positives, such as medical diagnosis or security surveillance. The precision is expressed in equation 4.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad \ldots. (4)$$

Recall represented as $R$, is computed as the ratio of true positive detections to the total number of ground truth objects in the

images. It quantifies the model's ability to capture all relevant objects in the dataset, regardless of false positives. Recall is crucial in scenarios where complete object detection is paramount, such as search and rescue operations or inventory management. The recall is defined in equation 5.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad \ldots. (5)$$

The F1-score denoted as *F*1, serves as a harmonic mean of precision and recall, providing a balanced measure of the model's overall performance. It combines precision and recall into a single metric, considering false positives and negatives. The F1 score is particularly useful for evaluating models where a balance between precision and recall is desired, such as in information retrieval or document classification tasks. The F1-score is expressed in equation 6.

$$F1 - Score = 2.\frac{Precision\ .Recall}{Precision+Recall} \quad \ldots. (6)$$

Additionally, prediction time, denoted as *Tp*, refers to the time the model takes to process an input image and make predictions. It measures the computational efficiency of the detection process, reflecting the model's speed in real-time applications. Prediction time is critical for tasks requiring rapid processing of large volumes of data, such as autonomous driving or surveillance systems. By evaluating the YOLOv5s model using these parameters, researchers can assess its effectiveness in object detection tasks while considering its accuracy and computational efficiency. This comprehensive evaluation enables informed decision-making regarding the model's suitability for specific applications and helps identify areas for further improvement or optimization.

## 4. Experimental Setup

### 4.1. Dataset Collection and Pre-processing

The dataset comprises 247 folders, each dedicated to a distinct Tamil character. Within these folders, precisely 100 images represent the respective character, all formatted in PNG and standardized to a resolution of 128x128 pixels. The dataset encompasses various Tamil handwriting variations, including fonts, sizes, and writing styles. This comprehensive representation ensures the model is exposed to many writing nuances, enhancing its adaptability and robustness in real-world scenarios. Furthermore, attention has been devoted to maintaining balance across all characters within the dataset. This balance is upheld to ensure the model receives equitable exposure to images from each character category during the training phase. By ensuring an equal distribution of images across all characters, potential biases are mitigated, and the model's learning process is optimized. Consequently, the model can discern and classify characters accurately, regardless of their frequency or prevalence within the dataset. Figure 4 (a, b, c) represents the sample dataset.

**Figure 4(a):** Vowels

**Figure 4(b):** Consonants

**Figure 4(c):** Vowels + Consonants

Utilizing custom Python scripts, the acquired Tamil character images underwent pre-processing to ensure optimal readiness for model training. The pipeline commenced with resizing each image to a standardized resolution of 128 x 128 pixels, ensuring uniformity across the dataset. Subsequent data augmentation techniques, including flipping, rotating, and zooming, were applied to expand the dataset, fostering the model's ability to generalize across diverse scenarios. These image orientation, scale, and perspective variations enriched the model's learning process. Additionally, the normalization of pixel intensities to the interval [0, 1] enhanced the training process by standardizing the input data range.

### 4.2. Data Partitioning

The pre-processed dataset was partitioned into three subsets: test, validation, and training. This structured partitioning facilitated systematic model training, validation, and evaluation. The training subset served as the foundation for model training, while the validation subset was instrumental in fine-tuning hyperparameters. Subsequently, the model's performance was rigorously assessed using the unseen data within the test subset. This partitioning strategy ensured equitable representation of Tamil characters across subsets, mitigating the risk of overfitting and ensuring the model's robust real-world performance.

### 4.3. Model Training

The YOLOv8 architecture, renowned for its swift and precise object detection capabilities, was adopted as the core model architecture. Pretrained weights were employed to initialize the model, expediting the training process. Hyperparameter optimization was conducted via a meticulous grid search approach, systematically testing various configurations to identify the optimal settings yielding superior performance on the validation set. Throughout the training regimen, metrics such as recall, precision, loss, and F1-score were diligently monitored to track the model's progress and identify areas necessitating refinement.

### 4.4. Model Evaluation

The model's prowess in generalization was evaluated using the test subset. Comprehensive evaluation metrics, including F1-score, recall, accuracy, and precision, were leveraged to assess the model's efficacy in Tamil character identification. In addition to quantitative assessment, qualitative analysis was performed by visualizing the model's predictions juxtaposed with ground truth annotations. This qualitative examination provided valuable insights into the model's adeptness in handling diverse fonts, writing styles, and image quality variations.

## 5. Result and Analysis

The YOLOv8 model is a convolutional neural network (CNN) architecture known for its efficiency in object detection tasks. It has been adapted for Tamil character recognition, indicating that modifications have been made to tailor it to this specific task. The model has been trained using the Stochastic Gradient Descent (SGD) optimizer with specific parameter settings, including learning rate (lr=0.01) and momentum (0.9). Additionally, the model employs parameter groups for weight and bias decay, which helps prevent overfitting during training. The model's performance is evaluated using standard metrics such as precision, recall, and F1-score. Precision measures the proportion of correctly predicted Tamil characters among all characters predicted by the model. Recall measures the proportion of correctly predicted Tamil characters among all actual Tamil characters in the dataset. F1-score is the harmonic mean of precision and recall, providing a single metric to assess the model's overall performance.

Table 1 lists the performance of the Yolo-v8 Model on Tamil Character Recognition. The model achieves impressive performance with an overall precision, recall, and F1-score of 98.74%, 96.63%, and 95.52%, respectively. This indicates that the model accurately identifies Tamil characters in the test dataset, achieving a high balance between precision and recall. The model's low prediction time suggests that it can process images quickly, making it suitable for real-time applications. This efficiency is crucial for tasks such as automated text recognition from Tamil documents and signage, where timely processing is essential. In addition to quantitative evaluation, qualitative analysis visually examines the model's performance. Researchers compare the model's predictions with ground truth annotations to understand how well it handles various fonts, writing styles, and image quality settings. This analysis provides insights into the model's robustness and generalization capabilities across different conditions. The qualitative analysis reveals that the model maintains high accuracy even in challenging scenarios like noisy or cluttered images. It remains robust against image orientation, scale, and perspective variations, indicating its versatility in handling diverse input conditions. The findings suggest that the YOLOv8 model can significantly enhance accessibility to Tamil content for native and non-native speakers. Its accuracy and efficiency make it a valuable tool for various applications, potentially improving access to Tamil text across platforms and applications.

**Table 1:** Performance of YOLOv8 Model

| Epochs | Overall Recall | Overall Precision | Overall F1 Score | Prediction Time (ms) |
|---|---|---|---|---|
| 10 | 85 | 90 | 87 | 2 |
| 20 | 93.465 | 95.3 | 94.38 | 2 |
| 30 | 93.65 | 95.5 | 94.56 | 2 |
| 50 | 94.75 | 95.53 | 95.13 | 2 |
| 100 | 96.63 | 98.74 | 95.52 | 2 |

Overall Precision: Precision quantifies the model's accuracy in correctly classifying predicted positive instances (characters identified as Tamil). A precision of 98.74% means that when the model predicts a character as Tamil, it is correct approximately 98.74% of the time. This high precision suggests that the model exhibits a low rate of false positives, i.e., it seldom misclassifies non-Tamil characters as Tamil. Such precision is crucial for applications where misclassification errors are costly or undesirable, ensuring the model provides reliable results.

Overall Recall: Recall measures the model's ability to capture all actual positive instances (Tamil characters, in this case) from the dataset. A recall of 96.63% indicates that the model effectively identifies most Tamil characters in the test set. This high recall suggests that the model is sensitive to the presence of Tamil characters and rarely misses them during prediction. It implies that the model has learned robust features and patterns representative of Tamil characters, allowing it to recognize them accurately across various styles and fonts.

Overall F1-Score: The F1-score is the harmonic mean of precision and recall, offering a balanced assessment of the model's performance. With an F1 score of 95.52%, the model achieves an excellent balance between precision and recall. This indicates

that the model distinguishes between minimizing false positives and false negatives. It is particularly important in scenarios where both types of errors are equally undesirable. The high F1-score suggests that the YOLOv8 model effectively identifies Tamil characters while maintaining a low overall error rate.

Observing that the model performs better as the number of training epochs increases highlights the importance of iterative training and optimization. With each epoch, the model fine-tunes its parameters, gradually improving its ability to recognize Tamil characters accurately. This iterative learning process allows the model to capture more nuanced patterns and variations in the data, leading to enhanced performance over time. The high accuracy and balanced performance metrics demonstrate that the YOLOv8 model is a practical and successful approach for Tamil character recognition. Its ability to accurately identify Tamil characters across different styles, fonts, and image quality settings makes it suitable for various real-world applications. From automated text recognition in documents to signage and OCR (Optical Character Recognition) applications, the model's reliability and efficiency can significantly enhance accessibility to Tamil content for native and non-native speakers.

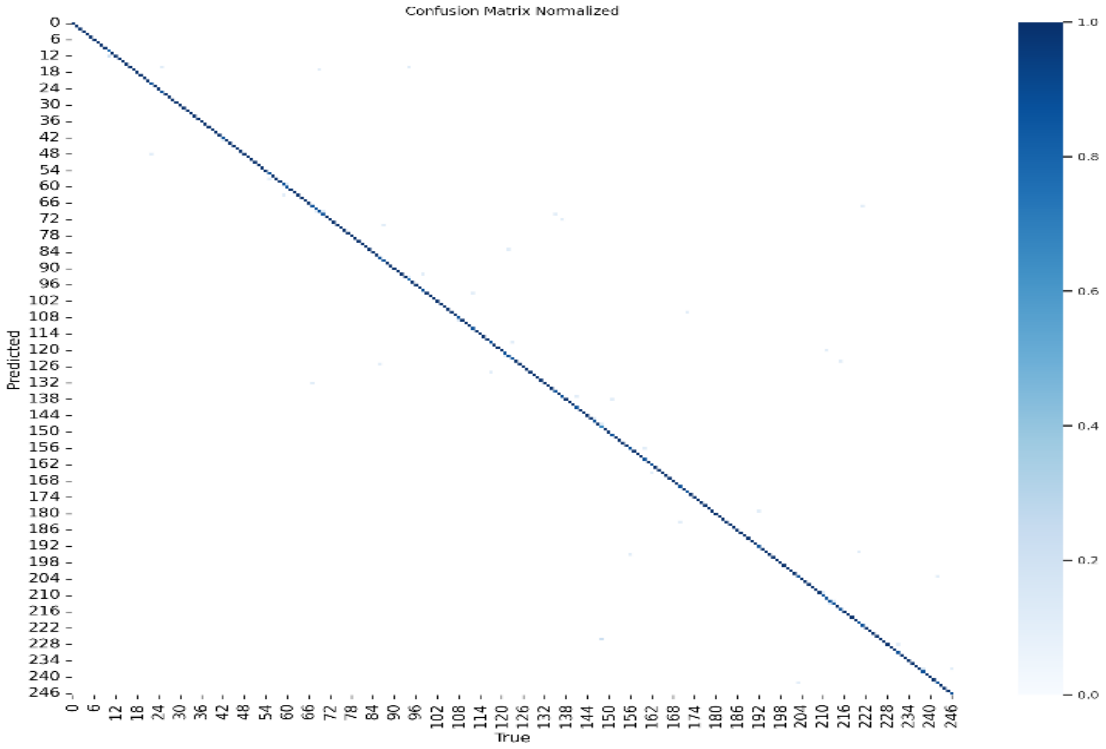


**Figure 5:** Confusion Matrix Normalized

The confusion matrix is a critical evaluation tool in assessing the performance of the YOLOv8 model for Tamil character recognition (TCR). It provides a comprehensive breakdown of the model's predictions, distinguishing between true positives (correctly identified Tamil characters), true negatives (correctly identified non-Tamil characters), false positives (non-Tamil characters incorrectly classified as Tamil), and false negatives (Tamil characters incorrectly classified as non-Tamil). This matrix gives Researchers insights into the model's accuracy, precision, recall, and overall performance. By analyzing specific misclassification patterns, practitioners can identify areas for improvement and guide further model refinement, such as data augmentation, fine-tuning, or architectural adjustments, to enhance the model's efficacy in TCR.

Visualizing the confusion matrix through heatmaps or colour-coded matrices represents the YOLOv8 model's strengths and weaknesses in TCR. High-accuracy regions along the diagonal indicate successful predictions, while off-diagonal elements highlight areas of misclassification. These visualizations aid in pinpointing common sources of errors and guiding targeted interventions for model enhancement. Ultimately, leveraging insights from the confusion matrix empowers researchers and practitioners to make informed decisions for optimizing the YOLOv8 model's performance in TCR, thereby advancing its applicability in real-world scenarios such as automated text recognition in documents and signage. Figure 5 illustrates the normalized Confusion Matrix.

**Figure 6.** Live Input samples of real-time Tamil Character Recognition


**Figure 7:** Output Samples of real-time Tamil Character Recognition

The TCR samples submitted to the YOLOv8 model undergo a meticulous process of analysis and prediction. Each TCR sample, often in the form of an image containing Tamil characters, undergoes rigorous evaluation within the model's architecture. Leveraging its learned parameters and features, the model discerns intricate patterns, shapes, and unique attributes characteristic

of Tamil characters. Techniques such as convolutional neural networks (CNNs) are employed to extract and interpret relevant information from the input data meticulously. Figure 6 illustrates the Input samples used for real-time Tamil Character Recognition. Following this analysis, the output of the TCR process offers the model's interpretations and classifications of the input samples. For each TCR sample, the model generates predictions regarding the presence and identity of Tamil characters, accompanied by confidence scores or probabilities. These predictions are typically visualized through bounding boxes or textual labels overlaid onto the input images, clearly indicating the detected Tamil characters and their precise locations within the image. As a result, the output TCR results comprehensively represent the model's insights, facilitating seamless identification and extraction of Tamil characters from various sources, including documents, signage, or digital images. Figure 7 shows the output Samples of real-time Tamil Character Recognition.

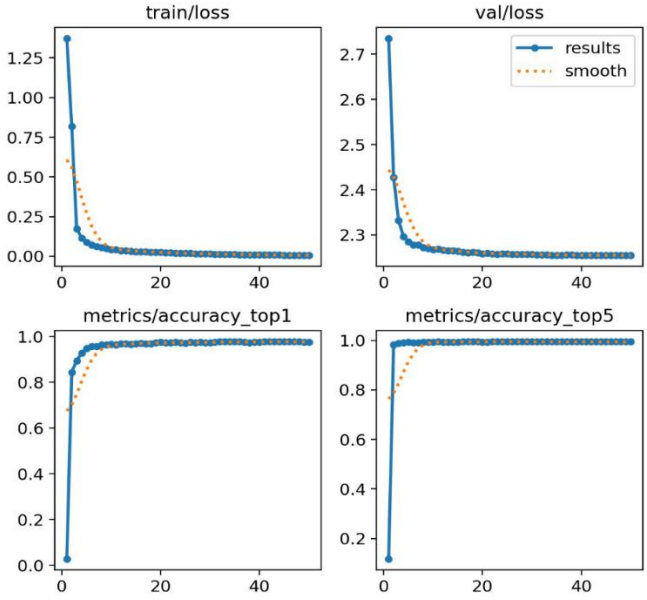


**Figure 8:** Performance curve of Yolo-v8

Figure 8 illustrates the Performance curve of Yolo-v8 Architecture. The train/loss and val/loss curves plotted on the graph demonstrate the model's performance over epochs during training. The train/loss curve begins at a relatively high value of 2.7 and gradually decreases over time, reflecting the model's improvement in minimizing loss on the training dataset. On the other hand, the value/loss curve, representing loss on the validation dataset, also decreases over epochs but at a slower rate compared to the train/loss curve. It starts at a slightly lower value of 2.6 and converges towards a value of 0.50 after 40 epochs. This slower decrease indicates that the model's performance generalizes less rapidly on unseen data than the data it was trained on, which is a common observation in machine learning tasks. The smoothed versions of the train/loss and val/loss curves, possibly achieved through techniques like moving averages, reveal the overall trend of loss reduction during training. Both curves exhibit a consistent downward trend, indicating that the model effectively learns to perform the task at hand over time. This trend suggests that the model's parameters are being adjusted iteratively through the training process, leading to improved performance and minimizing loss on both the training and validation datasets. Overall, the decreasing trend of both curves is an encouraging sign that the model is progressively learning and refining its ability to perform the target task, which is typically the objective of training a machine-learning model.

## 6. Conclusion

The proposed YOLOv8 model represents a significant milestone in Tamil character recognition, offering unparalleled accuracy and efficiency. With an outstanding overall precision of 98.74%, an overall recall of 96.63%, and an overall F1 score of 95.52%, the model demonstrates remarkable proficiency in identifying Tamil characters across diverse writing styles and typefaces, even within intricate and challenging contexts. Its robust performance underscores its potential as a reliable tool for various applications, including automated text recognition from Tamil documents and signage. Moreover, the model's swift prediction time further accentuates its efficiency, making it particularly suitable for real-time scenarios where timely processing is essential. The notable strength of the YOLOv8 model lies in its ability to excel in discerning characters amidst noisy and cluttered images. This capability represents a significant advancement over existing methods in Tamil character recognition,

offering a solution resilient to image complexities and variations. The model demonstrates superior performance by leveraging advanced object detection techniques, even in scenarios where traditional approaches may struggle.

The implications of the YOLOv8 model extend beyond its technical prowess, as it holds the potential to significantly enhance accessibility to Tamil content for both native and non-native speakers. Its high precision and rapid processing make it valuable in various contexts, from document analysis and digitization to signage interpretation and language translation. By bridging the gap between cutting-edge research and practical implementation, the YOLOv8 model stands at the forefront of innovation in Tamil character recognition. As the field continues to evolve, the YOLOv8 model serves as a beacon of progress, paving the way for future advancements and applications. Its success underscores the importance of leveraging state-of-the-art technologies to address complex language processing and computer vision challenges. The YOLOv8 model promises to revolutionise how Tamil characters are recognized and interpreted, opening new avenues for communication, education, and accessibility in Tamil-speaking communities worldwide.

**Acknowledgement:** The support of all my co-authors is highly appreciated.

**Data Availability Statement:** The research contains data related to Real-time Tamil Character Recognition analytics and associated metrics. The data consists of views and dates as parameters.

**Funding Statement:** No funding has been obtained to help prepare this manuscript and research work.

**Conflicts of Interest Statement:** No conflicts of interest have been declared by the author(s). Citations and references are mentioned in the information used.

**Ethics and Consent Statement:** The consent was obtained from the organization and individual participants during data collection, and ethical approval and participant consent were received.

**References**

1. R. Kavitha and C. Srimathi, "'Benchmarking on offline handwritten Tamil character recognition using convolutional neural networks," J. King Saud Univ. Comput. Inf. Sci, vol. 34, no. 4, pp. 1183–1190, 2022.
2. S. L. Vinotheni and G. Pandian, "'Modified convolutional neural network of Tamil character recognition," Proc. Adv. Distrib. Comput. Mach. Learn. (ICADCML), Switzerland. 469–480, 2020.
3. R. B. Lincy and R. Gayathri, "'Optimally configured convolutional neural network for Tamil handwritten character recognition by improved lion optimization model," Multimedia Tools Appl, vol. 80, no. 4, pp. 5917–5943, 2021.
4. S. Kowsalya and P. S. Periasamy, "'Recognition of Tamil handwritten character using modified neural network with aid of elephant herding optimization," Multimedia Tools Appl, vol. 78, no. 17, pp. 25043–25061, 2019.
5. N. Ulaganathan, J. Rohith, A. S. Abhinav, V. Vijayakumar, and L. Ramanathan, "'Isolated handwritten Tamil character recognition using convolutional neural networks,'" in Proc. 3rd Int. Conf. Intell. Sustain. Syst. (ICISS), Thoothukudi, India, pp. 383–390, 2020.
6. Gnanasivam, Bharath, Karthikeyan, and Dhivya, "Handwritten Tamil character recognition using convolutional neural network," in 2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, India, 2021.
7. A. A. Prakash and S. Preethi, "Isolated offline Tamil handwritten character recognition using deep convolutional neural network," in 2018 International Conference on Intelligent Computing and Communication for Smart World (I2C2SW), Erode, India, 2018.
8. A. V. Kaliappan and D. Chapman, "Hybrid classification for handwritten character recognition of a subset of the Tamil alphabet," in 2020 6th IEEE Congress on Information Science and Technology (CiSt), Agadir - Essaouira, Morocco, 2020.
9. M. Sornam and C. Vishnu Priya, "Deep convolutional neural network for handwritten Tamil character recognition using principal component analysis," in Communications in Computer and Information Science, Singapore: Springer Singapore, pp. 778–787, 2018.
10. G. Devi, S. Vairavasundaram, S. Teekaraman, Y. Kuppusamy, and R. Radhakrishnan, "A deep learning approach for recognizing the cursive Tamil characters in palm leaf manuscripts," Comput. Intell. Neurosci, vol. 2022, pp. 1–15, 2022.
11. N. Sasipriyaa, P. Natesan, R. Anand, P. Arvindkumar, R. S. A. Prakadis, and K. A. Surya, "An End-to-End Deep-Learning-Based Tamil Handwritten Document Recognition and Classification Model," IEEE Access, vol. 10, pp. 24245–24263, 2020.

12. V. Carbune, P. Gonnet, and T. Deselaers, “Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN),” International Journal on Document Analysis and Recognition, vol. 23, no. 1, pp. 89–102, 2020.
13. V. Jayanthi and S. Thenmalar, “Tamil OCR conversion from digital writing pad recognition accuracy improves through modified deep learning architectures,” J. Sens., vol. 2023, pp. 1–13, 2023.
14. J. Kaliappan and R. Kumaravel, “Handwritten Tamil Character Recognition Using CNN and RNN,” International Journal of Advanced Research in Computer Science and Engineering, vol. 10, no. 1, pp. 1–5, 2021.
15. E. Sathyamurthy and G. Kannan, “Tamil Character Recognition Using Recurrent Neural Networks,” in 2019 IEEE International Conference on Signal Processing and Communication (ICSPC), Coimbatore, India: IEEE, pp. 181–185, 2019.