# RECOGNITION OF ANCIENT STONE INSCRIPTION CHARACTERS USING AR AND ML

**A PROJECT REPORT**


***Submitted by***

**BERLIN MARIA V (210918104005)**

**RESHMA R (210918104039)**


***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**


**LOYOLA INSTITUTE OF TECHNOLOGY, CHENNAI**

**ANNA UNIVERSITY :: CHENNAI 600 025**



**JUNE 2022**

# ANNA UNIVERSITY::CHENNAI 600 025

## BONAFIDE CERTIFICATE

Certified that this project report **"RECOGNITION OF ANCIENT STONE INSCRIPTION CHARACTERS USING AR AND ML"** is the bonafide work of **BERLIN MARIA V (210918104005)** and **RESHMA R (210918104039)** who carried out the project work under my supervision.

**SIGNATURE**

**Dr. G. BHUVANESWARI,**

**HEAD OF THE DEPARTMENT,**

**PROFESSOR,**

Computer Science and Engineering,

Loyola Institute of Technology,

Palanchur,

Chennai – 600 123.

**SIGNATURE**

**Dr. G. BHUVANESWARI,**

**SUPERVISOR,**

**PROFESSOR,**

Computer Science and Engineering,

Loyola Institute of Technology,

Palanchur,

Chennai – 600 123.

# ACKNOWLEDGEMENT

A project of this magnitude and nature requires kind cooperation and support from many for successful completion. We wish to express sincere thanks to all of those who were involved in the completion of the project.

It is our immense pleasure to express our deep sense of gratitude to our honorable Chairman **Rev. Fr. Dr. J. E. ARULRAJ**, for providing such an excellent lab facility, our Correspondent **Rev. Sr. M K TERESA** and our Administrator **Rev. Sr. NAMBIKAI MARY** for continuous motivation to complete the project.

We are extremely thankful to our beloved and dynamic Principal **Dr. SUJATHA JAMUNA ANAND,** for having given us an opportunity to demonstrate our innovative abilities.

We would like to thank our Head, Department of Computer Science and Engineering, **Dr. G. BHUVANESWARI,** for the necessary help to complete the project successfully.

We take this moment as a great opportunity to thank our project coordinator **Mrs. C. KUDIYARASU DEVI,** Assistant Professor, Department of Computer Science and Engineering and our guide **Dr. G. BHUVANESWARI,** Head of the Department, Department of Computer Science and Engineering, for giving us their constant encouragement, support and valuable guidance to us during the project.

Last but not least, we are very grateful to our parents, friends and well-wishers for their constant encouragement during the project.

BERLIN MARIA V (210918104005)

RESHMA R (210918104039)

# ABSTRACT

Ancient Stone Inscriptions are the way our Ancestors communicate their traditional lifestyle, culture, creative thinking, architectural designs and so on to their generations. It also tells a lot about the ancient people and their works to improve the culture and the society. If we come to known the Stone Inscriptions, it will be useful to implement that knowledge for the growth of our present world.

Many Archeologist work day and night to identify each and every stone inscription and record it in a digital format so that anyone can access it with an ease. But even to those epigraphists it is a long process to understand the stone inscriptions and to convert it into modern Tamil characters.

Thus to make this process little simple and easy, the aim of this project is to help them to recognize the ancient stone inscription characters (Tamil characters) using Augmented Reality with Machine Learning Algorithm to understand the ancient Tamil characters with a help of an android application. Thus it will reduce the time for the archeologist. This implementation not only helps the research people, it is also useful even to a common man who has a great curiosity in Tamil culture with a minimal knowledge in the language. This proposed project identifies the ancient Tamil characters to its equivalent modern Tamil characters. Thus making the character identification process simpler and instantly.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATION

| WORD | ABBREVIATION |
|---|---|
| **APK** | Android Application Package |
| **AI** | Artificial Intelligence |
| **API** | Application Programming Interface |
| **AR** | Augmented Reality |
| **Con2D** | Convolution layer |
| **CNN** | Convolution Neural Network |
| **CPU** | Central Processing Unit |
| **FBX** | Filmbox |
| **GPU** | Graphics Processing Unit |
| **IDE** | Integrated Development Environment |
| **ML** | Machine Learning |
| **MaxPool** | Max Pooling Layers |
| **NLP** | Natural Language Processing |
| **NPDM** | Normalized Positional Distance Metric |
| **OCR** | Optical Character Recognition |
| **ONNX** | Open Neural Network Exchange |
| **ReLu** | Rectified Linear Unit |
| **SVM** | Support Vector Machine |
| **VS Code** | Visual Studio Code Editor |

# CHAPTER 1

## INTRODUCTION

The ancient characters are ambiguous, as it is different from modern Tamil letters. Researchers who are in archeology department have a long process to identify the ancient characters. If it is easy to identify Tamil characters, even in Tourist places, Tourist able to understand the language instantly. Ancient Stone Inscriptions provides more details about our tradition and our ancestor's lifestyle, which helps a lot to the present world for its growth and welfare.

Ancient Character Recognition (ACR) System is very helpful to Archeological Department that translates ancient letters into current modern characters. The technical challenges in ancient character recognition arise from two sources. First is an image defect: imperfections in stone image due to natural climates like wind, rain, lighting and thunder. Second is deformation: cracks and dents on rock surface treated as part of character and time dependent distortion. An Ancient Tamil Stone Inscription Characters have been considered for experimental analysis. Tamil is one of the oldest southern languages of India. The evolution of Tamil started from 3rd century BC. The beginning of evolution of Tamil comprised the period between the 3rd century BC and the 6th century AD, Medieval Tamil existed between the 6th century AD and the 12th century AD, and Modern Tamil, from the 12th century down to the present day. The ancient Tamil characters i.e. the early and medieval Tamil can be mostly found in stone inscriptions and palm leaves. Only epigraphists can read those stone inscriptions. To extend the readability and to preserve the ancient historical values, we need a good recognition system that can convert the ancient text to modern text.

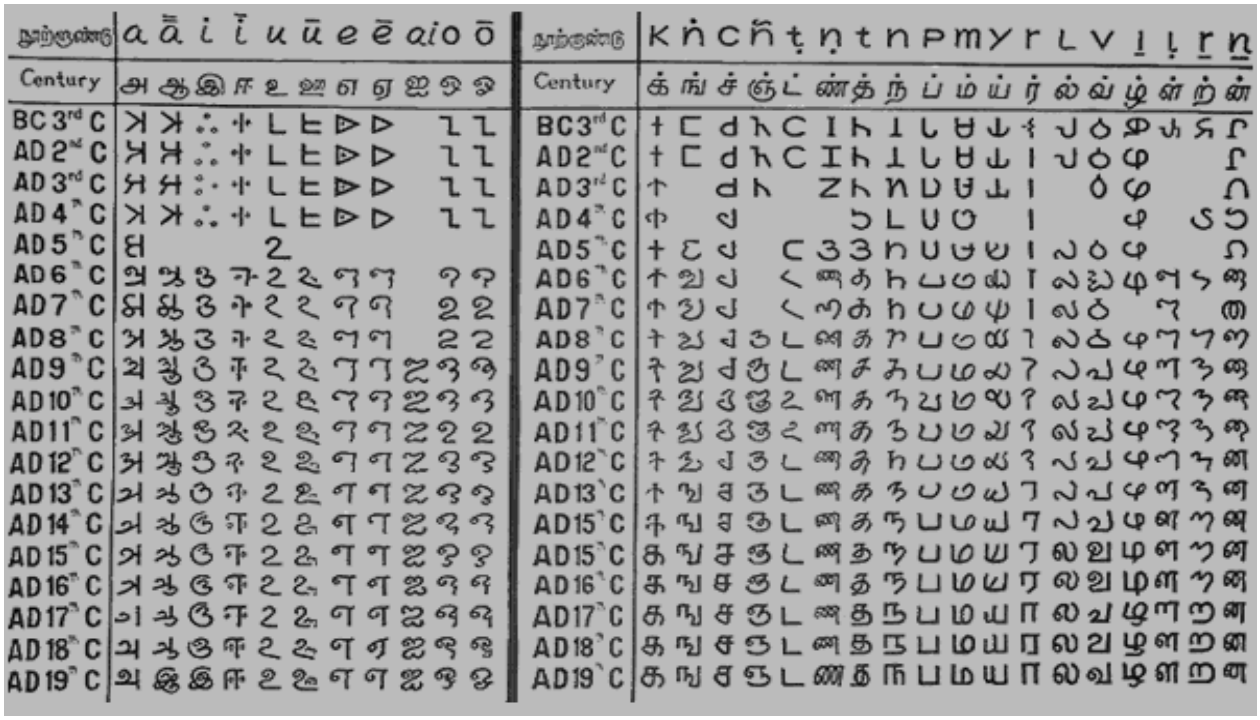


*Fig 1.1 Evolution of Tamil Script*

The above figure shows the Evolution History of Tamil Script from century to century.

Many people from various places are very much curious to know the solutions from ancestors as they lived a great life with minimal resources, But today's world with lots of resources, there are many problems which is unsolved. To learn solutions from our ancestors, the only way is to know the stone inscriptions and palm scripts. But the character cannot understand by a common man, Thus it again becomes a big problem.

To solve this with our digital technologies, This proposed system provides an Android application to identify the 11th Century Tamil characters and provides an equivalent Tamil Characters in the understandable format. Many researches still going on to make this complex process simpler. This paper provides a solution in a way of recognizing the ancient characters.

Technological Development brought many changes to our life and environment; it has become a revolution of this world. Artificial Intelligence and Augmented Reality made remarkable changes in our daily life. Everything is becoming automated now a day. One of the biggest boons to Technology is Machine Learning and Augmented Reality.

## Augmented Reality

Augmented Reality (AR) is the process where objects that reside on the real world are enhanced by computer generated perceptual information, sometimes across multiple sensory modalities, including visual, auditory, haptic, somatosensory and olfactory. It makes surrounding world more interactive which can also be manipulated digitally.

Augmented reality (AR) is an enhanced version of the real physical world that is achieved through the use of digital visual elements, sound, or other sensory stimuli delivered via technology. It is a growing trend among companies involved in mobile computing and business applications in particular. Digital elements are merely superimposed on real-world views.

With AR applications, we can bring experiences closer to users in their own environments through designs that are more directly engaging and personalized. It also has the integration of digital information with the user's environment in real time.

## Characteristics of Augmented Reality

- It allows the combination of the real world and the virtual world.
- **Depends on the context** - Thus, the information (images, sounds, videos, graphics, GPS data) displayed virtually is directly related to the information we see with our own eyes.
- It’s interactive in real time
- **Use the three dimensions -** The information is always shown with perspective, giving the feeling that it acquires the physical capacity of its surroundings. So, even if the information is virtual it’s integrated as if it has always been part of your world.

## Advantages of AR

- The AR system is highly interactive in nature and operates simultaneously with real time environment.
- It reduces line between real world and virtual world.
- It enhances perceptions and interactions with the real world.
- Due to its use in medical industry, life of patients has become safer. It helps in efficient diagnosis of diseases and in early detection of them.
- It can be used by anyone as per applications.
- It can be applied to part of training programs as it makes things memorable and eye catching.

## Machine Learning

Machine Learning (ML) is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. Machine Learning is the process of making the system to act without being programmed explicitly.

## Characteristics of Machine Learning

There were lots of examples that echo the characteristics of today's data rich world. Some of the characteristics are as follows.

**Automated Data Visualization** - Machine learning offers a number of tools that provide rich snippets of data which can be applied to both unstructured and structured data. By visualizing notable relationships in data, it not only make better decisions but build confidence as well.

**Automate task** - One of the biggest characteristics of machine learning is its ability to automate repetitive tasks and thus, increasing productivity.

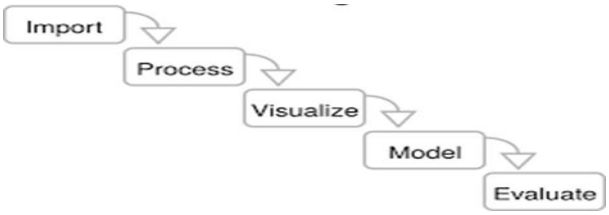



*Fig 1.2 Machine Learning workflow*

**Accurate Data Analysis** - Traditionally, data analysis has always been encompassing trial and error method, an approach which becomes impossible when we are working with large and heterogeneous datasets. Machine learning comes as the best solution to all these issues by offering effective alternatives to analyzing massive volumes of data. By developing efficient and fast algorithms, as well as, data-driven models for processing of data in real-time, machine learning is able to generate accurate analysis and results.

## Advantages of ML

**Continuous Improvement -** Machine Learning algorithms are capable of learning from the data we provide. As new data is provided, the model's accuracy and efficiency to make decisions improve with subsequent training.

**Automation for everything -** A very powerful utility of Machine Learning is its ability to automate various decision-making tasks.

**Trends and patterns identification** - ML Technologies explains how the various Supervised, Unsupervised and Reinforced learning algorithms can be used for various classification and regression problems.

## Augmented Reality and Machine Learning

Augmented Reality (AR) and Machine Learning (ML) are two big players in the technological world right now. Both of them are advancing at a rapid pace in an attempt to fuel the tech needs in their respective fields. Using AR and ML we can instantiate the real world objects with a new object which is augmented.

# OBJECTIVE

- To instantiate the ancient characters with the present Tamil characters, making the present world to understand the ancient characters without any ambiguity.
- To implement the ML techniques to identify the letters and AR to instantiate the present characters with the ancient characters in stone inscriptions.

# CHAPTER 2

## LITERATURE SURVEY

### [2.1] G.Bhuvaneswari, Dr.V.SubbiahBharathi,"Recognition of ancient Stone Inscription Characters using Normalized Positional Distance Metric Features",Asian Journal of Research in Social Sciences and Humanities Vol. 6, No. 5, May 2016, pp. 604-615. ISSN 2249-7315

This paper proposed a new feature called Positional Distance Metric which was independent to any language script to address the various issues occurred on stone inscribed images. Then Normalized Positional Distance Metric (NPDM) feature is computed and combined with structural and regional features to yield a better recognition rate. The proposed ZNPDM algorithm though simple, works efficiently for stone inscribed letters when combined with structural and regional properties of characters.

**Advantage**

- It classifies the characters using web application with an accuracy of 84.8% accuracy.

**Limitation**

- It does not utilize immediate recognition process as it requires a pre captured image for its recognition process.
- It does not implement android application for its process.

**[2.2] Dr.G.Bhuvaneswari, Dr.G.Manikandan "Recognition of Ancient Stone Inscription Characters Using Histogram of Oriented Gradients", International Conference on Recent Trends in Computing, Communication and Networking Technologies Oct 18-19, 2019. SSRN 3432300**

This paper proposed a new feature using Histogram of Oriented Gradients is proposed for ancient stone inscribed character recognition that is independent to any language script. This feature counts occurrences of gradient orientation in localized portions of an image. Here 11th century stone inscription digital images are used for creating database. Once the features are computed, the system is trained up with Support Vector Machine for classifying them into modern characters. Experiments are performed on 10 characters of 300 database size and showed significant recognition rate.

**Advantage**

- The HOG algorithm works efficiently for stone inscribed letters on grayscale images.

**Limitation**

- It does not utilize an Android Application for its recognition process.

**[2.3] T Manigandan, V Vidhya, V Dhanalakshmi, B Nirmala "Tamil character recognition from ancient epigraphical inscription using OCR and NLP", 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing August 2017, INSPEC 17859620**

This work mainly focus on recognition of various Tamil characters between 9th and 12th centuries using OCR and NLP techniques. During the segmentation process the color images were converted to gray image and to binary image based on threshold value. From segmented, image features like number of lines, curves, loops and dots have been extracted using Scale Invariant Feature Transform (SIFT) algorithms for each letter to identify the exact character.

**Advantage**

- It uses Unicode Mapping techniques, for identification of character between 9th and 12th century.

**Limitation**

- This system can be improved to get tested with manuscripts to know more information about the ancient Tamil Culture.

**[2.4] Faustina Jeya Rose.R, Bhuvaneswari.G, "Word Recognition Incorporating Augmented Reality For Linguistic E-Conversion", International Conference on Electrical, Electronics, and Optimization Techniques Nov 2016, INSPEC 16487731.**

This project worked with the development and implementation of Augmented Reality software for linguistic conversion on smart phones. This exploits smart phone's processing capabilities for sensing external objects using camera, to augment the real world objects with virtual data for character recognition and conversion from English Text to Tamil Language. The Tamil Text is overlaid on the captured English Text by providing the meaning to the word captured in modern language.

**Advantage**

- This development and implementation of a mobile based augmented reality application, which can detect English Text and translate into Tamil language in real time provides 90% accuracy.

**Limitation**

- Feature Enhancement can be improved.
- Hand written recognition is not possible.

**[2.5] Giridharan.R, Vellingiriraj.E.K, Dr. Balasubramanie "Identification of Tamil Ancient Characters and Information Retrieval from Temple Epigraphy Using Image Zoning", IEEE 2016 Fifth International Conference on Recent Trends in Information Technology Apr 2016, INSPEC 16320638.**

This paper proposed to develop a system that involves character recognition and information retrieval of Brahmi, Vattezhuthu and Grantha letters from temple epigraphy and their conversion to the present Tamil digital text format. It uses Image Zoning technique. It utilizes the web application for the working of this process.

**Advantage**

- The overall output of our algorithm is 84.57% accuracy.

**Limitation**

- Features extracted are low and some of the vowels are not recognized properly.
- Error rate is little higher.

**[2.6] G. Janani, V. Vishalini, P. Mohan Kumar "Recognition and analysis of Tamil inscriptions and mapping using image processing techniques" 2016 Second International Conference on Science Technology Engineering and Management (ICONSTEM) Mar 2016, INSPEC 16285367.**

This paper conveyed the result of finding each and every letter in the image and segmenting it separately and matching it with the template and finding it what it is according to present Tamil language. This process of segmenting each and every Tamil letter in the Tamil inscription we captured and matching it with the Tamil template and finding what the letter is according to modern Tamil language.

**Advantage**

- It utilizes the segmentation process, thus to provide high performance.

**Limitation**

- It can be improvised to utilize it in real time.

## [2.7] S. RajaKumar, V. Subbiah Bharathi, “Eighth century tamil consonants recognition from stone inscriptions” 2012 International Conference on Recent Trends in Information Technology. June 2012, INSPEC 12769959

The objective of this system was to employ support vector machines (SVM) in identifying various fonts in Tamil. The feature vectors are extracted by making use of Gabor filters and the proposed SVM is trained using these features. It is observed that this method is content independent.

**Advantage**

- It is content independent

**Limitation**

- It does not provide word recognition process.
- Feature extraction need to be improved.

**[2.8] Y.Liang, M.C.Fairhurst and R.M.Guest, "A Synthesisd Word Approach to Word Retrieval in Handwritten Documents", International Tamil Internet Conference Proceedings, Vol.45, Jun 2012, PP 4225-4236.**

This paper discussed about the implementation of a stemming algorithm that is available as Open Source Software. The algorithm implemented is a rule based iterative affix stripping algorithm. The algorithm is implemented using Snowball, a string processing language specifically used for implementing stemming algorithms. The algorithm was tested against the Tamil WordNet data.

**Advantage**

- It is used to identify and recognize complete meaningful word.
- It uses Tamil Electronic dictionary to identify the characters.

**Limitation**

- It does not identify the ancient Tamil characters.

## [2.9] Giuseppe Pirlo, Donato Impedovo, "Adaptive Membership Functions for Handwritten Character Recognition by Voronoi-Based Image Zoning", IEEE Trans on Image Processing, Vol 21, No 9, 2012,pp. 3827-3836

The objective of this paper is to develop computer software that can recognize the Ancient Tamil handwritten characters by using the genetic algorithm technique. First the image acquisition module collects an unknown input character from a user. Second, the input image is transformed into a suitable image for the feature extraction module. Third, the system extracts character features from the image.

**Advantage**

- It provides proper recognition of hand written Tamil characters through feature extraction.

**Limitation**

- To identify the ancient characters it cannot identify the loops and strikes of the character, thus leading to high error rate.

## [2.10] Bharath A, SriganeshMadhvanath, “Hidden Markov Models for Online Handwritten Tamil Word Recognition.” Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), HPL-2007-108, July 6, 2007

In this system, a data-driven HMM-based online handwritten word recognition system for Tamil, an Indic script, is proposed. The results are promising and warrant further research in this direction. The results are also encouraging to explore possibilities for adopting the approach to other Indic scripts as well.

**Advantage**

- Inter symbol pen-up strokes were modelled explicitly using two state left-to-right HMMs to capture the relative positions between symbols in the word context.

**Limitation**

- It causes relatively low performance in the case of high lexicon size.

# CHAPTER 3

## SYSTEM ANALYSIS

### 3.1 EXISTING SYSTEM

The existing system uses [1] the Zone Based Normalized Positional Distance Metric algorithm, it utilizes web based application and requires already pre scanned image of stone inscription along with a manual selection of each character to identify its equivalent modern Tamil character.

In [2] a new feature using Histogram of Oriented Gradients is proposed for ancient stone inscribed character recognition that is independent to any language script. In [3] it focus on recognition of various Tamil characters between 9th and 12th centuries using OCR and NLP techniques.

In [4] it worked with the development and implementation of Augmented Reality software for linguistic conversion on smart phones. [5] uses Image Zoning technique. It utilizes the web application for the working of this process.

In [6], it conveyed the result of finding each and every letter in the image and segmenting it separately and matching it with the template and finding it what it is according to present Tamil language. [7] was to employ support vector machines (SVM) in identifying various fonts in Tamil.

In [8], algorithm implemented is a rule based iterative affix stripping algorithm. The algorithm is implemented using Snowball, a string processing language specifically used for implementing stemming algorithms. [9] is to develop computer software that can recognize the Ancient Tamil handwritten characters by using the genetic algorithm technique.

The syntactical approach utilizes the syntax to identify the pattern of each character and recognize the ancient letters [10,11]. The statistical approach of character recognition uses the probability functions with measurements of characteristic pattern in the input features[12,13,14,15]. In [16] the system the input features like height, width is used for Unicode recognition. In [17] angle and Wavelet features are used to recognize the character. In [18, 19] the pattern is recognized through predictable structure of the neural networks to classify the ancient characters.

**Limitations**

- The existing system provides recognition of ancient stone inscription characters which are pre captured i.e. it can only recognize the characters from already captured image.
- It doesn’t utilize Android application to identify the characters instantly on-site.

## 3.2 PROPOSED SYSTEM

The proposed System uses Convolution Neural Network (CNN) Algorithm to train the data set and the pre trained model is integrated with the Unity 3D in the form of Open Neural Network Exchange (ONNX) model. The Android application is created using Unity 3D, which first classifies and compares the classified model with the pre trained model, if the threshold value of the comparison is above 80% it invokes the classified model's equivalent present 3D Model. Then it is deployed as apk file and built as an Android application which utilizes the Back Camera of the android devices to scan the input data.

**Advantages**

- It uses Android application to recognize the ancient stone inscription.
- It provides 86.26% overall accuracy.
- There is no need of internet connection to run the application.

# CHAPTER 4

## SYSTEM DESCRIPTION

## 4.1 ARCHITECTURE DIAGRAM

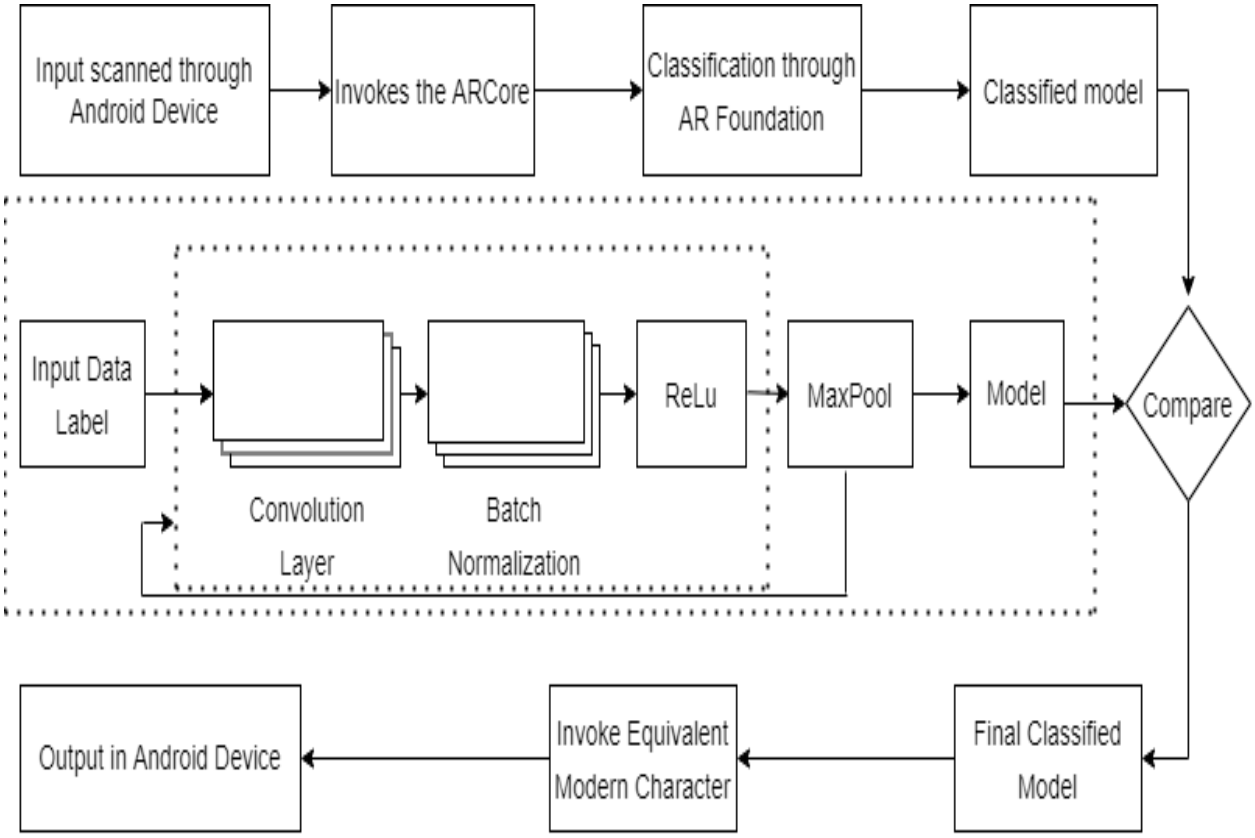



*Fig 4.1. Architecture diagram of our proposed system*

Based on the architecture shown in figure 4.1, the input is scanned through the back camera of android device, Once the image is tracked the application invokes the AR Core, The tracked image undergoes classification using the pre trained model which is integrated with the application in the onnx format. The training phase includes several layers to train the labels and to classify the model with greater accuracy. The classified model invokes the equivalent modern character which is instantiated as an output in the android display.

# CHAPTER 5

## SYSTEM SPECIFICATION

## 5.1 REQUIREMENT ANALYSIS

Requirement analysis determines the requirements of our proposed system. Thus our project analyzes the product and resource requirement. In product requirement the input and output is included. In our system the input is ancient stone inscription. With the input the output is instantiated over the input in the form of 3D Model of present Tamil Characters. The resource requirement includes the Hardware and Software requirements for our system.

### Hardware Requirements

The Hardware is the important thing to run any system. Thus Developers and software engineers use this hardware requirement as their first step in designing a system.

- In our system as our mobile application runs in an Android Device thus it must have the following requirements.

  PROCESSOR : HIGH SPEED PROCESSOR

  RAM : 2-4GB

- It also requires Graphic card (AMD R5M330) for developing the application.

### AMD R5M330

The AMD Radeon R5 M330 is a low-end dedicated graphics card for laptops. The core clock is pretty high with up to 1030 MHz, however, the bottleneck is the 64-bit DDR3 graphics memory (clocked at 1000 MHz, 2000 MHz effective). It is built using the 28nm Manufacturing process and its core configurations are 320:20:8.

### Software Requirements

Software Requirements are the software specification of the system. The software specification helps us to obtain the cost, team planning and also the performance ratio of the system.

In our system, the Windows software requirements for development are as follows,

- Operating System : Windows 10 Pro – 64bit.

- Software Tools : Unity 3D, Barracuda Package, Jupyter Notebook, Visual Studio Code Editor.
- Language : Python, C++.

The Mobile Software Requirements are as follows

- Operating System : Android/ ios
- Mobile Application : Deployed in apk format

## Unity 3D

Unity 3D is a gaming and movie software (Like Maya). Unity3D is a powerful cross-platform 3D engine and a user friendly development environment. Easy enough for the beginner and powerful enough for the expert.

The version used in Unity is 2021.3.2f1. This engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences. Unity3D is designed to democratize game development. Users have access to features that allow real-time graphics development in both 3D and 2D environments.

Unity is an incredible 3D package used for making video games, architectural and medical imaging and more. Unity is able to publish to Windows, OS X, and the web via the Unity Web Player. The Web Player is a browser plugin that works in all major browsers and offers the same performance available on the desktop.

## Barracuda Package

The Barracuda package is a lightweight cross-platform neural network inference library for Unity. The version used is 1.0.4. Barracuda can run neural networks on both the Graphics Processing Unit (GPU) and Central Processing Unit (CPU).Barracuda is a simple, developer-friendly API for neural network execution.

The Barracuda neural network import pipeline is built on the ONNX (Open Neural Network Exchange) format, which let us to bring in neural network models from a variety of external frameworks, including Pytorch, TensorFlow, and Keras.

## Jupyter Notebook

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience. It provides all the library files to be utilized in our pre trained model. It provides us support to implement the Convolution Neural Network for training the model with the labelled data set using ImageNet Database.

## Visual Studio Code Editor

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE. It when integrated with Unity supports only C++.

Visual Studio Code can be a great companion to Unity for editing and debugging C# files. All of the C# features are supported and more. Unity has built-in support for opening scripts in Visual Studio Code as an external script editor on Windows and macOS.

## Mobile Application

The application classifies the frames in realtime, displaying the top most probable configuration. It requires an ARCore supporting package to run our application. The minimum API level for Android is 19 (Android 4.4 – “Kitkat”) and the maximum API Level is 30 (Android 11 – “HoneyComb”). It utilizes the back camera of the android device to scan the ancient stone inscription.

It provides the output in the form of imposing a 3D model of the present Tamil Character over the scanned Ancient Tamil character. This is done through Augmented Reality.

## 5.2 BACK END

In the Back end of our mobile application Tensor Flow Lattice is implemented. The trained models from Tensor flow Lattice in either quantized or float model is deployed at the back end of our mobile application.

## Keras Model

Keras is used to build the pre trained model, which act as the backend for the mobile application. Keras is a neural network Application Programming Interface (API) for Python that is tightly integrated with TensorFlow, which is used to build machine learning models.

Keras' models offer a simple, user-friendly way to define a neural network, which will then be built for you by TensorFlow. Keras is a powerful and easy-to-use free open source Python library for developing and evaluating machine learning and deep learning models. A model is the basic data structure of Keras. Keras models define how to organize layers.

Keras is a high-level API that runs on top of TensorFlow. Keras simplifies the implementation of complex neural networks with its easy to use framework.

# CHAPTER 6

## MODULES DESCRIPTION

A Modular design reduces the complexity and it will be easier to implement the projects. Software with effective modularity will become easy for developers to develop the software. Each software is divided into modules. These modules are then integrated to satisfy the requirements of the software.

Modularity is the single attribute of software that allows a program to be intellectually manageable. The five important criteria that enable us to evaluate a design method with respect to its ability to define an effective modular design are: Modular decomposability, Modular Comps ability, Modular Understandability, Modular continuity, Modular Protection.

The following are the modules of this Project, which is planned in aid to complete this proposed system by overcoming the existing system and supporting for the future enhancements.

6.1 TRAINING PHASE.

6.2 INTEGRATION PHASE.

6.3 DEVELOPING AN ANDROID APPLICATION.

## 6.1 Training Phase

The training Phase utilizes the CNN algorithm to train the labels. The data sets for 18 characters with 850 labels are collected. The 18 characters are Ka, Nga, Sa, Nja, Ta, Na, Tha, N-a, Pa, Ma, Ya, Ra, La, Va, Zha, L-a, R-a, Nna. The Below Table explains the Number of Data collected in each Letters.

*Table 6.1 Data Set.*

| CHARACTER | TAMIL LETTER | NO. OF DATA |
|---|---|---|
| Ka | க | 80 |
| Nga | ங | 24 |
| Sa | ச | 29 |
| Nja | ஞ | 84 |
| Ta | ட | 64 |
| Na | ண | 38 |
| Tha | த | 55 |
| N-a | ந | 38 |
| Pa | ப | 80 |
| Ma | ம | 73 |
| Ya | ய | 24 |
| Ra | ர | 39 |
| La | ல | 69 |
| Va | வ | 36 |
| Zha | ழ | 26 |
| La | ள | 24 |
| R-a | ற | 24 |
| Nna | ன | 43 |
| Total Characters in data set | | 850 |

These data sets are trained using CNN which uses multiple layers to improve the accuracy of the trained data sets.

The accuracy of the trained model is as follows,

- 93.2% as training accuracy,
- 91.6% as testing accuracy,
- overall trained model accuracy is 90.3%.

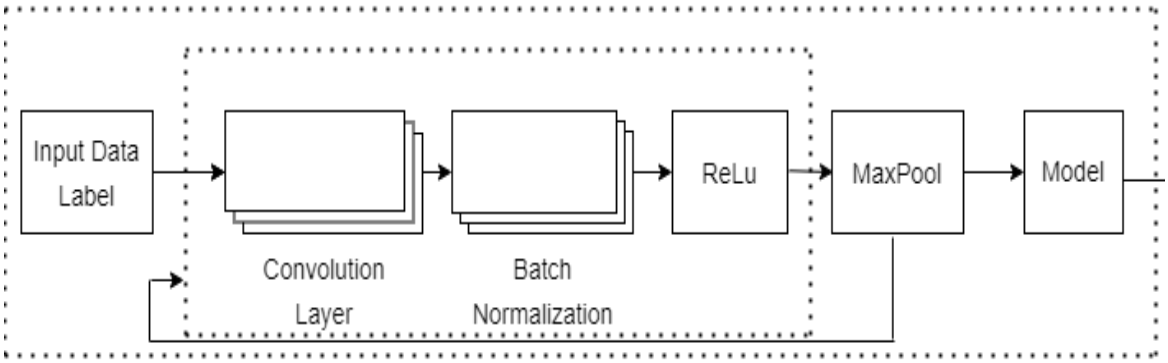



*Fig 6.1. Training Model Architecture.*

## Convolution Neural Network

Convolution Neural Network (CNN) is a type of artificial Neural Network used in image recognition process which is designed in the way of using the pixel values to process the data set.

A CNN uses a system much like a multilayer perceptron that has been designed for reduced processing requirements. The layers of a CNN consist of an input layer, an output layer and a hidden layer that includes multiple convolutional layers, pooling layers and normalization layers. The removal of limitations and increase in efficiency for image processing results in a system that is far more effective, simpler to trains limited for image processing and natural language processing.

## Advantages of CNN

- The main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision.
- It provides an efficient dense network which performs the prediction or identification etc. efficiently.
- Little dependence on preprocessing, decreasing the needs of human effort developing its functionalities.

- CNN uses parameter sharing and dimensionality reduction, which makes models easy and quick to deploy. They can be optimized to run on any device, even on smartphones.

## i. Convolution Layer

It uses Convolution layer (Con2D) which implements 64 convolutions along with 3 Padding and 2,2 strides for each block with frame kernel with the value 7. The height and width of the image is converted in the ratio of 224,224. Thus the output of the convolution layer will change the size of the image with its padding operation.

The resized window is determined using the formula,

Output window size of Con2D = (n+2p-f/s) +1

Where n denotes the ratio of height and width of the square image which is 224, p refers to the padding, f refers to the frame of the kernel, s refers to the strides.

**Advantage**

- This layer ensures the spatial relationship between pixels by learning image features using small squares of input data.

## ii. Batch Normalization Layer

In Batch Normalization the layer are scaled with 64 layers, which is done along small batches, instead of the full data set. It speeds up the learning rate and make the neural network faster and stable with adding of extra layers in a neural network. It regularizes the distortions in the photometric of the image. Thus the output of this batch normalization is determined by the formula,

Normalized Value = (x-E[x]) / sqrt(var(x))

Where x is the output value of the Con2D layer, E[x] is the mean of the batch, var(x) is the variance of the batch.

**Advantages**

- Reduces the dependence of gradients on the scale of the parameters or their initial values.
- Regularizes the model and reduces the need for dropout, photometric distortions, local response normalization and other regularization techniques.

### iii. Rectified Linear Unit

The Rectified Linear Unit (ReLU) used to increase the linearity in the labels. As the image labels has lot of non-linear functions like the transitions between pixels, borders, colors, etc. It is used as a hidden layer in the CNN process. It act as the activation layer for each block to classify the image. It provides the output directly if the input is positive, else it will provide value as 0. If the value is 0 then no process takes place, else it starts its classification process.

**Advantage**

- It allows the model to learn faster and perform better.

### iv. MaxPooling Layer

The Max Pooling Layers are similar to the Convolution Layer, but instead of applying the convolution operation, the selection of maximum values in each input class is done. It invokes the block for five times to get the maximum value for this pre trained model. The maximum value of this block is used to recognize the ancient character, which is provided as the output model of the training phase.

**Advantage**

- It reduces the computational cost by reducing the number of parameters to learn and provides basic translation invariance to the internal representation.

## 6.2 Integration Phase

The Trained Model is converted into Open Neural Network Exchange (onnx) format. The onnx model is integrated with the Unity 3D to utilize the Augmented Reality and to build the Android application. The Unity barracuda supports the onnx format to implement the trained neural network in the Unity3D to implement the Augmented Reality with the application.

The pre trained model is integrated along with its label.txt file which contains the entire data label along with its index values which is used for further classification of the model.

The integration phase helps us to utilize the Machine Learning model in the Unity framework.

## ONNX

ONNX provides a definition of an extensible computation graph model, as well as definitions of built-in operators and standard data types. Each computation dataflow graph is structured as a list of nodes that form an acyclic graph. Nodes have one or more inputs and one or more outputs. Each node is a call to an operator. The graph also has metadata to help document its purpose, author, etc.

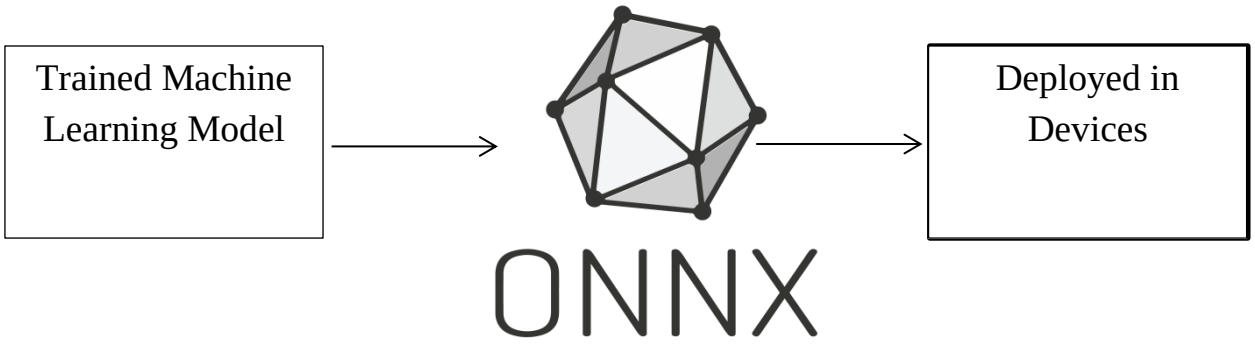



*Fig 6.2 ONNX Framework*

The trained model is converted as an ONNX format which can be utilized for the deployment in any devices, it act as transfer learning process which utilizes the Machine Learning model to be deployed in any device.

Operators are implemented externally to the graph, but the set of built-in operators are portable across frameworks. Every framework supporting ONNX will provide implementations of these operators on the applicable data types.

**Advantages of ONNX**

- Improvement in inference performance, inference time is considerably reduced.
- Reduced training time.
- It is useful for Transfer Learning.

## 6.3 Developing an Android Application

After the implementation of the pre trained model, the 3D models of the modern Tamil characters are imported to the asset folder and the classification is done by utilizing the pre trained model.

Then the request permission access for utilizing the android device camera is invoked on the installation of the application. It compares the classified output with the pre trained model, if the threshold value is greater than or equal to 80, it provides the classified model's equivalent modern character as output, else it iterates the classification process. The application is finally built using the AR Core Configuration which supports the application to run on an android device.

The application is stored in android application package format, with an extension of .apk file. It is then deployed in the mobile application to use it in real time.

# CHAPTER 7

## RESULT ANALYSIS

The Overall accuracy of our Android Application is 86.26% in recognizing the ancient stone inscription characters. The output is in the form of 3D Model of modern Tamil Characters in Film box (.fbx) format. The application is tested in Android Device with an API level of 28 (Android version 9 – "Pie") supported with ARCore Configuration.

The Application is tested with and without internet connection where both given the same accuracy. Thus our application can also be used in remote places where there is no internet bandwidth.

With all our observations we have made a confusion matrix which describes the performance of the classification model. The confusion matrix is given below.

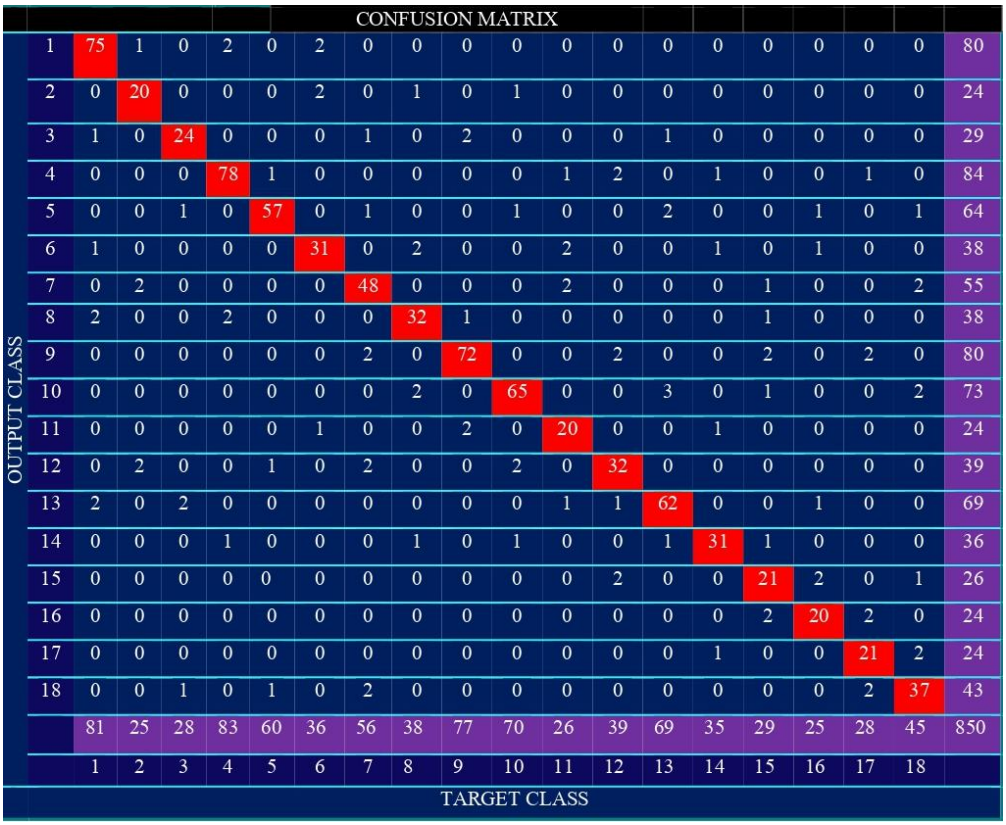



*Fig 7.1 Confusion Matrix*

The confusion matrix provides detailed information on how many letters are correctly identified in the data set, with the Diagonal data set and the total number of data in each character, the accuracy is identified for each character by dividing the data which is correctly identified with the total number of data collected.

The graph below gives the actual performance of our system after plotting the confusion matrix.

The Below graph gives average accuracy in detail for each data.

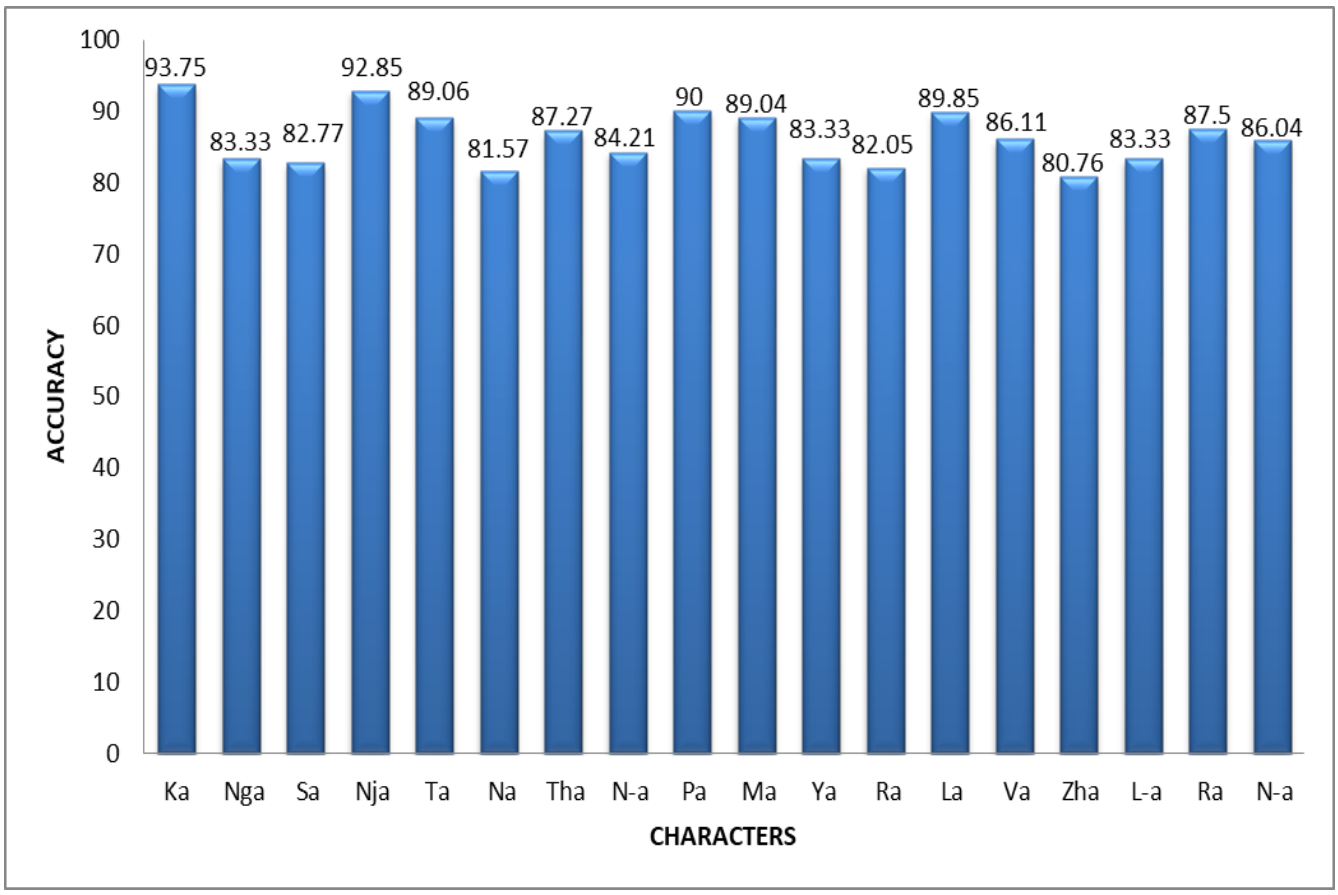



*Fig 7.2 Graph for the Accuracy Rate of each Character*

The overall accuracy rate is identified by summing the accuracy rate of all the character and dividing it with the total number of characters. The Below table provides a detailed information on result analysis of the Ancient Stone inscription Recognition.

*Table 7.1 Result Analysis*

| ANCIENT STONE CHARACTERS | CHARACTER | TAMIL LETTER | DATA COLLECTED | DATA IDENTIFIED CORRECTLY | ACCURACY RATE (in %) |
|---|---|---|---|---|---|
|  | Ka | க | 80 | 75 | 93.75 |
|  | Nga | ங | 24 | 20 | 83.33 |
|  | Sa | ச | 29 | 24 | 82.77 |
|  | Nja | ஞ | 84 | 78 | 92.85 |
|  | Ta | ட | 64 | 57 | 89.06 |
|  | Na | ண | 38 | 31 | 81.57 |
|  | Tha | த | 55 | 48 | 87.27 |
|  | N-a | ந | 38 | 32 | 84.21 |
|  | Pa | ப | 80 | 72 | 90 |

| | | | | | |
|---|---|---|---|---|---|
|  | Ma | ம | 73 | 65 | 89.04 |
|  | Ya | ய | 24 | 20 | 83.33 |
|  | Ra | ர | 39 | 32 | 82.05 |
|  | La | ல | 69 | 62 | 89.85 |
|  | Va | வ | 36 | 31 | 86.11 |
|  | Zha | ழ | 26 | 21 | 80.76 |
|  | La | ள | 24 | 20 | 83.33 |
|  | R-a | ற | 24 | 21 | 87.5 |
|  | Nna | ன | 43 | 37 | 86.04 |
| Average Accuracy | | | 86.26 | | |

Thus our system obtained 86.26% as its average accuracy for the recognition of ancient stone inscription characters.

# CHAPTER 8

## CONCLUSION AND FUTURE ENHANCEMENT

## CONCLUSION

Ancient Stone Character Recognition is very much useful in Archeology Department and for Tourism Purpose. It reduces the time the epigraphist takes to identify each and every character. This application is user friendly and cost- effective. This application produces more accuracy. It can also be used in Remote places as there is no need of any internet connection. This would be a satisfactory solution to everyone who is more curious to learn the ancient people lifestyle and their habits.

## FUTURE ENHANCEMENT

Every system has its own advantages and limitations as well. We have discussed all the pros in our system but this system also has some limitations which can be enhanced in future.

- This system only provides Recognition of limited characters from $11^{th}$ century Tamil Characters; it can be enhanced by training more data from other century characters.
- This system provides only the character recognition, it can be implemented with NLP and improvised to word recognition.
- The accuracy can be improved by collecting more number of data in each character.
- It can only be implemented in the Android Devices which support AR Core Configuration, this can be improvised to utilize in all the devices.
- It causes a time delay of 10seconds to identify the characters.

# APPENDICES

## APPENDIX 1

## CODE

### Training Model (Saved_Model.onnx)

```
#SPLIT FOLDER

!pip install split_folders
import splitfolders

input_folder='basedata/Input_dataset'
output='basedata/processed_data'
splitfolders.ratio(input_folder,output,seed=42,ratio=(.6,.2,.2))
help(splitfolders.ratio)

#TRAINING DATA SET

import tensorflow as tf
import matplotlib.pyplot as plt
import cv2
import os
import numpy as np

from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.layers import Conv2D, Flatten, Dense, MaxPool2D,
BatchNormalization, GlobalAveragePooling2D
from tensorflow.keras.applications.resnet50 import preprocess_input,
decode_predictions
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.preprocessing import image
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import Model

img_height, img_width=(224,224)
batch_size=32

train_data_dir=r"basedata\processed_data\train"
valid_data_dir=r"basedata\processed_data\val"
test_data_dir=r"basedata\processed_data\test"

train_datagen = ImageDataGenerator(preprocessing_function=preprocess_input,
                                   rotation_range=40,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=False,
                                   validation_split=0.4,
                                   brightness_range=(0.5,1.5))
train_generator = train_datagen.flow_from_directory(train_data_dir,
                                                    target_size=(img_height,img_width),
                                                    batch_size=batch_size,
                                                    class_mode='categorical',
                                                    subset='training') #set as training data
valid_generator=train_datagen.flow_from_directory(valid_data_dir,
                                                  target_size=(img_height,img_width),
                                                  batch_size=batch_size,
                                                  class_mode='categorical',
                                                  subset='validation') #set as validation data

test_generator= train_datagen.flow_from_directory(test_data_dir,
                                                  target_size=(img_height,img_width),
```

```
                                    batch_size=1,
                                    class_mode='categorical',
                                    subset='validation')

x,y=test_generator.next()
x.shape

train_generator.num_classes
base_model = ResNet50(include_top=False,weights='imagenet')

x=base_model.output
x=GlobalAveragePooling2D()(x)
x=Dense(1024,activation='relu')(x)
predictions=Dense(train_generator.num_classes, activation='softmax')(x)
model= Model(inputs=base_model.input, outputs=predictions)

for layer in base_model.layers:
    layer.trainable=False
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(train_generator, epochs=25)

model.save(r'basedata\Saved_Model\NeuralNetworks.h5')
test_loss,test_acc=model.evaluate(test_generator,verbose=2)
print(test_acc)
```

#CONFUSION MATRIX

```
import pandas as pd
import seaborn as sn
import tensorflow as tf
model=tf.keras.models.load_model(r"basedata\Saved_model\NeuralNetworks.h5")
filenames=test_generator.filenames
nb_samples=len(test_generator)
y_prob=[]
y_act=[]
test_generator.reset()
for _ in range(nb_samples):
    X_test,Y_test=test_generator.next()
    y_prob.append(model.predict(X_test))
    y_act.append(Y_test)
predicted_class=[list(train_generator.class_indices.keys())[i.argmax()]for i in y_prob]
actual_class=[list(train_generator.class_indices.keys())[i.argmax()]for i in y_act]
out_df=pd.DataFrame(np.vstack([predicted_class,actual_class]).T,columns=['predicted_
class','actual_class'])
confusion_matrix=pd.crosstab(out_df['actual_class'],out_df['predicted_class'],rownames
=['Actual'],colnames=['Predicted'])
sn.heatmap(confusion_matrix,cmap='Blues',annot=True,fmt='d')
plt.show()
print('test accuracy: {}'. format( (np.diagonal (confusion_matrix) .sum() /
confusion_matrix.sum().sum()*100)))
```

#SAVING IN ONNX FORMAT

```
import tensorflow as tf
from tensorflow.python.saved_model import builder as pb_builder
pre_model =
tf.keras.models.load_model(r"basedata\Saved_model\NeuralNetworks.h5")
pre_model.save(r"basedata\Saved_model\NeuralNetworks.h5")
# set learning phase to 0 since the model is already trained
tf.keras.backend.set_learning_phase(0)
#load the model
pre_model =
tf.keras.models.load_model(r"basedata\Saved_model\NeuralNetworks.h5")
#convert h5 to protobuffer
builder = pb_builder.SavedModelBuilder (r'basedata\Pb3')
builder.save()
builder.save(b'basedata\\Pb\\saved_model.bytes')
import tensorflow as tf
from tensorflow.keras.models import load_model
model=load_model(r"basedata\Saved_model\NeuralNetworks.h5")
model.summary()
#Convert the model to onnx format
import tensorflow as tf
import tf2onnx
import onnx
onnx_model, _ = tf2onnx.convert.from_keras(model, opset=9)
onnx.save(onnx_model, r"basedata\Saved_model\model.onnx")
```

**label.txt**

{

Ka, Nga, Sa, Nja, Ta , Na, Tha, N-a, Pa, Ma, Ya, Ra, La, Va, Zha, L-a, R-a, Nna

}

**Unity Package Importing (manifest.json)**


```
{
  "dependencies":
{
    "com.unity.collab-proxy": "1.15.16",
    "com.unity.ide.rider": "3.0.13",
    "com.unity.ide.visualstudio": "2.0.14",
    "com.unity.ide.vscode": "1.2.5",
    "com.unity.test-framework": "1.1.31",
    "com.unity.textmeshpro": "3.0.6",
    "com.unity.timeline": "1.6.4",
    "com.unity.ugui": "1.0.0",
    "com.unity.xr.arcore": "4.2.2",
    "com.unity.xr.arfoundation": "4.2.2",
    "com.unity.xr.arkit": "4.2.2",
    "com.unity.modules.ai": "1.0.0",
    "com.unity.modules.androidjni": "1.0.0",
    "com.unity.modules.animation": "1.0.0",
    "com.unity.modules.assetbundle": "1.0.0",
    "com.unity.modules.audio": "1.0.0",
    "com.unity.modules.cloth": "1.0.0",
    "com.unity.modules.director": "1.0.0",
    "com.unity.modules.imageconversion": "1.0.0",
    "com.unity.modules.imgui": "1.0.0",
    "com.unity.modules.jsonserialize": "1.0.0",
```

```
    "com.unity.modules.particlesystem": "1.0.0",
    "com.unity.modules.physics": "1.0.0",
    "com.unity.modules.physics2d": "1.0.0",
    "com.unity.modules.screencapture": "1.0.0",
    "com.unity.modules.terrain": "1.0.0",
    "com.unity.modules.terrainphysics": "1.0.0",
    "com.unity.modules.tilemap": "1.0.0",
    "com.unity.modules.ui": "1.0.0",
    "com.unity.modules.uielements": "1.0.0",
    "com.unity.modules.umbra": "1.0.0",
    "com.unity.modules.unityanalytics": "1.0.0",
    "com.unity.modules.unitywebrequest": "1.0.0",
    "com.unity.modules.unitywebrequestassetbundle": "1.0.0",
    "com.unity.modules.unitywebrequestaudio": "1.0.0",
    "com.unity.modules.unitywebrequesttexture": "1.0.0",
    "com.unity.modules.unitywebrequestwww": "1.0.0",
    "com.unity.modules.vehicles": "1.0.0",
    "com.unity.modules.video": "1.0.0",
    "com.unity.modules.vr": "1.0.0",
    "com.unity.modules.wind": "1.0.0",
    "com.unity.modules.xr": "1.0.0",
    "com.unity.barracuda":"1.0.4"
  }
}
```

**Unity Package Code (Package.json)**

```
{
  "dependencies": {
    "com.unity.collab-proxy": {
      "version": "1.15.16",
      "depth": 0,
      "source": "registry",
      "dependencies": {
        "com.unity.services.core": "1.0.1"  },
      "url": "https://packages.unity.com" },
    "com.unity.editorcoroutines": {
      "version": "1.0.0",
      "depth": 1,
      "source": "registry",
      "dependencies": {},
      "url": "https://packages.unity.com"    },
    "com.unity.ext.nunit": {
      "version": "1.0.6",
      "depth": 1,
      "source": "registry",
      "dependencies": {},
      "url": "https://packages.unity.com"
    },
    "com.unity.ide.rider": {
      "version": "3.0.13",
      "depth": 0,
      "source": "registry",
      "dependencies": {
        "com.unity.ext.nunit": "1.0.6" },
      "url": "https://packages.unity.com" },
```

```
"com.unity.ide.visualstudio": {
  "version": "2.0.14",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.test-framework": "1.1.9" },
  "url": "https://packages.unity.com"},
"com.unity.ide.vscode": {
  "version": "1.2.5",
  "depth": 0,
  "source": "registry",
  "dependencies": {},
  "url": "https://packages.unity.com" },
"com.unity.nuget.newtonsoft-json": {
  "version": "3.0.2",
  "depth": 2,
  "source": "registry",
  "dependencies": {},
  "url": "https://packages.unity.com" },
"com.unity.services.core": {
  "version": "1.3.1",
  "depth": 1,
  "source": "registry",

    "dependencies": {
    "com.unity.modules.unitywebrequest": "1.0.0",
    "com.unity.nuget.newtonsoft-json": "3.0.2",
    "com.unity.modules.androidjni": "1.0.0" },
  "url": "https://packages.unity.com"  },
"com.unity.subsystemregistration": {
```

```
  "version": "1.1.0",
  "depth": 2,
  "source": "registry",
  "dependencies": {
    "com.unity.modules.subsystems": "1.0.0" },
  "url": "https://packages.unity.com" },
"com.unity.test-framework": {
  "version": "1.1.31",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.ext.nunit": "1.0.6",
    "com.unity.modules.imgui": "1.0.0",
    "com.unity.modules.jsonserialize": "1.0.0" },
  "url": "https://packages.unity.com"},
"com.unity.textmeshpro": {
  "version": "3.0.6",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.ugui": "1.0.0" },
  "url": "https://packages.unity.com" },
"com.unity.timeline": {
  "version": "1.6.4",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.modules.director": "1.0.0",
    "com.unity.modules.animation": "1.0.0",
    "com.unity.modules.audio": "1.0.0",
```

```
    "com.unity.modules.particlesystem": "1.0.0" },
  "url": "https://packages.unity.com" },
"com.unity.ugui": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.ui": "1.0.0",
    "com.unity.modules.imgui": "1.0.0"}
},
"com.unity.xr.arcore": {
  "version": "4.2.2",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.xr.arsubsystems": "4.2.2",
    "com.unity.xr.management": "4.0.1",
    "com.unity.modules.androidjni": "1.0.0",
    "com.unity.modules.unitywebrequest": "1.0.0"      },
  "url": "https://packages.unity.com" },
"com.unity.xr.arfoundation": {
  "version": "4.2.2",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.xr.arsubsystems": "4.2.2",
    "com.unity.xr.management": "4.0.1",
    "com.unity.modules.particlesystem": "1.0.0"
  },
  "url": "https://packages.unity.com"    },
```

```
"com.unity.xr.arkit": {
  "version": "4.2.2",
  "depth": 0,
  "source": "registry",
  "dependencies": {
    "com.unity.editorcoroutines": "1.0.0",
    "com.unity.xr.arsubsystems": "4.2.2",
    "com.unity.xr.management": "4.0.1"      },
  "url": "https://packages.unity.com" },
"com.unity.xr.arsubsystems": {
  "version": "4.2.2",
  "depth": 1,
  "source": "registry",
  "dependencies": {
    "com.unity.subsystemregistration": "1.1.0",
    "com.unity.xr.management": "4.0.1" },
  "url": "https://packages.unity.com" },
"com.unity.xr.legacyinputhelpers": {
  "version": "2.1.9",
  "depth": 2,
  "source": "registry",
  "dependencies": {
    "com.unity.modules.vr": "1.0.0",
    "com.unity.modules.xr": "1.0.0"},
  "url": "https://packages.unity.com" },
"com.unity.xr.management": {
  "version": "4.2.0",
  "depth": 1,
  "source": "registry",
  "dependencies": {
```

```
    "com.unity.modules.subsystems": "1.0.0",
    "com.unity.modules.vr": "1.0.0",
    "com.unity.modules.xr": "1.0.0",
    "com.unity.xr.legacyinputhelpers": "2.1.7",
    "com.unity.subsystemregistration": "1.0.6" },
  "url": "https://packages.unity.com" },
"com.unity.modules.ai": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.androidjni": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.animation": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.assetbundle": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {}},
"com.unity.modules.audio": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
```

```
  "dependencies": {} },
"com.unity.modules.cloth": {
"version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.physics": "1.0.0"  }},
"com.unity.modules.director": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.audio": "1.0.0",
    "com.unity.modules.animation": "1.0.0" } },
"com.unity.modules.imageconversion": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.imgui": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.jsonserialize": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.particlesystem": {
```

```
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.physics": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.physics2d":
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {}  },
"com.unity.modules.screencapture": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.imageconversion": "1.0.0" }
},
"com.unity.modules.subsystems": {
  "version": "1.0.0",
  "depth": 1,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.jsonserialize": "1.0.0" }
},
"com.unity.modules.terrain": {
  "version": "1.0.0",
```

```
  "depth": 0,
  "source": "builtin",
  "dependencies": {}
},
"com.unity.modules.terrainphysics": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.physics": "1.0.0",
    "com.unity.modules.terrain": "1.0.0" }
},
"com.unity.modules.tilemap": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.physics2d": "1.0.0"  }
},
"com.unity.modules.ui": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {}
},
"com.unity.modules.uielements": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
```

```
    "com.unity.modules.ui": "1.0.0",
    "com.unity.modules.imgui": "1.0.0",
    "com.unity.modules.jsonserialize": "1.0.0",
    "com.unity.modules.uielementsnative": "1.0.0" }
},
"com.unity.modules.uielementsnative":{
  "version": "1.0.0",
  "depth": 1,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.ui": "1.0.0",
    "com.unity.modules.imgui": "1.0.0",
    "com.unity.modules.jsonserialize": "1.0.0" }
},
"com.unity.modules.umbra": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {} },
"com.unity.modules.unityanalytics": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.unitywebrequest": "1.0.0",
    "com.unity.modules.jsonserialize": "1.0.0" }
},
"com.unity.modules.unitywebrequest": {
  "version": "1.0.0",
  "depth": 0,
```

```
  "source": "builtin",
  "dependencies": {}
},
"com.unity.modules.unitywebrequestassetbundle": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.assetbundle": "1.0.0",
    "com.unity.modules.unitywebrequest": "1.0.0" }
},
"com.unity.modules.unitywebrequestaudio": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.unitywebrequest": "1.0.0",
    "com.unity.modules.audio": "1.0.0" }
},
"com.unity.modules.unitywebrequesttexture": {
  "version": "1.0.0",
  "depth": 0,
  "source": "builtin",
  "dependencies": {
    "com.unity.modules.unitywebrequest": "1.0.0",
    "com.unity.modules.imageconversion": "1.0.0" }
},
"com.unity.modules.unitywebrequestwww": {
  "version": "1.0.0",
  "depth": 0,
```

```
    "source": "builtin",
    "dependencies": {
      "com.unity.modules.unitywebrequest": "1.0.0",
      "com.unity.modules.unitywebrequestassetbundle": "1.0.0",
      "com.unity.modules.unitywebrequestaudio": "1.0.0",
      "com.unity.modules.audio": "1.0.0",
      "com.unity.modules.assetbundle": "1.0.0",
      "com.unity.modules.imageconversion": "1.0.0" }  },
  "com.unity.modules.vehicles": {
    "version": "1.0.0",
    "depth": 0,
    "source": "builtin",
    "dependencies": {
      "com.unity.modules.physics": "1.0.0" }    },
  "com.unity.modules.video": {
    "version": "1.0.0",
    "depth": 0,
    "source": "builtin",
    "dependencies": {
      "com.unity.modules.audio": "1.0.0",
      "com.unity.modules.ui": "1.0.0",
      "com.unity.modules.unitywebrequest": "1.0.0"}  },
  "com.unity.modules.vr": {
    "version": "1.0.0",
    "depth": 0,
    "source": "builtin",
    "dependencies": {
      "com.unity.modules.jsonserialize": "1.0.0",
      "com.unity.modules.physics": "1.0.0",
      "com.unity.modules.xr": "1.0.0" }  },
```

```
    "com.unity.modules.wind": {
      "version": "1.0.0",
      "depth": 0,
      "source": "builtin",
      "dependencies": {} },
    "com.unity.modules.xr": {
      "version": "1.0.0",
      "depth": 0,
      "source": "builtin",
      "dependencies": {
        "com.unity.modules.physics": "1.0.0",
        "com.unity.modules.jsonserialize": "1.0.0",
        "com.unity.modules.subsystems": "1.0.0"
      }    }  }
}
```

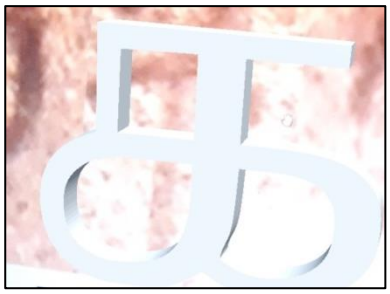
**Unity Classification Code (ImageClassifier.cs)**

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR;
using UnityEngine.XR.ARFoundation;
[RequireComponent(typeof(ARTrackedImageManager))]

public class ImageClassifier : MonoBehaviour{
    [SerializeField]
    private GameObject[] placablePrefabs;
    private Dictionary<string, GameObject> spawnedPrefabs = new Dictionary<string,
GameObject>();
    private ARTrackedImageManager trackedImageManager;
    private void Awake()    {
```

```
        trackedImageManager = FindObjectOfType<ARTrackedImageManager>();
        foreach(GameObject prefab in placablePrefabs)   {
            GameObject newPrefab = Instantiate(prefab, Vector3.zero, Quaternion.identity);
            newPrefab.name = prefab.name;
            spawnedPrefabs.Add(prefab.name, newPrefab);    }
    }
    private void OnEnable()  {
        trackedImageManager.trackedImagesChanged += ImageChanged;
    }
    private void ImageChanged(ARTrackedImagesChangedEventArgs eventArgs) {
        foreach(ARTrackedImage trackedImage in eventArgs.added)  {
            UpdateImage(trackedImage);        }
        foreach (ARTrackedImage trackedImage in eventArgs.updated) {
            UpdateImage(trackedImage);        }
        foreach (ARTrackedImage trackedImage in eventArgs.removed) {
            spawnedPrefabs[trackedImage.name].SetActive(false);        }    }
    private void UpdateImage(ARTrackedImage trackedImage)
     {
        string name = trackedImage.referenceImage.name;
        Vector3 position = trackedImage.transform.position;
        GameObject prefab = spawnedPrefabs[name];
        prefab.transform.position = position;
        prefab.SetActive(true);
        foreach(GameObject go in spawnedPrefabs.Values)
       {
            if (go.name != name)  {
                go.SetActive(false);  }
        }
    }
}
```

# APPENDIX 2 – SCREENSHOTS

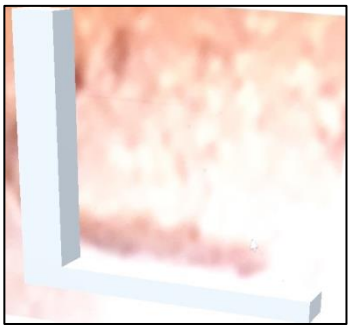


Ka – “க”

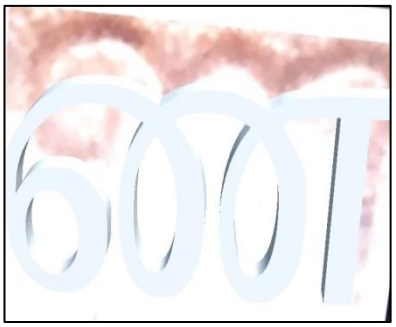
Nga – “ங”

Sa – “ச”

Nja – “ஞ”



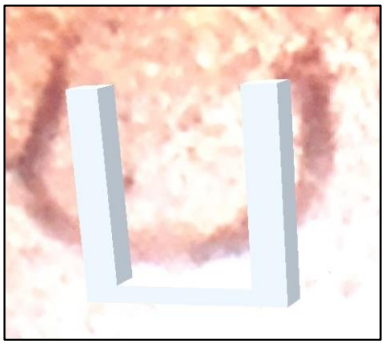
Ta – “ட”



Na – “ண”

Tha – “த”

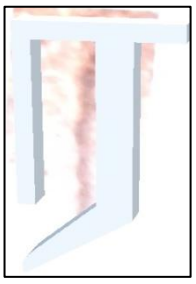
N-a – “ந”



Pa - “ப”

Ma – “ம”

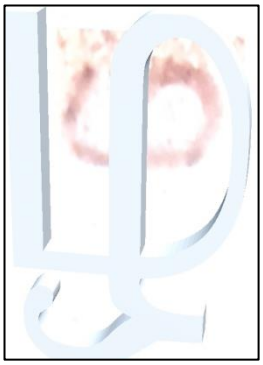
Ya – “ய”



Ra – “ர”

La – “ல”

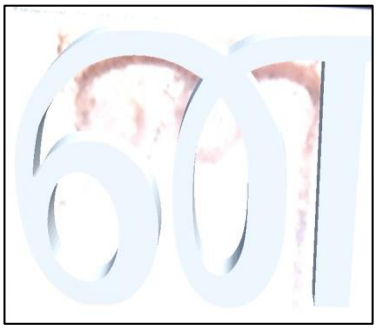
Va – “வ”



Zha – “ழ”

L-a – “ள”

R-a – “ற”



Nna – “ன”

*Fig 9.1 Screenshots of the output*

The above figures provide the screenshots of the scanned ancient Tamil Characters which is instantiated by the Modern Tamil Characters.

# REFERENCES

[1] G. Bhuvaneswari ; Dr. V.Subbiah Bharathi, “Recognition of Ancient Stone Inscription Characters using Normalized Positional Distance Metric Features”,Asian Journal of Research in Social Sciences and Humanities Vol. 6, No. 5, May 2016, pp. 604-615. ISSN 2249-7315

[2] Dr.G.Bhuvaneswari, Dr.G.Manikandan “Recognition of Ancient Stone Inscription Characters Using Histogram of Oriented Gradients”, International Conference on Recent Trends in Computing, Communication and Networking Technologies (ICRTCCNT’19) Oct 18-19, 2019.

[3] T Manigandan, V Vidhya, V Dhanalakshmi, B Nirmala “Tamil character recognition from ancient epigraphical inscription using OCR and NLP”, 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)

[4] Faustina Jeya Rose.R, Bhuvaneswari.G, “Word Recognition Incorporating Augmented Reality For Linguistic E-Conversion”, International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) – 2016.

[5] Giridharan.R, Vellingiriraj.E.K, Dr. Balasubramanie “Identification of Tamil Ancient Characters and Information Retrieval from Temple Epigraphy Using Image Zoning”, IEEE 2016 Fifth International Conference on Recent Trends in Information Technology (ICRTIT).

[6] G. Janani, V. Vishalini, P. Mohan Kumar “Recognition and analysis of Tamil inscriptions and mapping using image processing techniques” 2016 Second International Conference on Science Technology Engineering and Management (ICONSTEM).

[7] S. RajaKumar, V. Subbiah Bharathi, “Eighth century tamil consonants recognition from stone inscriptions” 2012 International Conference on Recent Trends in Information Technology.

[8] Y.Liang, M.C.Fairhurst and R.M.Guest, “A Synthesisd Word Approach to Word Retrieval in Handwritten Documents”, International Tamil Internet Conference Proceedings, Vol.45, Jun 2012, PP 4225-4236.

[9] Giuseppe Pirlo, DonatoImpedovo, “Adaptive Membership Functions for Handwritten Character Recognition by Voronoi-Based Image Zoning”, IEEE Trans on Image Processing, Vol 21, No 9, Sep 2012,pp. 3827-3836

[10] Bharath A, SriganeshMadhvanath, “Hidden Markov Models for Online Handwritten Tamil Word Recognition.” Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), HPL-2007-108, July 6, 2007

[11] F. Shafait, D. Keysers, and T. M. Breuel, “Performance Evaluation and Benchmarking of Six-Page Segmentation Algorithms,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 6, Jun 2008, pp. 941–954.

[12] Chomtip Pornpanomchai,Verachag Wongsawangtham, Satheanpong Jeungudomporn and NannaphatChatsumpun, “Thai Handwritten Character Recognition by Genetic Algorithm (THCRGA)”, IACSIT Journal of Engineering and Technology, Vol 3, No 2, Apr 2011.

[13] A Bharath and Sriganesh Madhvanath, “HMM-Based Lexicon-Driven and Lexicon-Free Word Recognition for Online Handwritten Indic Scripts”, IEEE Trans on Pattern Analysis and Machine Intelligence,Vol 34, No 4, Apr 2012.

[14] Chomtip Pornpanomchai, Dentcho N. Batanov and Nicholas Dimmitt, “Recognizing Thai handwritten characters and words for human computer interaction”, International Journal of Human-Computer Studies, 2001, pp 259-279.

[15] Chomtip Pornpanomchai, Pattara Panyasrivarom, Nuttakit Pisitviroj and Piyaphume Prutkraiwat, “Thai Handwritten Character Recognition by Euclidean Distance”, The 2nd International Conference on Digital Image Processing (ICDIP 2010), 2010, pp 53-58.

[16] SeethalakshmiR., Sreeranjani T.R., Balachandar T., Abnikant Singh, Markandey Singh, RitwajRatan, Sarvesh Kumar, "Optical Character Recognition for printedTamil text using Unicode", Journal of Zhejiang University SCIENCE, Vol. 6A No. 11, 2005.

[17] C. S. Sundaresan and S. S. Keerthi, “A Study of Representations for Pen based Handwriting Recognition of Tamil Characters”, Proceedings of the 5th International Conference onDocument Analysis and Recognition, 1999.

[18] Boontee Kruatrachue, Nattachat Pantrakarn and Kritawan Siriboon “State Machine Induction with Positive and Negative for Thai Character Recognition”, The International Conference on Communications, Circuits and Systems, 2007, pp 971-975.

[19] Parinya Sanguansat, Widhyakorn Asdorn wised and Somchai Jitapunkul, “Online Thai Handwritten Character Recognition Using Hidden Markov Models and Support Vector Machines”, The International Symposium on Communications and Information Technologies, 2004, pp.492-497.