EAI Endorsed Transactions
on Cloud Systems
Research Article EAI.EU

Intelligent Character Recognition System Using Convolutional Neural Network

S. Suriya[1,*], Dhivya S[2] and Balaji M[3]

[1]Associate Professor, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, India.
suriyas84@gmail.com , ss.cse@psgtech.ac.in
[2]PG Scholar, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, India.
dhivyasritas@gmail.com
[3]Software Engineer, Arcesium, balajimuthazhagan@gmail.com

Abstract

Computational Linguistics involves the techniques of Computer Science which play a vital role in recognizing written or printed characters such as numbers or letters to change them into a form that the computer can use it efficiently. Convolutional Neural Network differs from other approaches by extracting the features automatically. The proposed approach is capable of recognizing characters in a variety of challenging conditions using the Convolutional Neural Network, where traditional character recognition systems fail, notably in the presence of low resolution, substantial blur, low contrast, and other distortions. Intellectual Character Recognition System is an application that uses Convolutional Neural Network (CNN) to recognize the Tamil character dataset accurately developed by HP Labs India. The novelty of this system is that, it recognizes the characters of the Predominant Tamil language. With the help of suitable datasets consisting of the Tamil Scripts, the model is trained efficiently. This work has produced a training accuracy of 99.16% which is far better compared to the traditional approaches.

Keywords: Computational Linguistics, Character Recognition, Convolutional Neural Networks, Machine Learning.

Received on 10 May 2020, accepted on 19 September 2020, published on 16 October 2020

doi: 10.4108/eai.16-10-2020.166659

1. Introduction

Recognition of handwritten characters by machine is becoming more and more relevant in the modern world. Handwritten record acknowledgment is the capacity of a computer to get and decipher clear manually written information. The purpose of this project is to take Tamil handwritten characters as input and to recognize the character. The major challenges, as in the case of any handwritten character recognition problem, is the large variation in the writing styles of the individual at different times and among various people, for example, size shape, speed of composing and thickness of characters, and so on. The problem of printed character recognition is relatively well understood and solved with little constraints and available system yield as good as maximum recognition accuracy. But transcribed character acknowledgment frameworks have still restricted abilities. Other difficulties include the similarity of some characters with each other, an infinite variety of character shapes, etc. Handwritten Character Recognition is one of the most popular areas of research in pattern recognition because of its immense application potential. Handwritten character recognition is relatively matured in languages like English, Chinese, Korea, Japanese, and Arabic. In Indian languages, studies are active in Devanagari and Bangla. Some promising research findings are reported in south Indian (Dravidian) languages like Telugu and Kannada. In Tamil, though many research papers are published in this area, the results reported are inadequate for the design of efficient

*Corresponding author. Email: suriyas84@gmail.com

Handwritten Character Recognition systems. This is the motivation behind this thesis.
Linguistic is the scientific study of human language, the main aim is to establish a theory by studying nature of the language and by applying the established theory to describe other languages. It involves analyzing language form, language meaning, and language in context. Linguists traditionally analyze human language by observing interplay between sound and meaning. Computational linguistics (CL) is the application of computer science to the analysis, synthesis and comprehension of written and spoken language. It is an interdisciplinary field dealing with the statistical and logical modeling of natural language from a computational perspective. Computational linguistics is used in instant machine translation, speech recognition (SR) systems, text-to-speech (TTS) synthesizers, interactive voice response (IVR) systems, search engines, text editors and language instruction materials. The interdisciplinary field of study requires expertise in machine learning (ML), deep learning (DL), artificial intelligence (AI), cognitive computing and neuroscience. The main goal of CL is to make the computers understand human language since computer language doesn't match structure of human thought. Computational Linguistics is closely related to Natural Language Technology, Natural Language Engineering, Natural Language Processing and Artificial Intelligence. A computational understanding of language provides human beings with insight into thinking and intelligence. Computers that are linguistically competent not only help facilitate human interaction with machines and software, but also make the textual and other resources of the internet readily available in multiple languages.

2. Literature Survey

Nicole Dalia Cilia et al. [1] proposed A ranking-based feature selection approach for handwritten character recognition using an new selection approach for feature extraction. Its aim is to reduce the computational cost of the classification task, in an attempt to increase, or not to reduce, the classification performance. There were several limitations, which include the computational complexity, the dependence on the adopted classifiers and the difficulty in evaluating the interactions among features. In this approach several drawbacks were overcome by adopting feature-ranking-based techniques and different univariate measures were introduced to produce a feature ranking. Greedy search approaches were proposed for choosing the feature subset able to maximize the classification results. The computational cost of the procedures was high for searching effective feature subsets, and the difficulties were faced to take into account the effects of the interactions among features. In order to eradicate these drawbacks ranking techniques for feature extractions has been combined using greedy approaches to select feature subsets with an increasing number of features, obtained by adding features progressively according to their position in the ranking. By significantly reducing the computational complexity of the whole recognition system with very limited effects on the classification performance with the error rate of 21%. The result of this work shows that it is possible to reduce significantly the number of features, and consequently the complexity of the classification tasks, accepting a limited reduction of the recognition rate.

Manpreet Kaur et al. [2] introduced Proposed Approach for Layout & Handwritten Character Recognition for recognizing character from the datasets which contains heterogeneous data. Radial Sector Coding algorithm has been used in this approach for the detection of arbitrarily oriented text in an image. Information energy approach is being used to segment the lines that can be embedded. To achieve rotation invariance was a challenging task in this approach and was achieved it by finding Axis of Reference which is a rotation invariant feature for all characters then the Line of Reference from
Axis of Reference which is considered as 0 lines for feature generation and thus generated Translation, Rotation and Scale invariant features. The 26 uppercase English characters were used for performance evaluation of RSC and different rotated scaled versions of Arial and Tahoma fonts for each character were used. The two sets of characters were used one set for training of artificial neural network and other set is used for performance test and the imposed algorithm is implemented in suitable environment of MATLAB. This method applied on the document which contains the three type of language Arabic, English, French is in the form of printed text and handwritten text. The first step in this preprocessing step is used for the noise removal of small connected component (cc). In the second step the text & non text classification is done on based of learning based approach on the basis of cc and its neighborhood to the feed an MLP classifier. Third step is layer separation which is used to classify the typed text and handwritten on the behalf of code book method. The fourth step is block segmentation. In this first step include applying RLSA algorithm to connect close cc and the second step is segment the document by white space to filter the small rectangle. The main motive is to make this approach user specific so that the shopkeeper or the other person who is not so computer friendly store and analysis their bills.More enhanced techniques were applied on the images for better analysis of the heterogeneous images.

Yongchun Zhu et al. [3] proposed an Adaptively Transfer Category-Classifier for Handwritten Chinese Character Recognition and introduced a new neural network structure for Handwritten Chinese Character Recognition (HCCR) to make full use of a large amount of labeled source domain data and a small number of target domain data to learn the model parameters. Furthermore transfer on the category-classifier level, and adaptively assign different weights were used to category-classifiers according to the usefulness of source domain data. The experiments were constructed from three data sets demonstrate the effectiveness of the model compared with several state-of-

the-art baselines. Transfer learning aims to adapt the knowledge from related source domain data to the model learning in the target domain, which provides the possibility of success for HCCR tasks, a transfer handwritten Chinese character recognition model has been developed based on the successful deep network structure AlexNet. Specifically, for both source and target domain data, the network parameters shared with five convolution layers and three pooling layers, and then learn the parameters of three fully connected layers separately. In addition, to adaptively transfer the category knowledge from the source domain to the target domain, a regularization item were imposed with different weights to learn the similarity of category-classifiers trained from the source to the target domain and extensive experiments on three data sets to validate the effectiveness of our model was calculated finally. The approach was tested on the HCL2000, CASIA-HWDB1.1 and MSS-HCC datasets. This dataset contains 6600 shapes of handwritten characters written by 50 persons. The dataset is divided into a training set of 5280 images and a test set of 1320 images. The result was promising with an error classification rate of 2.1%.

Raymond Ptucha et al. [4] proposed Character recognition using fully convolutional neural networks, the preprocessing step in this approach normalizes the input blocks to a canonical representation which negates the need for costly recurrent symbol alignment correction. Character based classification is implied without relying on predefined dictionaries or contextual information. This work focuses on extracted handwritten symbol blocks and the input is defined as a tightly cropped gray scale image of an arbitrary 1D sequence of symbols, the word symbol emphasizes the model which was not limited to Latin based characters, neither spaces between characters, nor the need to predict space. Convolution filters alternately align on symbol and blanks, it can be seen perfect alignment requires each symbol to be of identical width. Wider filters ensure the complete symbol is in the receptive field as the filter steps across the word block. The RIMES dataset were used which contains 60,000 French words, by over 1000 authors. There are several versions of the RIMES dataset, where each newer release is a super-set of prior releases. Generically trained models, while not as good as those fine-tuned for a particular dataset performed quite well. The size of the lexicon used by the Lexicon CNN to be application dependent and comparatively Lexicon CNN to less than 2000 words as the fully convolutional Symbol Prediction FCN is performing very well. This work demonstrates how to replace recurrent neural networks with fully convolutional methods when processing variable length temporal streams of offline handwriting imagery. These streams are firstly broken into their constituent parts, where each part is measured in length and resampled to a canonical representation compatible with a fully convolutional network. This divide and conquer fully convolutional approach is input length agnostic, and does not suffer from exploding or vanishing gradients.

ThirumalaiMurugan et al. [5] propose A Novel Approaches to Handwritten English Character Alphabet and Number Recognition Based on Neural Networks, due to variation in shape, slope and size of individual characters they are handled by better pre-processing and feature extraction techniques. The Feed Forward Algorithm gives insight into the enter workings of a neural network; followed by the Back Propagation Algorithm which compromises Training and Testing. The performances of Handwritten English Character Alphabet and Number Recognition Based on Neural Networks by using Geometric Feature Extraction and Gradient Technique. The Edge Detection Algorithm was used which has a list called traverse list. It is the list of pixel already traversed by the algorithm. A geometry based technique for feature extraction applicable to segmentation-based word recognition systems. The proposed system extracts the geometric features of the character contour. This feature is based on the basic line types that form the character skeleton. The system gives a feature vector as its output. The feature vectors the generated from a training set were then used to train a pattern recognition engine based on Neural Networks so that the system. A gradient feature vector is composed of the strength of gradient accumulated separately in different directions. In Neural Network, each node perform some simple computation and each connection conveys a signal from one node to another labeled by a number and extended to which signal is amplified. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined. Networks with biases, a sigmoid layer, and a linear output layer are capable of approximating any function with a finite number of discontinuities.T he result shows that the back propagation network provides good recognition accuracy of more than 90% of handwritten English characters.

Muna Ahmed Awel et al. [6] imposed a Review on Optical Character Recognition, this papers is on review of some researches has been made in English, Arabic and Devanagari characters and the methodology used and challenges faced during development of Optical character recognition. For feature extraction global geometrics and geometric density classifier were used and the evaluation of the system has achieved for Geometric Density 77.89% and Geometric Feature 76.44% accuracy rate. Support vector system (SVM) is selected for recognition and the method recognized for the small dataset was of 86% accuracy. From the review of related papers the major steps used was preprocessing, segmentation, feature extraction and post processing. Challenges faced during recognition process were Scene Complexity, Conditions of Uneven Lighting, Skewness (Rotation), Blurring and Degradation, Fonts and style and Multilingual Environments. In the research works revised in this paper, character recognition

system use different approaches and many of them get good accuracy and the feature extraction techniques should be choose according to the character you working because each scripts or alphabets has its own nature therefor need to find techniques which fit or suitable for characters. The better able to extract features from character more we can detect and recognize characters in highest accuracy.

AnupamGarg et al. [7] proposed Offline handwritten Gurmukhi character recognition: k-NN vs. SVM classifier to analyze the impact of combination of feature extraction and classification techniques. Also principal component analysis (PCA) has been used to find efficient features from peak extent based and modified division point (MDP) based features which have further been used in the classification process. For classification, k-NN and SVM whose three different kernels, namely, linear-SVM, polynomial-SVM and RBFSVM have been considered for recognition accuracy in this paper. For selecting the training and testing dataset, five different partitioning strategies and k-fold cross validation techniques have been used. The experimentation has been performed on the dataset of 8960 samples of offline handwritten Gurmukhi characters written by 160 unique writers.

The proposed framework consists of the stages, namely, digitization, pre-processing, feature extraction, and classification. Digitization is the process of converting the paper based handwritten document into electronic format. Digitization produces the digital image, which is fed to the pre-processing phase. During pre-processing, we have converted the digitized image to thinned image (stroke width single pixel).The partitioning strategy and k-fold cross validation technique for selecting the training and the testing patterns have been experimented in this work. The classifiers that have been employed in this work are k-NN, Linear-SVM, Polynomial-SVM and RBF-SVM and combinations of this using database partitioning strategy and five-fold cross validation. A recognition accuracy of 92.3% has been achieved, using the combination of linear-SVM, polynomial-SVM and k-NN classifiers and with the partitioning strategy of 80% data as training dataset and remaining data as testing dataset.

AriyonoSetiawan et al. [8] proposed Handwriting Character Recognition Javanese Letters Based on Artificial Neural Network; Javanese character is one of Indonesia's cultural heritages that must be preserved. Manuscript form of the Javanese character is one of the priceless inheritances. Javanese characters are often called the Hanacaraka font. Back propagation was used which is a learning method that is usually used by perceptron with many layers to change the weights associated with neurons in the hidden layer. Back propagation method uses its error output to change its weighted value in back-forward. To get this error, the feed-forward stage must be performed. The Scanned images are transformed into grayscale images the process will detect the edge of each character that exists using the Canny algorithm. After getting the edges, the process of threshold the image were turned into black and white form and thickening the edges of the image was done to reinforce the shape of the image that has been threshold. Image data used are 200 pieces for the training process and 100 for the testing process. The details are for training each of the 10 characters, while for testing as many as 5 each. So the total data used is 300 images. From the training results obtained by running the system as much as 6 times obtained from the total number of outputs that match the system or have reached the target of 179 characters. Testing data is taken from several people with different handwriting. In this system the iteration is 100000 that means the training process stops at iteration 100000 with an error target close to 0 for each character trained. From the test results it was shown that there are some images that cannot reach the target. The image did not reach the target because there are similar values between the script pattern and each other. There are some images that have a shape that is somewhat similar so that the characters are not recognized. The trial results obtained an average accuracy of 74%.

Tianwei Wang et al. [9] introduced Radical aggregation network for few-shot offline handwritten Chinese character recognition, a novel radical aggregation network (RAN) is proposed for few-shot/zero-shot offline handwritten Chinese character recognition. The RAN comprises of three segments, an extreme mapping encoder (RME), an extreme conglomeration module (RAM), and a character investigation decoder (CAD). Zero-shot refers to the classifier which is trained with limited Chinese character classes, containing all radicals; and the recognition is then performed on the unseen classes. Few-shot means that few support samples of the unseen classes are also added for training. Compared to the large-scale and ever-increasing characters in Chinese languages, approximately 1000 radicals can be used to compose over 10,000 characters. All of Chinese characters can be decomposed into a unique radical string. Regardless of the enormous number of Chinese characters, current best in class techniques accomplished radical-based Chinese character zero-shot acknowledgment with encoder-decoder engineering. The radical aggregation module (RAM) conducts a distance metric between the radical representation and radical prototypes; it then aggregates radicals to its own prototypes while distancing it from others. The character analysis decoder (CAD) analyses the radical representations sequentially and transcripts them into character. Compared to handwritten Chinese character samples, printed Chinese character samples are much easier to obtain. The new methodology RAN were used , which introduces a distance metric criterion for radical features to improve its robustness, and integrates an end-to-end decoding strategy with few support samples used for training and recognized good results with minimum error rate.

Ashlin Deepa et al. [10] proposed A novel nearest interest point classifier for Tamil handwritten character recognition, a study made on image to image matching is included through feature analysis without using machine learning approaches. The main concept of proposed

method is making local-level decision on the class of the test image based on individual features called IPs of the training images. The global decision on the class of the test image is obtained after processing these local decisions. NIP threshold value were found to be greater than the probability, the matching pair of IPs and the corresponding distances with training images were also found using these values. This process gives local decision votes for NIPs. The classifier which classifies the IPs based on NIP is termed as NIP classifier. The proposed NIP classifier was compared with several classifiers by providing standard dataset. The computational complexity of k-means clustering is O(n*k*I*d) where n is the number of samples, k is the number of clusters, I is the number of iterations and d is the number of features. The procedure was repeated till the classification error decreased to some quantity, leading to an increase in computation complexity. A set of groups of character classes are produced as the output of stage. The proposed NIP classifier introduces the concept of local feature decision in the phase, and later global decision is taken to make a conclusion on the class of the test character image. Without compromising on the recognition accuracy, the proposed method simplifies the use of variable length as well as high dimensional feature vector which are measured as major issues of HCR systems. Benchmark database is used to demonstrate robust recognition performance. The state-of-the-art performance is achieved as NIP classifier calculates collective class similarity voting and thus reducing false recognitions. In contrast from conventional classification approaches, which depend on feature reduction methods, the result presented in this paper proves that the proposed classifier can successfully operate on the greater availability of features in the problems with high dimensional images.

Nibaran Das et al. [11] introduced Multiobjective optimization for recognition of isolated handwritten Indic scripts, the efficient region sampling for identifying the most informative local region were identified. The local regions are ranked according to their contribution to the recognition accuracy on the cross-validation dataset. These rankings are used as a guiding factor for our algorithm. The contribution of a local region is determined by computing the negative of the recognition accuracy of SVM classifier on the cross validation dataset, ignoring the features contributed and taking all other features into consideration. The ranks of all the local regions are then determined by arranging them in descending order of their contributions, such that the earlier ranking local regions are more informative than the lower ranking local regions. The algorithm is initialized with an empty harmony memory that will contain the final pareto-optimal solution upon termination. The exploration (HMCR), exploitation (PAR) and opposition-based learning (jump rate or JR) parameters are self-adjusting parameters, that tune themselves with the passage of every generation for a given harmony memory size. If the number of members in the current population is less than the size of the harmony memory, random local regions that are not already present in the harmony memory are added, and their costs are computed .Recognition accuracy has been measured using SVM classifier. Recognition time is the average time taken to extract the features from the images. Redundancy is a measure of how many similar looking local regions are being computed for the images. The performance of the algorithm was tested on four different datasets, a dataset of isolated handwritten basic Bangla characters, a dataset of handwritten Bangla numerals, a dataset of handwritten English numerals. The system performed a trade-off between recognition cost, recognition accuracy, and redundancy.

Rumman Rashid Chowdhury et al. [12] proposed Bangla Handwritten Character Recognition utilizing Convolutional Neural Network with Data Augmentation. The dataset utilized in this trial is the BanglaLekha-Isolated dataset [1]. Utilizing Convolutional Neural Network, this model accomplishes 91.81% exactness on the letter sets (50 character classes) on the base dataset, and in the wake of growing the quantity of pictures to 200,000 utilizing information enlargement. The model was facilitated on a web server for the simplicity of testing and collaboration with the model. The framework was actualized on Google Colab, which is a cloud based web interface to run AI tests, and is accessible for AI scientists for nothing. The framework has 12 Gigabytes of memory, Intel Xeon CPU running at 2.20GHz clock speed and a vigorous GPU (Nvidia Tesla K80). It gives access to an online python journal which goes about as the UI. To give a graphical interface to the prepared model, the Flask library of Python was used, which gives the backend of a web server where the model will be facilitated. For the frontend, HTML, CSS and Javascriptwas utilized, where the client can collaborate with a canvas and compose Bangla letters in order. The server, facilitated at localhost:5000 by Flask, gets a picture input and resizes it to 50x50 which the model expects, at that point it experiences the prepared model and predicts the likelihood of each class. The one with the most extreme likelihood is chosen to be the letter which was drawn. The outcome was printed out to the site page. . It was additionally seen that utilizing a bigger measure of information with variety can assist the model with learning the highlights or qualities of the classes all the more viably. The web interface additionally furnishes a simple method to communicate with the model and perform ongoing approval.

AdeelYousaf et al. [13] proposed Size Invariant Handwritten Character Recognition using Single Layer FeedforwardBackpropagation Neural Networks. a recognition system based on neural network that follows offline handwritten characters has been proposed for Latin digits and alphabets. Each of the characters that are extracted through query image is then resized dynamically to 60x40 pixels' size and is then passed to the neural networks for the process of recognition. Dynamic resizing enables size invariance in the proposed system and also maintains the aspect ratio of the character so that the image is not distorted during resizing. Neural systems are

prepared with 19,422 English letters in order's example and 7,720 digits' examples that are composed through 150 distinct journalists in different styles of penmanship. the proposed calculation takes character picture as a sort of question and afterward utilizes one element like force and delivers results that are seen progressively perfect to calculations that utilize various highlights for the procedure of characterization.The query image acts as a scanned image on paper of A4 size that involves the handwritten text in English. However, the proposed algorithm can be extended to any of the font style or to any language, since it is seen dependent on the values of pixel and it is not seen based over the particular features of font or language. Once text objects are obtained and prepared for recognition using image processing techniques explained in the previous section, the CCs are transferred to neural network for the process of recognition. The design of two different types of one layered neural networks have been made through using the feed forward back-propagation. The training of neural network that is used for recognizing alphabets has been done using the sample size of 13,596 of various alphabets that have been written in various styles by around 150 writers. The input that has been provided to the neural network is in the form of vector of around 2400 length and it is resized version of 60x40px image of one alphabet. High recognition rates have been achieved even without feature extraction. Dynamic resizing has been employed to maintain the quality of connected components extracted from a query image while resizing them to the standard size. The proposed system successfully segments out the handwritten text characters from a query image and achieves a precision of 95.69%. Sampath et al. [14] proposed Handwritten optical character acknowledgment by mixture neural system preparing calculation, proposed a crossover neural system preparing calculation for English transcribed OCR. At first, the commotion in the info picture is evacuated utilizing the middle channel, and the picture is resized. At that point, the capabilities, positional, and auxiliary descriptors are removed from the information picture. When the capabilities are extricated, the proposed FLM based neural system distinguishes the manually written character. The FLM proposed by consolidating the Firefly and the Levenberg–Marquardt (LM) calculation for preparing the neural system. At long last, the proposed FLM-based neural system is incorporated inside the feed forward neural system. , a cross breed neural system preparing calculation for written by hand optical character acknowledgment framework is proposed for arranging and perceiving the 62 characters, for example, 26 capitalized English letters in order, 26 lowercase English letters in order and zero to nine digits. From the outset, the pixel esteems are gotten from the resized characters. At that point, the highlights sets are removed from the picture utilizing the proposed descriptors of both H-descriptors and G-descriptors. At that point, the extricated include set is applied to the proposed arrange preparing calculation for perceiving the character. In this paper, the FLM based neural system preparing calculation is proposed, which has been successfully formulated by joining the firefly and Levenberg–Marquardt calculation for the preparation procedure of the neural system. At long last, the proposed half breed and 90% precision was gotten. Dhurgham Ali Mohammed et al. [15] proposed Off-line manually written character acknowledgment utilizing a coordinated DBSCAN-ANN plot, proposed a built up a novel technique for transcribed Arabic characters by joining the Density-Based Clustering strategy with factual and morphological highlights. The main stage in acknowledgment of manually written character picture has been finished by binarization the picture at that point applies commotion expulsion systems. The Density-Based Algorithm used to sort and discover any state of groups dependent on pixel data positions. This method separated the picture into characters. Each character will break down into four locales from the centroid followed by highlight extraction. These highlights incorporate vertical and even projections, upper and lower profile, rectangularity and direction. The aftereffects of the current procedure will move to the Neural Network (NN) stage which produces a significant level of rightness and exactness via preparing. The testing results contrasted and two of condition of-craftsmanship looks into. The proposed Arabic word acknowledgment framework is outfitted towards the cutting edge disconnected content system techniques. The written by hand character pictures IFN-ENT dataset is utilized to cover explicit states of Arabic characters. It comprises of in excess of 2900 different characters with Bitmap picture type. The procedure begins with binarization of word picture followed by division of the chose word into letter fragments. DBSCAN can sort and discover any state of bunches dependent on pixel data places that untruth near one another in Arabic character. A coordinated DBSCAN-ANN plot has been created dependent on character highlights extraction and character acknowledgment. The neurons of information portrayal can be dictated by highlight vector length. Additionally, the information characters considered 168 components dependent on 28 neurons as a yield layer. The procedures recognized the characters dependent on two layer log-sigmoid exchange work which considered as ideal for learning. The capacity creates yield go somewhere in the range of 0 and 1. Additionally, the system date arbitrarily isolated into two classifications. The first is for tanning which is viewed as 80% of the information and the second is 20% which is utilized for testing the framework. Back engendering preparing strategy is utilized dependent on guideline of slope drop. Vijaya Kumar et al. [16] proposed Handwritten Hindi Character Recognition utilizing Deep Learning Techniques, to perceive transcribed Hindi characters utilizing profound learning approaches like Convolutional Neural Network (CNN) With Optimizer RMSprop (Root Mean Square Propagation), Deep Feed Forward Neural Networks(DFFNN). The proposed framework has been prepared on tests of enormous arrangement of database pictures and tried on tests pictures from client characterizes informational index and from this analysis we accomplished extremely high acknowledgment results. Versatile Moment (Adam) Estimation is another strategy

that figures versatile learning rates for every parameter. Adam, a calculation for first-request angle based improvement of stochastic target capacities, in light of versatile assessments of lower-request minutes. The strategy is clear to actualize, is computationally proficient, has little memory necessities, is invariant to slanting rescaling of the slopes, and is appropriate for issues that are enormous as far as information and additionally parameters.The feed-forward Neural Network input layer contains n-dimensional vector as contribution to the system and contains L-1 shrouded layers as center layers for the most part two concealed layers are utilized and may increment dependent on the prerequisite. At long last there is one yield layer containing k number of yield classes. Every neuron in the shrouded layer and yield layer can be part into two sections: preactivation and activation.In a large portion of FFNN will have two concealed layers with 16 or 32 neurons and the sky is the limit from there, Hidden layers are increased with various arbitrary load of picture pixel information which is between 0 to 1. Be that as it may, in Deep Feed forward Neural Network was plan with same two shrouded layers and each concealed layers comprises of enormous arrangement of neurons for example we utilized 512 neurons are taken and this are duplicated with irregular weights.These strategies are train and test on a standard client characterize dataset which is gather from various clients. From exploratory outcomes, it is seen that DFFNN, CCN-Adam and CNN-RMSprop yield the best exactness for Handwritten Hindi characters contrasted with the elective strategies.BakiKoyuncu et al. [17] presented Handwritten Character Recognition by utilizing Convolutional Deep Neural Network, is audited to perceive the written by hand characters in this investigation. Numerous scientists have created frameworks for transcribed character acknowledgment. A few significant frameworks are referenced in this work. Character acknowledgment systems have been designed using diverse reason. The system created by certain inquires about can be built by utilizing equipment with huge scope joining hardware (VLSI). The information character acknowledgment of this structure is impervious to dynamic movement. Different explores used hamming mistake revising codes from correspondence hypothesis with neural system framework in their structure. Another procedure was created to recognize the composed hand characters in various vernaculars in its Neural System. These structures created exact outcomes yet in addition committed errors if the composed hand characters are in extraordinary organization. One of the scientists has even offered a procedure to relate the reliance between hand essayists and their handwriting. These investigations have for the most part used the Multi-layer feed forward neural system framework in their methods.In this examination, Modified National Institute of Standards and Technology (MNIST) database distributed by US division of trade is sent. This database contains countless pictures of composed hand characters. Lessening the size of the photos decreases the general time taken to set up the neural system framework to work. A convolutional neural framework is investigated for perusers' consideration. This framework is very unique for perceivingwritten hand characters. This work relies upon the gathering of characters at the contribution of (CNN). Appear differently in relation to other profound learning designs, CNN has ideal execution in the two pictures and enormous information. The mean to utilize profound learning was to take focal points of the intensity of CNN that can oversee enormous components of information and offer their loads. Soman et al. [18] proposed On creating manually written character picture database for Malayalam language content. The target of this paper is to construct a written by hand character picture database for Malayalam language script.The one of a kind orthographic portrayal of the Malayalam characters frames the distinctive character classes, and the present variant of the database contains 85 character classes every now and again utilized recorded as a hard copy Malayalam content. Written by hand information tests gathered from 77 local Malayalam essayists. For separating the character pictures from the manually written information sheets, dynamic shape model-based picture division calculation used. Acknowledgment tests led on the made character picture database by utilizing diverse element extraction procedures. The character based acknowledgment execution is assessed for MalCharDb, utilizing distinctive component extraction and order calculations. The component extraction process extricates educational descriptors from the pictures and helps the classifiers in assessing choice limits among taking an interest classes. Typically the component descriptors used for grouping incredibly influences the acknowledgment execution of the fundamental framework. The component portrayal dependent on curvelet change (CT), is gotten by figuring vertical and level projection profile of the coarse bend co-productive determined through curvelet change on the character pictures The character classes are framed dependent on the one of a kind orthographic character shapes present in Malayalam content. 85 Malayalam character classes including Malayalam vowels, consonants, half consonants, vowel and consonant modifiers and conjunct characters are considered for database creation. Written by hand picture information gathered from 77 local Malayalam authors, and dynamic form model based minimization procedure is utilized for character division. The current Malayalam character picture database, Amrita_MalCharDb contains 29,302 Malayalam character picture designs. Acknowledgment execution of Amrita_MalCharDb is assessed by utilizing distinctive component extraction procedures. Dissipating convolutional organize based highlights could accomplish 91.05% acknowledgment exactness among thought about strategies. Liang Xu et al. [19] presented Recognition of Handwritten Chinese Characters Based on Concept Learning. Idea learning is a hominine learning approach. Dissimilar to existing profound learning models, calculated model learning can be acknowledged by utilizing as meager as one example. This paper is the first to propose a written by hand Chinese character acknowledgment technique dependent on idea learning. Unique in relation to

the current picture portrayal based character acknowledgment strategies, the proposed strategy assembles a meta stroke library with earlier information, and afterward, presents a Chinese character applied model dependent on stroke relationship getting the hang of utilizing a character stroke extraction technique and Bayesian program learning. During character acknowledgment, Monte Carlo Markov bind testing is used to get the character age model for each character calculated. This age model can ascertain the likelihood of the objective and preparing characters being a similar characterization, and accordingly decides the arrangement of the objective character. In the idea learning-based transcribed Chinese character acknowledgment strategy, the strokes developing a specific character are separated utilizing the previously mentioned character strokeextraction method. Chinese character calculated model was worked by utilizing character stroke extraction and Bayesian program learning, and a character age model for each character theoretical model worked by utilizing Monte Carlo Markov Chain examining during the character acknowledgment. The trial results show that the proposed technique can prepare the reasonable model for character characterization expectation utilizing as not many as one character test. Ahmed TalatSahloler al. [20] presented Handwritten Arabic Optical Character Recognition Approach Based on Hybrid Whale Optimization Algorithm With Neighborhood Rough Set. a half and half AI approach that uses neighborhood harsh sets with a twofold whale advancement calculation to choose the most suitable highlights for the acknowledgment of written by hand Arabic characters. To approve the proposed approach, we utilized the CENPARMI dataset, which is a notable dataset for AI tests including written by hand Arabic characters. The outcomes show away from of the proposed approach as far as acknowledgment precision, memory impression, and processor time than those without the highlights of the proposed strategy. When looking at the aftereffects of the proposed technique with other ongoing best in class enhancement calculations, the proposed approach beat all others in all investigations. Also, the proposed approach shows the most elevated acknowledgment rate with the littlest utilization time contrasted with profound neural systems, for example, VGGnet, Resnet, Nasnet, Mobilenet, Inception, and Xception. The proposed approach was likewise contrasted and as of late distributed works utilizing the equivalent dataset, which further affirmed the remarkable grouping precision and time utilization of this methodology. The proposed approach comprises of four phases. The first is preprocessing of the dataset, which intends to expel commotion and clean the information. The second is highlight extraction, which expects to remove highlights from the information, for example, slope highlights, vertical and even projection highlights, vertical/level/askew projection highlights, and different highlights. The significant third stage is highlight determination, which is viewed as the fundamental commitment of this paper. In this stage, the component choice methodology begins by creating an arbitrary populace that speaks to a lot of arrangements. At that point, every arrangement is changed over into a paired form, where the highlights that relate to 1's are viewed as significant highlights, while different highlights are disregarded. From that point, the nature of the chose highlight (in view of the present arrangement) is assessed through registering the goal work. The principle reason for this paper is to fabricate a cross breed approach that can choose adequate highlights that improve the exhibition of written by hand Arabic characters in the littlest measure of time with a low memory impression. The outcomes show that the BWOA-NRS approach can choose the most fitting highlights, which evidently improves the characterization execution. The outcomes were contrasted with the latest component determination calculations, for example, ABC, SCA, GWA, ALO, and SSA, it very well may be seen that the BWOA-NRS calculation beats different methodologies dependent on swarm strategies.

3. Existing System

Handwritten character recognition is a difficult problem due to the great variations of writing styles, different size and orientation angle of the characters. Among different branches of handwritten character recognition it is easier to recognize English alphabets and numerals than Tamil characters. Early techniques in handwritten character recognition failed, notably in the presence of blur, low contrast, low resolution, high image noise, and other distortions. In order to avoid the distortions Convolutional Neural Network algorithm has been used to recognize and classify the image.

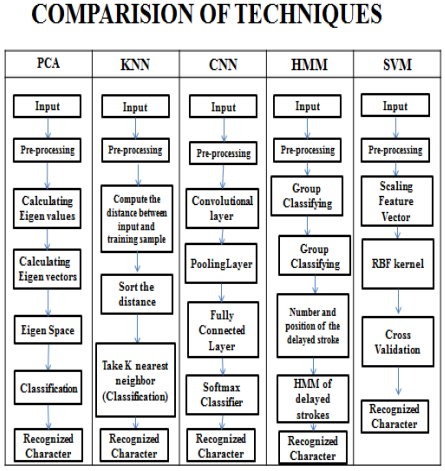


Figure 1. Comparison of Existing Algorithms

The purpose of this project is to take handwritten Tamil characters as input, process the character, train the neural network algorithm, to recognize the pattern from the word and modify the character to a beautified version of the input. This project is aimed at developing a model which will be helpful in recognizing characters of Tamil language from the word. To recognize and classify the character image even if the image is blur or of any distortions.

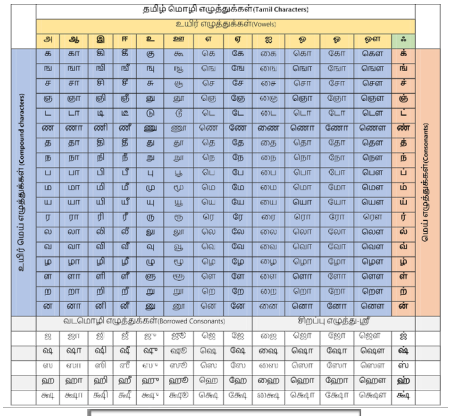
- A detailed study on different image preprocessing and feature extraction method and different classifiers.
- To propose new or modified feature extraction method(s) suitable for Tamil character.
- To propose new algorithms suitable for Tamil character dataset.
- To speed up the process of character recognition, since my character recognition system recognize visual patterns directly from pixel images with minimal preprocessing.

4. Proposed System

Character Recognition is the acknowledgment of printed or composed content characters by a PC. This includes photographs examining the content character-by-character, the examination of the checked-in picture, and afterward interpretation of the character picture into character codes, for example, ASCII, ordinarily utilized in the information preparing. Recognition is an area that covers various fields such as, face recognition, fingerprint recognition, image recognition, character recognition, numerals recognition, etc.

This Character Recognition System is a shrewd framework utilizing a Convolutional Neural Network that groups written by hand Characters as human distinguishes. Character characterization is a significant piece of numerous PC vision and picture having issues like Optical character acknowledgment, tag acknowledgment, and so forth. The manually written character grouping is a troublesome errand because of the diverse penmanship styles of the scholars. Manually written Tamil Character Recognition has extensive consideration as of late. The proposed approach is fit for perceiving characters in an assortment of testing conditions utilizing the Convolutional Neural Network, where conventional character acknowledgment frameworks fall flat, eminently within the sight of low goals, significant haze, low differentiation, and different bends. A dataset is a collection of data. Data like database table or an Excel Spreadsheet are traditional structure of data (Structured Data). A single row of data is called an instance. A single column of data is called a feature. Features have a data type. A dataset that is feed into the machine learning algorithm to train our model. Testing Dataset: A dataset that is being used to validate the accuracy of the model but is not used to train the model. This work uses the Isolated Handwritten Tamil Character data- set developed by HP Labs India. This dataset consists of 156 different Tamil characters (hpl-tamil-iso-char) written by native Tamil writers from various cities of Southern India using HP TabletPC1. The dataset contains approximately 500 samples for each class (with very few classes having around 300 samples) with a total of 82,928 samples and is freely available. The entire Tamil character set can be represented with these 156 unique characters (Table 1).

Table 1. HPL Tamil Character Datasets



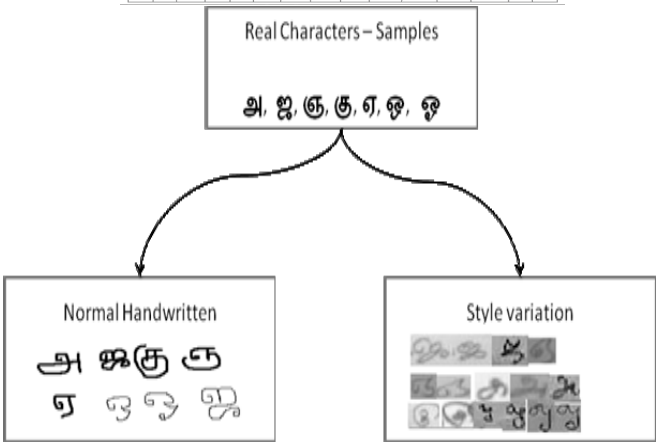


Figure 2. Character Sample at Various Levels

Handwritten character recognition is a difficult problem due to the great variations of writing styles, different size and orientation angle of the characters. Among different branches of handwritten character recognition it is easier to recognize English alphabets and numerals than Tamil characters.

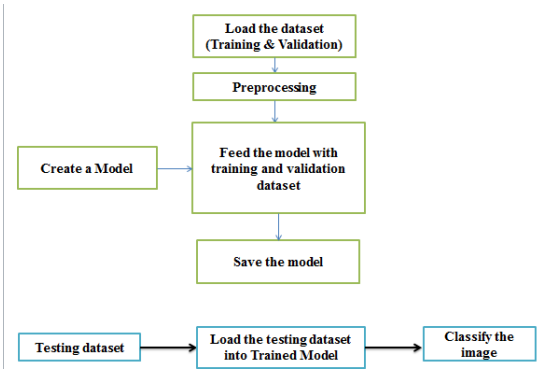


Figure 3. Training and Validation of Proposed Model

The proposed model consists of two parts: Training Part Recognition Part.

- Training Part - Training part involves data pre-processing, building the network architecture and training the network with the preprocessed data.
- Recognition Part - Recognition part involves recognizing the character using the trained model.

- Pre-Processing is a typical name for tasks with pictures at the most reduced degree of deliberation both inf0rmation and yield are force pictures.

The point of pre-preparing is an improvement of the picture information that stifles undesirable twists or upgrades some picture highlights significant for additional handling. Different groupings of picture pre-processing strategies exist. Picture pre-handling techniques utilize impressive excess in pictures. Neighboring pixels comparing to one item in genuine pictures have basically the equivalent or comparative splendor esteem.

In this manner, the misshaped pixels can frequently be reestablished as a normal benefit of neighboring pixels. The idea of clamor (as a rule its ghastly attributes) is in some cases known as information about items that are scanned for in the picture, which may streamline the preprocessing impressively. Pixel Brightness Transformations are brightness changes alter pixel splendor that change relies upon the properties of a pixel itself. Splendor rectifications and Grayscale changes, Brightness revision considers a unique splendor pixel position in the picture. Grayscale changes change the splendor regardless of position in the picture. Position subordinate brilliance amendment, the affectability of picture obtaining and digitization gadgets ought not to rely upon the situation in the picture, the planar change has been cultivated, and new point organizes (x,y) were acquired. The situation of the point doesn't, all in all, fit the discrete raster of the yield picture. Qualities on the whole number lattice are required.

Every pixel in the yield picture raster can be gotten by splendor addition of some neighboring no integer tests. The splendor interjection issue is normally communicated in a double manner (by deciding the brilliance of the first point in the info picture that compares to the point in the yield picture lying on the discrete raster). Figuring the brilliance estimation of the pixel (x,y) in the yield picture where x and y lie on the discrete raster. The dataset contains 82,929 images in tiff or png format. These images are obtained from the online version using simple piece- wise linear interpolation and a constant thickening factor. The images are bi-level images with background being white (255) and the foreground in black (0).

The images are of varying sizes which were size normalized to 64 64 using bilinear interpolation technique and scaled to 0, 1 range. We performed training on two set of inputs, one with the original images and another with inverted images (foreground as 1 and background as 255). Preprocessing is the first step, once the datasets are preprocessed and after converting the raw data into cleaned data they are normalized as they are of varying sizes.

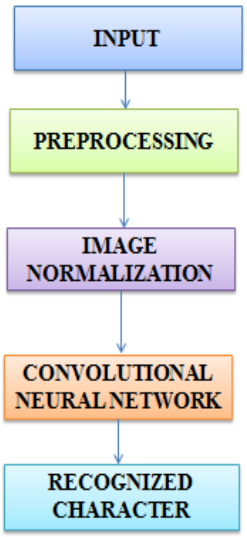


Figure 4. Preprocessing of the model

Normalization or standardization is a procedure that changes the scope of pixel force esteems. Applications incorporate photos with poor difference because of glare, for instance. Standardization is now and then called differentiate extending or histogram extending. In increasingly broad fields of information handling, for example, computerized signal preparing, it is alluded to as unique range expansion. The reason for dynamic range development in the different applications is generally to bring the picture, or other sort of sign, into a range that is progressively recognizable or ordinary to the faculties, henceforth the term standardization.

Assume dataset X, which has N rows(entries) and D columns(features). X[:,i] represent feature i and X[j,:] represent entry j.

$$\hat{X}[:,i] = \frac{X[:,i]-\mu_i}{\sigma_i}, (\mu_i = \frac{1}{N} * \sum_{k=1}^{N} X[k,i], \sigma_i = \sqrt{\frac{1}{N-1} * \sum_{k=1}^{N} (X[k,i]-\mu_i)^2})$$

This transformation sets the mean of data to 0 and the standard deviation to 1. In most cases, standardization is used feature-wise.

$$\hat{X}[:,i] = \frac{X[:,i]-\min(X[:,i])}{\max(X[:,i])-\min(X[:,i])}$$

This method rescales the range of the data to [0,1]. In most cases, standardization is used feature-wise as well. its normal purpose is to convert an input image into a range of pixel values that are more familiar or normal to the senses, hence the term normalization.

Figure 5. Before Normalization

Figure 6. After Normalization

Data Augmentation is a system that empowers specialists to altogether build a decent variety of information accessible for preparing models, without really gathering new information. Information enlargement methods, for example, trimming, cushioning, and even flipping are regularly used to prepare enormous neural systems. Profound Learning in some cases may run into an issue where information has a restricted size. To show signs of improvement speculation the model needs more information and as much variety conceivable in the information. Now and again, the dataset isn't sufficiently large to catch enough variety; in such cases producing more information from the given dataset is finished. That is the place Data Augmentation assumes a significant job. Information growth implies expanding the measure of preparing information utilizing data accessible from the preparation information. It's an assortment of procedures for "improving" preparing information. Regularly used to mentor the models to overlook superfluous varieties in the information. For instance, if the preparation of a picture classifier, needs to take care of the model initially and flipped/pivoted forms of each preparation picture (not such a smart thought on the off chance that you were preparing an OCR model!). Likewise basic yet astute: "arbitrary eradication", in which a little square shape of a preparation picture is haphazardly darkened, to attempt to make a model powerful to impediments. A generative ill-disposed system was utilized to change manufactured preparing pictures to make them look like genuine photos (the preparation task was to decide the look heading of a human subject). Information increase is utilized here and there to make the model increasingly powerful to over-fitting. Now and again, pictures, essentially that is taken from the preparation set and change it (pivot, flip, shading variety, clamor,..). It's a basic method to construct a greater information set. Barely any mainstream conventional Data growth methods: 1. Flip: Flipping images on horizontal or vertical axis, 2. Rotation: Rotate an image with certain degree, 3. Crop: Randomly, crop a section from given image and resize, 4. Add Noise: Adding Gaussian noise to a given image and 5. Color Jittering: Random color manipulation.

Equality Rate = Number of images in one label / Total Number of images in all labels

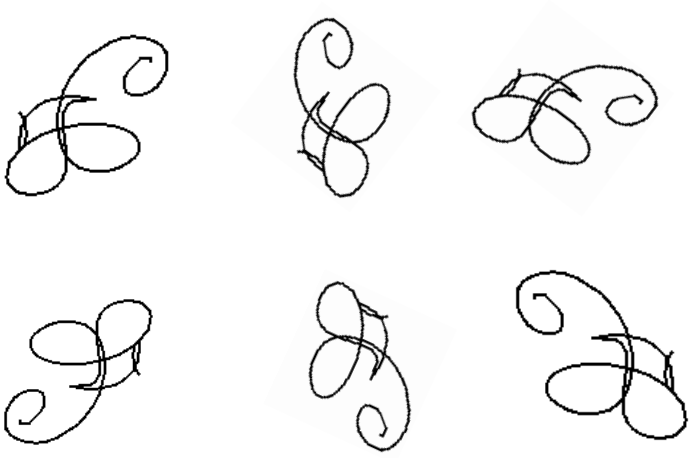


Figure 7. Sample Augmentation of Dataset

Convolutional Neural Networks (CNN) are assuming an indispensable job these days in each part of PC vision applications. A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning calculation that can take in an information picture, allot significance (learnable loads and predispositions) to different angles/protests in the picture and have the option to separate one from the other. The pre-preparing required in a ConvNet is a lot of lower when contrasted with other characterization calculations. While in crude strategies channels are hand-designed, with enough preparation, ConvNets can gain proficiency with these channels/qualities. The engineering of a ConvNet is undifferentiated from that of the availability example of Neurons in the Human Brain and was roused by the association of the Visual Cortex. Singular neurons react to upgrades just in a limited area of the visual field known as the Receptive Field. An assortment of such fields covers to cover the whole visual territory. Every neuron gets a few data sources, plays out a speck item, and alternatively tails it with a non-linearity. The entire system despite everything communicates a solitary differentiable score work: from the crude picture pixels toward one side to class scores at the other. They despise everything to have a loss work (for example SVM/Softmax) on the last (completely associated) layer. Input Layer - The input layer of a neural system is made out of counterfeit info neurons and carries the underlying information into the framework for additional handling by ensuing layers of fake neurons. The info layer is the earliest reference point of the work process for the fake neural system. Convolution Layer - The convolutional layer is the center structure square of a CNN. The layer's parameters comprise of a lot of learnable channels (or portions), which have a little open field, however, reach out through the full profundity of the info volume. Pooling Layer - A pooling layer is another structure square of a CNN. Its capacity is to logically diminish the spatial size of the portrayal to lessen the measure of parameters and calculation in the system. The

pooling layer works on each element map autonomously. Pooling layers are utilized to decrease the components of the element maps. In this way, it diminishes the number of parameters to learn and the measure of calculation acted in the system. The pooling layer condenses the highlights present in a district of the element map created by a convolution layer. In this way, further activities are performed on abridged highlights rather than correctly situated highlights produced by the convolution layer. This makes the model progressively vigorous to varieties in the situation of the highlights in the information picture.

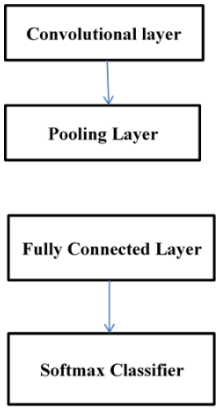


Figure 8. CNN Model

Types of Pooling Layers. Max Pooling - Max pooling is a pooling activity that chooses the most extreme component from the locale of the element map secured by the channel. Along these lines, the yield after the max-pooling layer would be an element map containing the most conspicuous highlights of the past element map. Average Pooling - Normal pooling figures the normal of the components present in the district of highlight map secured by the channel. Along these lines, while max-pooling gives the most unmistakable element in a specific fix of the component map, normal pooling gives the normal of highlights present in a fix. Global Pooling - Worldwide pooling diminishes each direct in the element guide to a solitary worth. In this way, a nh x nw x nc include map is diminished to 1 x 1 x nc highlight map. This is equal to utilizing a channel of measurements nh x nw, for example, the components of the element map. Further, it tends to be either worldwide max pooling or worldwide normal pooling. A Fully associated layer is the genuine segment that does the discriminative learning in a Deep Neural Network. It's a straightforward Multi-layer perceptron that can learn loads that Diverse actuation capacities have been utilized across different structures of convolution neural systems. Nonlinear actuation capacities, for example, ReLU, LReLU, PReLU, and Swish have demonstrated wagered results when contrasted with the great sigmoid or digression functions. These nonlinear capacities have helped in accelerating the preparation. In this work, we have attempted distinctive enactment capacities and seen ReLU as more successful than others. A Convolutional Neural Network (CNN, or ConvNet) is a unique sort of multi-layer neural systems, intended to perceive visual examples legitimately from pixel pictures with negligible preprocessing. The ImageNet venture is a huge visual database intended for use in visual item acknowledgment programming research. The ImageNet venture runs a yearly programming challenge, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where programming programs contend to accurately characterize and recognize articles and scenes.

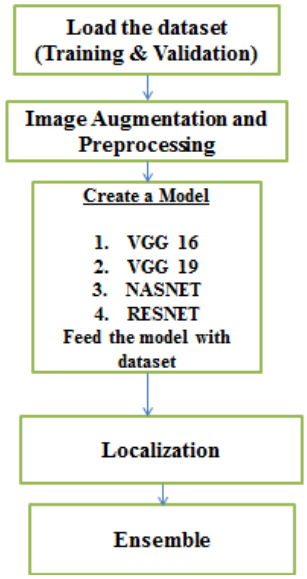


Figure 9. CNN Models used in Proposed Wok

6This system is portrayed by its effortlessness, utilizing just 3×3 convolutional layers stacked on one another in expanding profundity. Decreasing volume size is dealt with by max pooling. Two completely associated layers, each with 4,096 hubs are then trailed by a Softmax classifier (above). The "16" and "19" represent the number of weight layers in the system. The VGG organize engineering was presented by Simonyan and Zisserman. Simonyan and Zisserman discovered preparing VGG16 and VGG19 testing (explicitly in regards to the assembly on the more profound systems), so as to make preparing simpler, they initially prepared littler renditions of VGG with fewer weight layers. The littler systems combined and were then utilized as statements for the bigger, more profound systems this procedure is called pre-preparing. While seeming well and good, pre-preparing is a very tedious, dull assignment, requiring a whole system to be prepared before it can fill in as an introduction for a more profound system. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is a yearly PC vision rivalry. Every year, groups contend on two assignments. The first is to recognize questions inside a picture originating from 200 classes, which is called object restriction. The second is to group pictures, each marked with one of 1000 classifications, which is called picture order. VGG 16 was proposed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University in 2014 in the paper "Extremely DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE

RECOGNITION". This model won the first and second spots on the above classifications in the 2014 ILSVRC challenge. The ImageNet dataset contains pictures of the fixed size of 224*224 and has RGB channels. In this way, we have a tensor of (224, 224, 3) as our information. This model procedure the info picture and yields the vector of 1000 qualities. Built using: 6Convolutions layers (used only 3*3 size), Max pooling layers (used only 2*2 size), Fully connected layers at end.Total 16 layers, Model size: 528MB

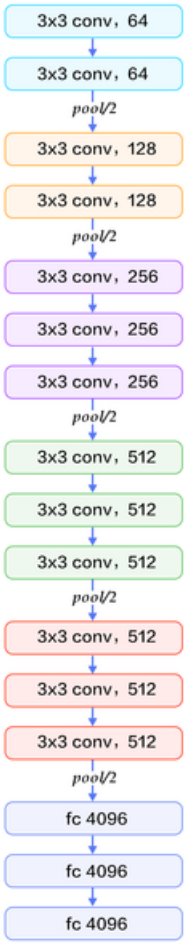


Figure 10. Layers of VGG Network

The contribution to the system is picture of measurements (224, 224, 3). The initial two layers have 64 channels of 3*3 channel size and same cushioning. At that point after a maximum pool layer of step (2, 2), two layers which have convolution layers of 256 channel size and channel size (3, 3). This followed by a maximum pooling layer of step (2, 2) which is same as past layer. At that point there are 2 convolution layers of channel size (3, 3) and 256 channel. After that there are 2 arrangements of 3 convolution layer and a maximum pool layer. Each have 512 channels of (3, 3) size with same cushioning. This picture is then passed to the heap of two convolution layers. In these convolution and max pooling layers, the channels we use is of the size 3*3 rather than 11*11 in AlexNet and 7*7 in ZF-Net. In a portion of the layers, it additionally utilizes 1*1 pixel which is utilized to control the quantity of info channels. There is a cushioning of 1-pixel (same cushioning) done after every convolution layer to forestall the spatial component of the picture. The layers are listed down:

1. Convolution using 64 filters
2. Convolution using 64 filters + Max pooling
3. Convolution using 128 filters
4. Convolution using 128 filters + Max pooling
5. Convolution using 256 filters
6. Convolution using 256 filters
7. Convolution using 256 filters + Max pooling
8. Convolution using 512 filters
9. Convolution using 512 filters
10. Convolution using 512 filters + Max pooling
11. Convolution using 512 filters
12. Convolution using 512 filters
13. Convolution using 512 filters + Max pooling
14. Fully connected with 4096 nodes
15. Fully connected with 4096 nodes
16. Output layer with Softmax activation with 1000 nodes

Application: Given image → find object name in the image It can detect any one of 1000 images. It takes input image of size 224 * 224 * 3 (RGB image)

5. Experimental Results

It is the sensible portrayal of the ResNet.At last, at the ILSVRC 2015, the alleged Residual Neural Network (ResNet) by Kaiming He et al presented a novel design with "skip associations" and highlights substantial group standardization. Such skip associations are otherwise called gated units or gated intermittent units and have a solid comparability to ongoing fruitful components applied in RNNs. On account of this system they had the option to prepare a NN with 152 layers while as yet having lower multifaceted nature than VGGNet. It accomplishes a best 5 mistake pace of 3.57% which beats human-level execution on this dataset. ResNet was at first structured as a technique to take care of the disappearing inclination issue. This is where backpropagated angles become incredibly little as they're increased again and again, restricting the size of a neural system. The ResNet engineering endeavors to comprehend that by utilizing skip associations, that is adding alternate ways that permit information to skirt past layers. The model comprises of a progression of convolutional layers + skip associations, at that point normal pooling, at that point a yield completely associated (thick) layer. For move learning, we just need the convolutional layers as those to contain the highlights we're keen on, so we would need to overlook them when bringing in the model. The inclination to include such a large number of layers by profound learning experts is to remove significant highlights from complex pictures. Along these lines, the primary layers may identify edges, and the ensuing layers toward the end may distinguish unmistakable shapes, similar to feels worn out on a vehicle. However, on the off chance that we add in excess of 30 layers to the system, at that point its presentation endures and it accomplishes a low exactness.

This is in opposition to the reasoning that the expansion of layers will improve a neural system. This isn't expected to overfitting, in light of the fact that all things considered, one may utilize dropout and regularization systems to unravel the issue out and out. It's chiefly present on account of the mainstream disappearing slope issue. The ResNet152 model with 152 layers won the ILSVRC Imagenet 2015 test while having lesser parameters than the VGG19 arrange, which was extremely mainstream around then. A remaining system comprises of lingering units or squares which have skip associations, likewise called personality associations.

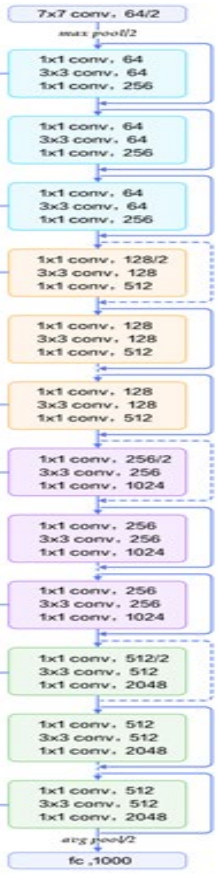


Figure 11. Layers of RESNet

A leftover square has a 3 x 3 convolution layer followed by a cluster standardization layer and a ReLU actuation work. This is again proceeded by a 3 x 3 convolution layer and a group standardization layer. The skip association essentially avoids both these layers and includes straightforwardly before the ReLU actuation work. Such leftover squares are rehashed to frame a remaining system.

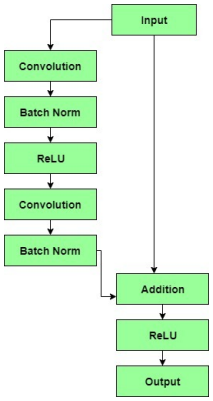


Figure 12. Block of RESNet

The yield of the past layer is added to the yield of the layer after it in the leftover square. The bounce or skip could be 1, 2 or even 3. While including, the elements of x might be not quite the same as F(x) because of the convolution procedure, bringing about a decrease of its measurements. In this manner, we include an extra 1 x 1 convolution layer to change the elements of x. Another regularization system called Scheduled Drop Path is likewise proposed which essentially improves the speculation in the models. Finally, this model accomplishes cutting edge results with littler model size and lower unpredictability (FLOPs). In this system, however the general design is predefined as appeared over, the squares or cells are not predefined by creators. Rather, they are looked by fortification learning search strategy. for example the quantity of theme redundancies N and the quantity of introductory convolutional channels are as free parameters, and utilized for scaling. In particular, these phones are called Normal Cell and Reduction Cell. Ordinary Cell: Convolutional cells that arrival an element guide of a similar measurement. Decrease Cell: Convolutional cells that arrival an element map where the element map stature and width is diminished by a factor of two. Just the structures of (or inside) the Normal and Reduction Cells are looked by the controller RNN (Recurrent Neural Network).

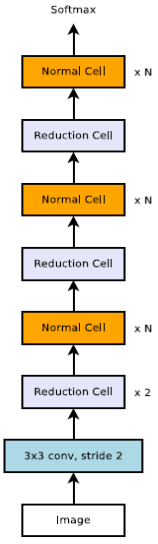


Figure 13. Layers of NASNet

The undertaking of article confinement is to foresee the item in a picture just as its limits. The contrast between object restriction and item location is unpretentious. Just, object restriction plans to find the principle (or generally noticeable) object in a picture while object discovery attempts to discover all the articles and their limits. Item discovery can be performed utilizing a system called "sliding window recognition". We train a ConvNet to recognize questions inside a picture and use windows of various sizes that we slide on it. For every window, we play out a forecast. Its huge drawback is the computational cost, which is extremely broad since we can have a great deal of windows. The answer for that is the sliding window recognition registered convolutionally. Particular quest calculation is utilized for object acknowledgment.

Work process

- Split a picture into S×S cells. In the event that an article's inside falls into a cell, that cell is "mindful" for distinguishing the presence of that object. Every cell predicts (a) the area of B jumping boxes, (b) a certainty score, and (c) a likelihood of article class molded on the presence of an item in the bouncing box.

- The directions of bouncing box are characterized by a tuple of 4 qualities, (focus x-coord, focus y-coord, width, stature) — (x,y,w,h), where x and y are set to be balanced of a cell area. In addition, x, y, w and h are standardized by the picture width and stature, and in this manner all between (0, 1].

- A certainty score demonstrates the probability that the cell contains an item: Pr(containing an article) x IoU(pred, truth); where Pr = likelihood and IoU = connection under association.

- If the cell contains an item, it predicts a likelihood of this article having a place with each class Ci,i=1,… ,K: Pr(the object has a place with the class C_i | containing an item). At this stage, the model just predicts one lot of class probabilities per cell, paying little mind to the quantity of bouncing boxes, B.

- In absolute, one picture contains S×S×B jumping boxes, each container comparing to 4 area forecasts, 1 certainty score, and K contingent probabilities for object characterization. The complete expectation esteems for one picture is S×S×(5B+K), which is the tensor state of the last conv layer of the model.

- The last layer of the pre-prepared CNN is adjusted to yield an expectation tensor of size S×S×(5B+K).

Figure 14. Sample Localization of Datasets

The loss comprises of two sections, the restriction loss for jumping box counterbalance expectation and the order loss for contingent class probabilities. The two sections are registered as the aggregate of squared mistakes. Two scale parameters are utilized to control the amount we need to build the loss from jumping box organize expectations (λcoord) and the amount we need to diminish the loss of certainty score forecasts for boxes without objects (λnoobj). Down-weighting the loss contributed by foundation boxes is significant as the majority of the jumping boxes include no occurrence. In the paper, the model sets λcoord=5 and λnoobj=0.5.

The dataset used for evaluation is the Tamil Character Classification from HP Labs India. The dataset consists of 85000 images of Tamil Characters. Each letter consists of approx. 300 images. Each of images are of different shapes.The models that have been analyzed are tested using Cross Entropy. Cross entropy is used to measure the performance of the classical model. It finds the probability of the event drawn. There is no specific analysis measure for the same and the efficiency of the algorithm can be visualized with the task it is entitled to.

The results show that a convolutional neural network is capable of achieving record breaking results on the Tamil dataset.
Screenshots

Finding the Equality rate and count of image in each folder

Equality Rate = Number of images in one label/ Total number of images in all labels

Sample Output:
['ணீ', 'ணு', 'ணூ', 'ண்', 'த', 'தி', 'தீ', 'து', 'தூ', 'த்', 'ந', 'நி', 'நீ', 'நு', 'நூ', 'ந்', 'ன', 'னி', 'னீ', 'னு', 'னூ', 'ன்',

'ப', 'பி', 'பு', 'பீ', 'பூ', 'ம', 'ப்', 'மி', 'மீ', 'மு', 'மூ', 'ம்', 'ய', 'யி', 'யீ', 'யு', 'யூ', 'ய்', 'ர', 'ரி', 'ரீ', 'ரு', 'ரூ', 'ர்', 'றி', 'ற', 'றீ', 'று', 'றூ', 'ற்', 'ல', 'லி', 'லீ', 'லு', 'லூ', 'ல்', 'ள', 'ளி', 'ளீ', 'ளு', 'ளூ', 'ள்', 'ழ', 'ழி', 'ழீ', 'ழு', 'ழூ', 'ழ்', 'வ', 'வி', 'வீ', 'வு', 'வூ', 'வ்', 'ஷ', 'ஷி', 'ஷீ', 'ஷு', 'ஷ்', 'ஸ', 'ஸி', 'ஸீ', 'ஸு', 'ஸ்', 'ஹ', 'ஹி', 'ஹீ', 'ஹு', 'ஹ்', 'ா', 'ெ', 'ே', 'ை', 'ஃ', 'அ', 'ஆ', 'இ', 'ஈ', 'உ', 'ஊ', 'எ', 'ஏ', 'ஐ', 'ஒ', 'ஓ', 'ஔ', 'க', 'கி', 'கீ', 'கு', 'கூ', 'க்', 'கூ', 'கூி', 'கூீ', 'கூு', 'கூ்', 'ங', 'ஙி', 'ஙீ', 'ஙு', 'ஙூ', 'ங்', 'ச', 'சி', 'சீ', 'சு', 'சூ', 'ச்', 'ஜ', 'ஜி', 'ஜீ', 'ஜு', 'ஜூ', 'ஜ்', 'ஞ', 'ஞி', 'ஞீ', 'ஞு', 'ஞூ', 'ஞ்', 'ட', 'டி', 'டீ', 'டு', 'டூ', 'ட்', 'ண', 'ணி']
[353, 337, 330, 342, 332, 339, 336, 342, 343, 344, 326, 329, 336, 339, 343, 341, 323, 342, 331, 337, 331, 338, 339, 339, 332, 339, 340, 339, 333, 331, 332, 335, 341, 339, 337, 336, 336, 336, 339, 342, 326, 338, 331, 338, 343, 328, 340, 335, 329, 331, 331, 339, 335, 338, 338, 333, 329, 343, 335, 339, 330, 325, 340, 333, 332, 338, 330, 326, 277, 341, 337, 339, 339, 338, 340, 339, 312, 325, 335, 282, 315, 169, 335, 336, 293, 336, 333, 331, 335, 283, 330, 329, 347, 329, 337, 340, 347, 341, 340, 332, 339, 345, 341, 342, 334, 329, 324, 344, 337, 334, 342, 337, 322, 336, 299, 313, 331, 571, 304, 331, 331, 314, 336, 310, 324, 331, 337, 334, 334, 323, 327, 344, 169, 169, 168, 162, 170, 333, 335, 342, 336, 327, 337, 341, 338, 337, 336, 338, 342, 341, 338]
Equality rate for ணீ = 0.007126562089919851
Equality rate for ணு = 0.006803545111339915
Equality rate for ணூ = 0.006662225183211193
Equality rate for ண் = 0.006904487917146145
Equality rate for த = 0.0067026023055336845
Equality rate for தி = 0.006843922233662407
Equality rate for தீ = 0.006783356550178669
Equality rate for து = 0.006904487917146145
Equality rate for தூ = 0.006924676478307391
Equality rate for த் = 0.006944865039468637
Equality rate for ந = 0.006581470938566208
Equality rate for நி = 0.006642036622049947
Equality rate for நீ = 0.006783356550178669
Equality rate for நு = 0.006843922233662407
Equality rate for நூ = 0.006924676478307391
Equality rate for ந் = 0.006884299355984899
Equality rate for ன = 0.00652090525508247
Equality rate for னி = 0.006904487917146145
Equality rate for னீ = 0.006682413744372439
Equality rate for னு = 0.006803545111339915
Equality rate for னூ = 0.006682413744372439
Equality rate for ன் = 0.006823733672501161
Equality rate for ப = 0.006843922233662407
Equality rate for பி = 0.006843922233662407
Equality rate for பு = 0.0067026023055336845
Equality rate for பீ = 0.006843922233662407
Equality rate for பூ = 0.006864110794823653
Equality rate for ம = 0.006843922233662407
Equality rate for ப் = 0.00672279086669493
Equality rate for மி = 0.006682413744372439
Equality rate for மீ = 0.0067026023055336845
Equality rate for மு = 0.006763167989017423
Equality rate for மூ = 0.006884299355984899
Equality rate for ம் = 0.006843922233662407
Equality rate for ய = 0.006803545111339915
Equality rate for யி = 0.006783356550178669
Equality rate for யீ = 0.006783356550178669

Checking equality rate for augmenting

After finding the number of images in each label and equality rate, finding the equality rate after data augmenting to increase the efficiency of the system in learning and recognizing the character.

Sample Output:

['ணீ', 'ணு', 'ணூ', 'ண்', 'த', 'தி', 'தீ', 'து', 'தூ', 'த்', 'ந', 'நி', 'நீ', 'நு', 'நூ', 'ந்', 'ன', 'னி', 'னீ', 'னு', 'னூ', 'ன்', 'ப', 'பி', 'பு', 'பீ', 'ம', 'பூ', 'ப்', 'மி', 'மீ', 'மு', 'மூ', 'ம்', 'ய', 'யி', 'யீ', 'யு', 'யூ', 'ய்', 'ர', 'ரி', 'ரீ', 'ரு', 'ரூ', 'ர்', 'றி', 'ற', 'றீ', 'று', 'றூ', 'ற்', 'ல', 'லி', 'லீ', 'லு', 'லூ', 'ல்', 'ள', 'ளி', 'ளீ', 'ளு', 'ளூ', 'ள்', 'ழ', 'ழி', 'ழீ', 'ழு', 'ழூ', 'ழ்', 'வ', 'வி', 'வீ', 'வு', 'வூ', 'வ்', 'ஷ', 'ஷி', 'ஷீ', 'ஷு', 'ஷ்', 'ஸ', 'ஸி', 'ஸீ', 'ஸு', 'ஸ்', 'ஹ', 'ஹி', 'ஹீ', 'ஹு', 'ஹ்', 'ா', 'ெ', 'ே', 'ை', 'ஃ', 'அ', 'ஆ', 'இ', 'ஈ', 'உ', 'ஊ', 'எ', 'ஏ', 'ஐ', 'ஒ', 'ஓ', 'ஔ', 'க', 'கி', 'கீ', 'கு', 'கூ', 'க்', 'கூ', 'கூி', 'கூீ', 'கூு', 'கூ்', 'ங', 'ஙி', 'ஙீ', 'ஙு', 'ஙூ', 'ங்', 'ச', 'சி', 'சீ', 'சு', 'சூ', 'ச்', 'ஜ', 'ஜி', 'ஜீ', 'ஜு', 'ஜூ', 'ஜ்', 'ஞ', 'ஞி', 'ஞீ', 'ஞு', 'ஞூ', 'ஞ்', 'ட', 'டி', 'டீ', 'டு', 'டூ', 'ட்', 'ண', 'ணி']
[1008, 968, 946, 976, 938, 964, 957, 985, 974, 981, 932, 932, 965, 951, 972, 974, 922, 979, 945, 962, 941, 962, 963, 972, 953, 972, 964, 966, 950, 950, 936, 951, 971, 976, 960, 945, 959, 958, 977, 974, 938, 961, 949, 957, 980, 935, 966, 962, 938, 935, 940, 968, 954, 959, 959, 964, 932, 977, 963, 976, 944, 930, 968, 963, 952, 969, 942, 931, 1057, 972, 957, 962, 959, 973, 976, 958, 887, 929, 955, 1084, 897, 500, 957, 952, 847, 958, 958, 946, 957, 811, 931, 941, 979, 938, 957, 978, 996, 984, 980, 944, 978, 986, 979, 989, 948, 936, 926, 980, 966, 941, 981, 964, 924, 964, 1124, 889, 949, 1571, 880, 948, 950, 905, 956, 903, 925, 943, 960, 950, 951, 930, 932, 977, 961, 975, 953, 923, 961, 956, 939, 965, 962, 940, 959, 974, 966, 965, 956, 964, 974, 972, 963]
Equality rate for ணீ = 0.006973076178090152
Equality rate for ணு = 0.006696366805943718
Equality rate for ணூ = 0.0065441766512631784
Equality rate for ண் = 0.006751708680373004
Equality rate for த = 0.006488834776833891
Equality rate for தி = 0.006668695868729074

Equality rate for தீ = 0.006620271728603448
Equality rate for து = 0.006813968289105952
Equality rate for தூ = 0.006737873211765682
Equality rate for த் = 0.006786297351891309
Equality rate for நு = 0.0064473283710119265
Equality rate for நி = 0.0064473283710119265
Equality rate for நீ = 0.006675613603032734
Equality rate for நு = 0.006578765322781482
Equality rate for நூ = 0.006724037743158361
Equality rate for ந் = 0.006737873211765682
Equality rate for ன = 0.006378151027975318
Equality rate for னி = 0.006772461883283987
Equality rate for னீ = 0.006537258916959518
Equality rate for னு = 0.006654860400121752
Equality rate for னூ = 0.006509587979744874
Equality rate for ன் = 0.006654860400121752
Equality rate for ப = 0.006661778134425413
Equality rate for பி = 0.006724037743158361
Equality rate for பு = 0.006592600791388804
Equality rate for பீ = 0.006724037743158361
Equality rate for ம = 0.006668695868729074
Equality rate for பூ = 0.006682531337336395
Equality rate for ப் = 0.006571847588477822
Equality rate for மி = 0.006571847588477822
Equality rate for மீ = 0.00647499930822657
Equality rate for மு = 0.006578765322781482
Equality rate for மூ = 0.0067171200088547
Equality rate for ம் = 0.006751708680373004
Equality rate for ய = 0.00664102493151443
Equality rate for யி = 0.006537258916959518
Equality rate for யீ = 0.00663410719721077
Equality rate for யு = 0.006627189462907109
Equality rate for யூ = 0.006758626414676665
Equality rate for ய் = 0.006737873211765682
Equality rate for ர = 0.006488834776833891

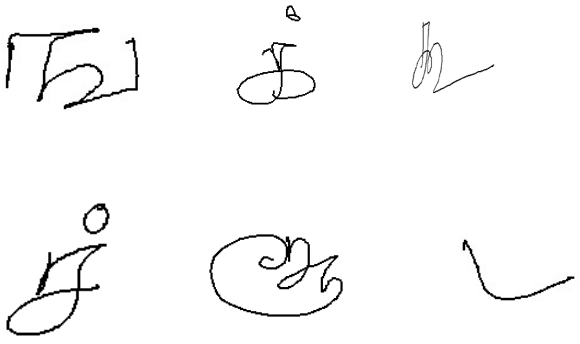
Saving the images in numpy format and Reading all the images from numpy array file

Sample Format:
/content/drive/My Drive/Colab Notebooks/datas/ஃ.npz
(978, 224, 224, 3)
(978,)
/content/drive/My Drive/Colab Notebooks/datas/அ.npz
(996, 224, 224, 3)
(996,)
/content/drive/My Drive/Colab Notebooks/datas/ஆ.npz
(984, 224, 224, 3)
(984,)
/content/drive/My Drive/Colab Notebooks/datas/இ.npz
(980, 224, 224, 3)
(980,)
/content/drive/My Drive/Colab Notebooks/datas/ஈ.npz
(944, 224, 224, 3)
(944,)
/content/drive/My Drive/Colab Notebooks/datas/உ.npz
(978, 224, 224, 3)
(978,)
/content/drive/My Drive/Colab Notebooks/datas/ஊ.npz
(986, 224, 224, 3)
(986,)
/content/drive/My Drive/Colab Notebooks/datas/எ.npz
(979, 224, 224, 3)
(979,)
/content/drive/My Drive/Colab Notebooks/datas/ஏ.npz
(989, 224, 224, 3)
(989,)
/content/drive/My Drive/Colab Notebooks/datas/ஐ.npz
(948, 224, 224, 3)
(948,)
/content/drive/My Drive/Colab Notebooks/datas/ஒ.npz
(936, 224, 224, 3)
(936,)

(10698, 224, 224, 3)
(10698,)
11

After finding the equality rate and storing the augmented images in numpy format, the datasets are preprocessed.

Sample Preprocessed Datasets



VCG Model 16 and 19

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg19 (Model)                (None, 7, 7, 512)         20024384
_________________________________________________________________
flatten_2 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_4 (Dense)              (None, 256)               6422784
_________________________________________________________________
dense_5 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 11)                1419
=================================================================
Total params: 26,481,483
Trainable params: 6,457,099
Non-trainable params: 20,024,384
```

Loss Function is Calculated

Sample Calculations:

Train on 8023 samples, validate on 2675 samples
Epoch 1/30
8023/8023 [==============================] - 38s 5ms/step - loss: 0.3422 - acc: 0.8967 - val_loss: 0.0590 - val_acc: 0.9836

Epoch 00001: val_loss improved from inf to 0.05900, saving model to /content/drive/My Drive/Colab Notebooks/avggz-01-0.3422-0.8967-0.0590-0.9836.h5
Epoch 2/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.1030 - acc: 0.9710 - val_loss: 0.0371 - val_acc: 0.9903

Epoch 00002: val_loss improved from 0.05900 to 0.03714, saving model to /content/drive/My Drive/Colab Notebooks/avggz-02-0.1030-0.9710-0.0371-0.9903.h5
Epoch 3/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0504 - acc: 0.9879 - val_loss: 0.0251 - val_acc: 0.9944

Epoch 00003: val_loss improved from 0.03714 to 0.02505, saving model to /content/drive/My Drive/Colab Notebooks/avggz-03-0.0504-0.9879-0.0251-0.9944.h5
Epoch 4/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0354 - acc: 0.9902 - val_loss: 0.0208 - val_acc: 0.9940

Epoch 00004: val_loss improved from 0.02505 to 0.02076, saving model to /content/drive/My Drive/Colab Notebooks/avggz-04-0.0354-0.9902-0.0208-0.9940.h5
Epoch 5/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0206 - acc: 0.9949 - val_loss: 0.0195 - val_acc: 0.9948

Epoch 00005: val_loss improved from 0.02076 to 0.01953, saving model to /content/drive/My Drive/Colab Notebooks/avggz-05-0.0206-0.9949-0.0195-0.9948.h5
Epoch 6/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0147 - acc: 0.9966 - val_loss: 0.0122 - val_acc: 0.9974

Epoch 00006: val_loss improved from 0.01953 to 0.01222, saving model to /content/drive/My Drive/Colab Notebooks/avggz-06-0.0147-0.9966-0.0122-0.9974.h5
Epoch 7/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0165 - acc: 0.9954 - val_loss: 0.0180 - val_acc: 0.9940

Epoch 00007: val_loss did not improve from 0.01222
Epoch 8/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0133 - acc: 0.9956 - val_loss: 0.0138 - val_acc: 0.9948

Epoch 00008: val_loss did not improve from 0.01222
Epoch 9/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0160 - acc: 0.9949 - val_loss: 0.0158 - val_acc: 0.9951

Epoch 00009: val_loss did not improve from 0.01222
Epoch 10/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0114 - acc: 0.9968 - val_loss: 0.0203 - val_acc: 0.9944

Epoch 00010: val_loss did not improve from 0.01222
Epoch 11/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0056 - acc: 0.9989 - val_loss: 0.0284 - val_acc: 0.9925

Epoch 00011: val_loss did not improve from 0.01222
Epoch 12/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0055 - acc: 0.9986 - val_loss: 0.0233 - val_acc: 0.9936

Epoch 00012: val_loss did not improve from 0.01222
Epoch 13/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0038 - acc: 0.9989 - val_loss: 0.0205 - val_acc: 0.9959

Epoch 00013: val_loss did not improve from 0.01222
Epoch 14/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0038 - acc: 0.9988 - val_loss: 0.0268 - val_acc: 0.9921

Epoch 00014: val_loss did not improve from 0.01222
Epoch 15/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0080 - acc: 0.9970 - val_loss: 0.0165 - val_acc: 0.9970

Epoch 00015: val_loss did not improve from 0.01222
Epoch 16/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0021 - acc: 0.9996 - val_loss: 0.0181 - val_acc: 0.9955

Epoch 00016: val_loss did not improve from 0.01222
Epoch 17/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0016 - acc: 1.0000 - val_loss: 0.0167 - val_acc: 0.9963

Epoch 00017: val_loss did not improve from 0.01222
Epoch 18/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0058 - acc: 0.9980 - val_loss: 0.0240 - val_acc: 0.9925

Epoch 00018: val_loss did not improve from 0.01222
Epoch 19/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0178 - acc: 0.9944 - val_loss: 0.0464 - val_acc: 0.9873

Epoch 00019: val_loss did not improve from 0.01222
Epoch 20/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0086 - acc: 0.9971 - val_loss: 0.0219 - val_acc: 0.9944

Epoch 00020: val_loss did not improve from 0.01222
Epoch 21/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0087 - acc: 0.9973 - val_loss: 0.0218 - val_acc: 0.9948

Epoch 00021: val_loss did not improve from 0.01222
Epoch 22/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0291 - acc: 0.9921 - val_loss: 0.0478 - val_acc: 0.9869

Epoch 00022: val_loss did not improve from 0.01222
Epoch 23/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0149 - acc: 0.9951 - val_loss: 0.0396 - val_acc: 0.9925

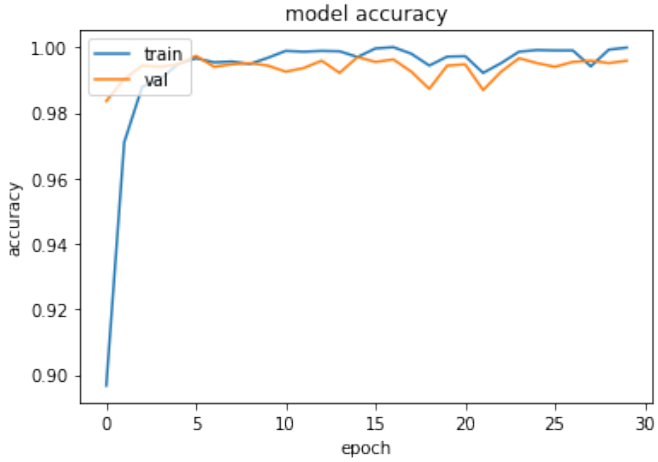
Epoch 00023: val_loss did not improve from 0.01222
Epoch 24/30
8023/8023 [==============================] - 35s 4ms/step - loss: 0.0059 - acc: 0.9986 - val_loss: 0.0206 - val_acc: 0.9966

Epoch 00024: val_loss did not improve from 0.01222
Epoch 25/30
8023/8023 [==============================] - 34s 4ms/step - loss: 0.0025 - acc: 0.9991 - val_loss: 0.0275 - val_acc: 0.9951

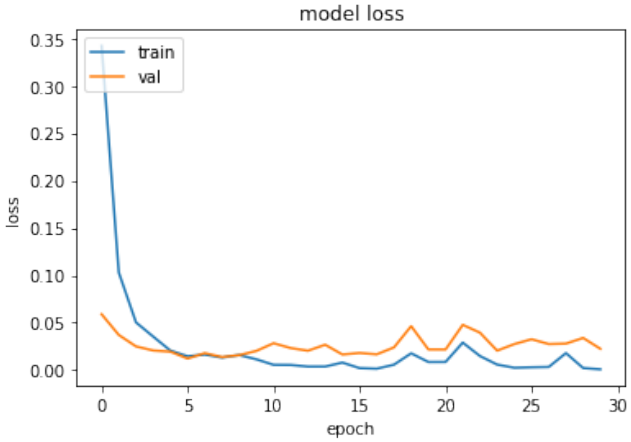
Epoch 00025: val_loss did not improve from 0.01222
Epoch 26/30
8023/8023 [==============================] - 34s 4ms/step - loss: 0.0030 - acc: 0.9990 - val_loss: 0.0326 - val_acc: 0.9940

Epoch 00026: val_loss did not improve from 0.01222
Epoch 27/30
8023/8023 [==============================] - 34s 4ms/step - loss: 0.0034 - acc: 0.9990 - val_loss: 0.0276 - val_acc: 0.9955

Epoch 00027: val_loss did not improve from 0.01222
Epoch 28/30
8023/8023 [==============================] - 34s 4ms/step - loss: 0.0180 - acc: 0.9941 - val_loss: 0.0280 - val_acc: 0.9959



Graph 1. Finding model accuracy and validation accuracy

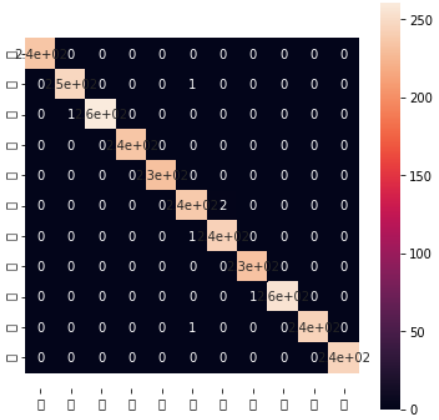


Graph 2. To find validation loss and model loss

```
2675/2675 [==============================] - 10s 4ms/step
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       235
           1       1.00      1.00      1.00       248
           2       1.00      1.00      1.00       262
           3       1.00      1.00      1.00       235
           4       1.00      1.00      1.00       232
           5       0.99      0.99      0.99       240
           6       0.99      1.00      0.99       244
           7       1.00      1.00      1.00       233
           8       1.00      1.00      1.00       256
           9       1.00      1.00      1.00       245
          10       1.00      1.00      1.00       245

    accuracy                           1.00      2675
   macro avg       1.00      1.00      1.00      2675
weighted avg       1.00      1.00      1.00      2675
```

Figure 15. To find f1 score (accuracy)

Figure 16. Confusion Matrix



NASNET

Training the model with datasets and finding the total number of parameters used for training those labels

Layer (type)	Output Shape	Param #
densenet169 (Model)	(None, 7, 7, 1664)	12642880
flatten_4 (Flatten)	(None, 81536)	0
dense_10 (Dense)	(None, 256)	20873472
dense_11 (Dense)	(None, 128)	32896
dropout_4 (Dropout)	(None, 128)	0
dense_12 (Dense)	(None, 12)	1548

Total params: 33,550,796
Trainable params: 20,907,916
Non-trainable params: 12,642,880

Train on 5400 samples, validate on 1800 samples
Loss Function is Calculated
Sample Calculations:

Epoch 1/30
5400/5400 [==============================] - 93s 17ms/step - loss: 3.6029 - acc: 0.4717 - val_loss: 1.4588 - val_acc: 0.5378

Epoch 00001: val_loss improved from inf to 1.45883, saving model to /content/drive/My Drive/Colab Notebooks/nasnet-01-3.6029-0.4717-1.4588-0.5378.h5
Epoch 2/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.6688 - acc: 0.7678 - val_loss: 1.1934 - val_acc: 0.6200

Epoch 00002: val_loss improved from 1.45883 to 1.19341, saving model to /content/drive/My Drive/Colab Notebooks/nasnet-02-0.6688-0.7678-1.1934-0.6200.h5
Epoch 3/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.4243 - acc: 0.8504 - val_loss: 1.2926 - val_acc: 0.5922

Epoch 00003: val_loss did not improve from 1.19341
Epoch 4/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.3082 - acc: 0.8870 - val_loss: 1.3678 - val_acc: 0.6033

Epoch 00004: val_loss did not improve from 1.19341
Epoch 5/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.2399 - acc: 0.9157 - val_loss: 1.2112 - val_acc: 0.6194

Epoch 00005: val_loss did not improve from 1.19341
Epoch 6/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.2127 - acc: 0.9239 - val_loss: 1.4998 - val_acc: 0.6111

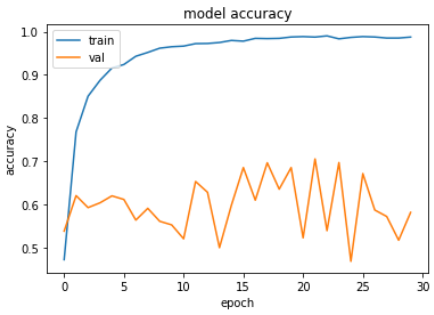
Epoch 00006: val_loss did not improve from 1.19341
Epoch 7/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.1612 - acc: 0.9430 - val_loss: 1.7445 - val_acc: 0.5633

Epoch 00007: val_loss did not improve from 1.19341
Epoch 8/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.1346 - acc: 0.9519 - val_loss: 2.0366 - val_acc: 0.5906

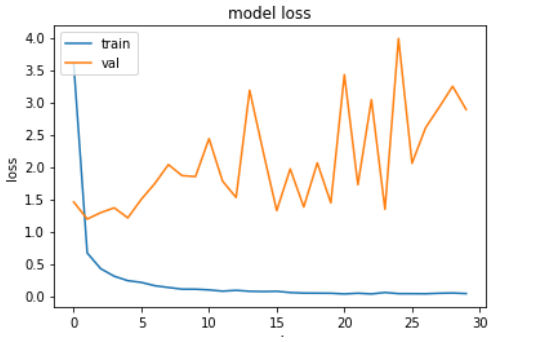
Epoch 00008: val_loss did not improve from 1.19341
Epoch 9/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.1087 - acc: 0.9617 - val_loss: 1.8663 - val_acc: 0.5606

Epoch 00009: val_loss did not improve from 1.19341
Epoch 10/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.1080 - acc: 0.9650 - val_loss: 1.8503 - val_acc: 0.5522
Epoch 00010: val_loss did not improve from 1.19341
Epoch 11/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0967 - acc: 0.9665 - val_loss: 2.4385 - val_acc: 0.5200
Epoch 00011: val_loss did not improve from 1.19341
Epoch 12/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0775 - acc: 0.9722 - val_loss: 1.7805 - val_acc: 0.6528
Epoch 00012: val_loss did not improve from 1.19341
Epoch 13/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0902 - acc: 0.9726 - val_loss: 1.5276 - val_acc: 0.6278
Epoch 00013: val_loss did not improve from 1.19341
Epoch 14/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0756 - acc: 0.9748 - val_loss: 3.1871 - val_acc: 0.4994
Epoch 00014: val_loss did not improve from 1.19341
Epoch 15/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0711 - acc: 0.9796 - val_loss: 2.2374 - val_acc: 0.5978
Epoch 00015: val_loss did not improve from 1.19341
Epoch 16/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.0753 - acc: 0.9780 - val_loss: 1.3248 - val_acc: 0.6850
Epoch 00016: val_loss did not improve from 1.19341
Epoch 17/30
5400/5400 [==============================] - 68s 13ms/step - loss: 0.0561 - acc: 0.9844 - val_loss: 1.9685 - val_acc: 0.6094

Epoch 00017: val_loss did not improve from 1.19341
Epoch 18/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.0474 - acc: 0.9839 - val_loss: 1.3805 - val_acc: 0.6961
Epoch 00018: val_loss did not improve from 1.19341
Epoch 19/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.0465 - acc: 0.9844 - val_loss: 2.0625 - val_acc: 0.6350
Epoch 00019: val_loss did not improve from 1.19341
Epoch 20/30
5400/5400 [==============================] - 66s 12ms/step - loss: 0.0452 - acc: 0.9876 - val_loss: 1.4447 - val_acc: 0.6850
Epoch 00020: val_loss did not improve from 1.19341
Epoch 21/30
5400/5400 [==============================] - 67s 12ms/step - loss: 0.0343 - acc: 0.9885 - val_loss: 3.4286 - val_acc: 0.5222
Epoch 00021: val_loss did not improve from 1.19341
Epoch 22/30
5400/5400 [==============================] - 65s 12ms/step - loss: 0.0456 - acc: 0.9874 - val_loss: 1.7253 - val_acc: 0.7050
Epoch 00022: val_loss did not improve from 1.19341
Epoch 23/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0341 - acc: 0.9902 - val_loss: 3.0402 - val_acc: 0.5389
Epoch 00023: val_loss did not improve from 1.19341
Epoch 24/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0571 - acc: 0.9833 - val_loss: 1.3445 - val_acc: 0.6967
Epoch 00024: val_loss did not improve from 1.19341
Epoch 25/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0390 - acc: 0.9867 - val_loss: 3.9883 - val_acc: 0.4678



Graph 3. Finding model accuracy and validation accuracy



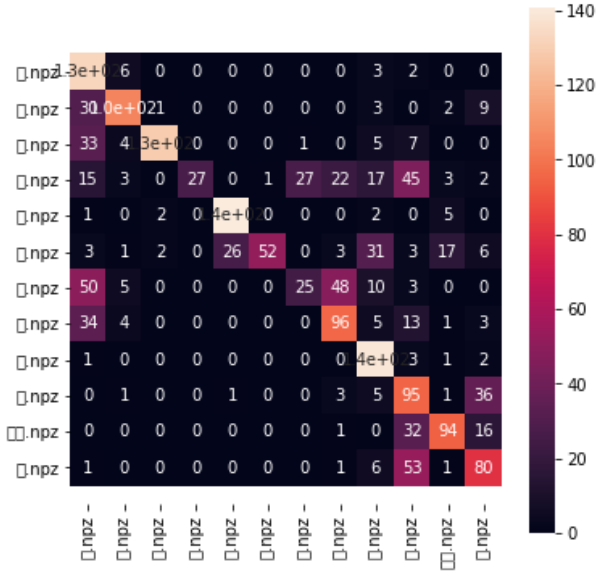
Graph 4. Finding validation loss and model loss

Figure 17. To find f1 score

1800/1800 [==============================] - 30s 17ms/step

	precision	recall	f1-score	support
0	0.44	0.92	0.60	144
1	0.81	0.70	0.75	150
2	0.96	0.72	0.82	179
3	1.00	0.17	0.29	162
4	0.84	0.93	0.88	151
5	0.98	0.36	0.53	144
6	0.47	0.18	0.26	141
7	0.55	0.62	0.58	156
8	0.62	0.95	0.75	146
9	0.37	0.67	0.48	142
10	0.75	0.66	0.70	143
11	0.52	0.56	0.54	142
accuracy			0.62	1800
macro avg	0.69	0.62	0.60	1800
weighted avg	0.70	0.62	0.60	1800

Figure 18. Confusion Matrix

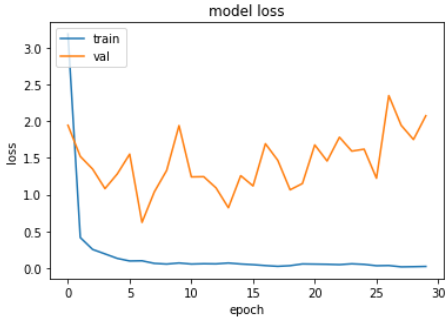


RESNET
Training the model with datasets and finding the total number of parameters used for training those labels

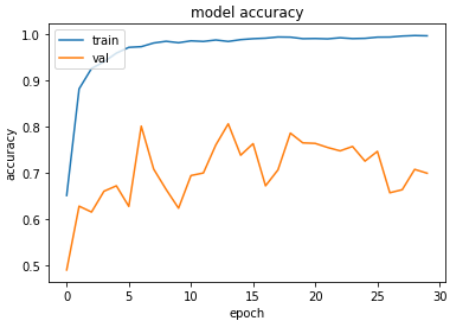
Layer (type)	Output Shape	Param #
densenet169 (Model)	(None, 7, 7, 1664)	12642880
flatten_4 (Flatten)	(None, 81536)	0
dense_10 (Dense)	(None, 256)	20873472
dense_11 (Dense)	(None, 128)	32896
dropout_4 (Dropout)	(None, 128)	0
dense_12 (Dense)	(None, 12)	1548

Total params: 73,550,796
Trainable params: 70,907,916
Non-trainable params: 12,642,880
Loss Function is Calculated
Sample Calculations:
Epoch 00025: val_loss did not improve from 1.19341
Epoch 26/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0383 - acc: 0.9885 - val_loss: 2.0569 - val_acc: 0.6711
Epoch 00026: val_loss did not improve from 1.19341
Epoch 27/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0370 - acc: 0.9876 - val_loss: 2.6068 - val_acc: 0.5867
Epoch 00027: val_loss did not improve from 1.19341
Epoch 28/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0451 - acc: 0.9852 - val_loss: 2.9216 - val_acc: 0.5717
Epoch 00028: val_loss did not improve from 1.19341
Epoch 29/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0492 - acc: 0.9852 - val_loss: 3.2483 - val_acc: 0.5167
Epoch 00029: val_loss did not improve from 1.19341
Epoch 30/30
5400/5400 [==============================] - 64s 12ms/step - loss: 0.0399 - acc: 0.9874 - val_loss: 2.8875 - val_acc: 0.5811
Epoch 00030: val_loss did not improve from 1.19341

2023.841032743454



Graph 5. Model accuracy

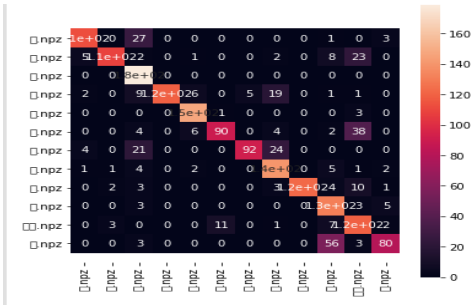


Graph 6. Model Loss

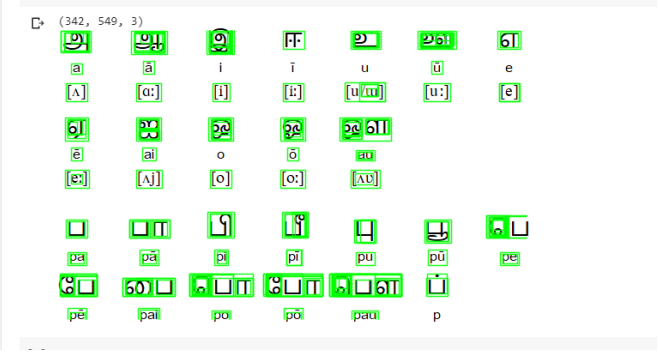
1800/1800 [==========] - 15s 8ms/step

	precision	recall	f1-score	support
0	0.90	0.78	0.84	144
1	0.95	0.73	0.82	150
2	0.70	1.00	0.82	179
3	1.00	0.73	0.85	162
4	0.91	0.97	0.94	151
5	0.88	0.62	0.73	144
6	0.95	0.65	0.77	141
7	0.73	0.90	0.80	156
8	1.00	0.84	0.91	146
9	0.61	0.92	0.73	142
10	0.59	0.83	0.69	143
11	0.86	0.56	0.68	142
accuracy			0.80	1800
macro avg	0.84	0.80	0.80	1800
weighted avg	0.84	0.80	0.80	1800

Figure 19. Confusion Matrix

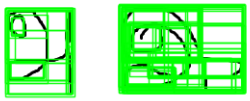


Localization:

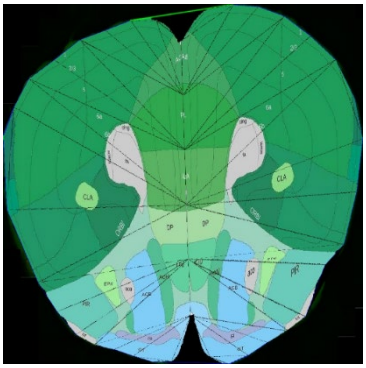


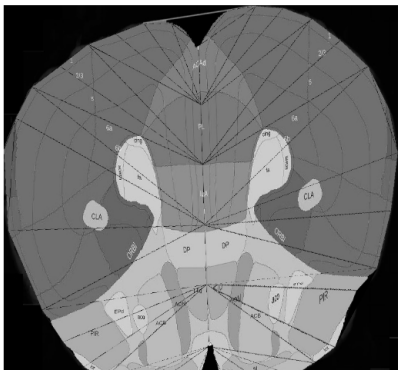

```
[ ] from google.colab.patches import cv2_imshow
    iimg = cv2.imread("/content/drive/My Drive/Colab Notebooks/T/index.png")
    ss.setBaseImage(iimg)
    ss.switchToSelectiveSearchFast()
    rects = ss.process()
    imo = iimg.copy()
    print(imo.shape)
    for i, rect in (enumerate(rects)):
      x, y, w, h = rect
      cv2.rectangle(imo, (x, y,), (x+w, y+h), (0, 255, 0), 1, cv2.LINE_AA)

    cv2_imshow(imo)
```

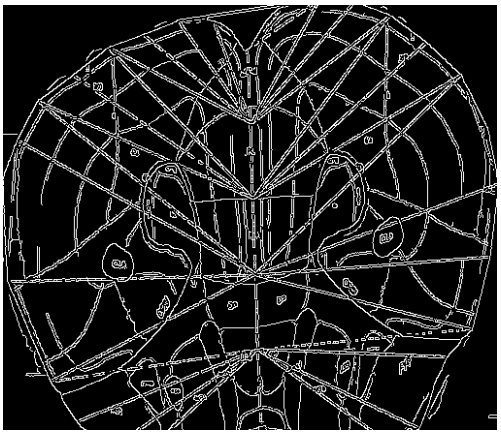
```
(148, 320, 3)
```



Detecting the features of the labels and segregating it.







The image is recognized in variety of challenging conditions and performance and accuracy is validated using cross entropy.

5. Conclusion

A lot of research work exists in the survey for Handwritten recognition. However, there is standard solution to identify all Tamil characters with reasonable accuracy. Various methods have been used in each phase of the recognition process, Challenges still prevails in the recognition of normal as well as abnormal writing, slanting characters, similar shaped characters, joined characters, curves and so on during recognition process. In this project, I have projected various aspects of each phase of the offline Tamil character recognition process. I have used maximum all character set. Coverage is not given for different writing styles and font size issues. So far I have trained the datasets with four models of CNN and character is being recognized from the word. The following key challenges can be further explored in the future. As a result, among the proposed algorithm, it has been found that different CNN models produces different result and is better which yields the highest recognition accuracy. The handwritten recognition system described in this project will find potential applications in handwritten character recognition from words. The proposed architecture has shown enhanced performance in recognizing the character.

References

[1] 'Off-line handwritten character recognition using an integrated DBSCAN-ANN scheme', Dhurgham Ali Mohammed, Alaa Abdul Hussein Mezher, HayderSabeehHadi, Indonesian Journal of Electrical Engineering and Computer Science Vol. 14, No. 3, June 2019, pp. 1443~1451 ISSN: 2502-4752, DOI: 10.11591/ijeecs.v14.i3.pp1443-1451.

[2] 'Handwritten Tamil Character Recognition and Conversion using Neural Network',C. Sureshkumar et. al. / (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 07, 2019, 2261-2267.

[3] 'Handwritten Digit Recognition Using Machine Learning Algorithms,' S. M. Shamim, Mohammad BadrulAlamMiah, AngonaSarker, MasudRana, Abdullah Al Jobair Indonesian Journal of Science & Technology 3 (1) (2018) 29-39.

[4] 'Analysis of Statistical Feature Extraction Approaches Used in Handwritten OCR' ,Antony Robert Raj.M1, Abirami.S2, Murugappan.S3,IEEE International Conference on Industrial and information systems, Page (s): 261 – 264, 2018.

[5] 'Towards Offline Arabic Handwritten Character Recognition Based on Unsupervised Machine Learning Methods: A Perspective Study',Ahmad Hasasneh1,Nael Salman2 and Derar Eleyan2,3,International Journal of Computing Academic Research (IJCAR) ISSN 2305-9184, Volume 8, Number 1 (February 2019), pp.1-8.

[6] 'An Effective Approach to Unsupervised Machine Translation',MikelArtetxe, GorkaLabaka, EnekoAgirre, International Workshop on, pages 438–443. IEEE, 2018.

[7] 'Benchmarking on offline Handwritten Tamil Character Recognition using convolutional neural networks', Kavitha B.R., Srimathi C, Journal of King Saud University – Computer and Information Sciences, 2019.

[8] 'Handwritten Character Recognition using SVM', D.K. Shamim Mohammad and C.V. Jawahar Document Analysis and Recognition (ICDAR), 2017 13th International Conference on, pp. 1041-1045.

[9] 'A novel fuzzy approach for handwritten Arabic character', M. Kef, L. Chergui, and S. Chikhi, recognition, Pattern Analysis and Applications, 2017, pp. 1-16.

[10] 'An effective approach to offline Arabic handwriting recognition, J. Al Abodi, and X. Li, Pattern Analysis and Applications, 40, 2017, pp. 1883-1901.

[11] 'High Performance Offline Handwritten Chinese Character Recognition Using GoogLeNet and Directional Feature Maps',ZhuoyaoZhong, Lianwen Jin+, ZechengXie Journal on Today's Ideas-Tomorrow's Technologies,vol. 2, no. 1, 2018.

[12] 'Deep Text-to-Speech System with Seq2Seq Model',Yuan (Gary) Wang, ", International journal of applied information system volume 7-no.2, 2018.

[13] 'Design and Implementation of Text To Speech Conversion for Visually Impaired People', ItunuoluwaIsewon ,JeliliOyelade International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA

Volume 7– No. 2,2018
[14] 'A novel SVM-based handwritten character recognition System', N. Shanthi Æ K. Duraiswamy, International Journal of Computer Applications, 64(8), 2018.
[15] 'Design and Implementation of Text To Speech Conversion for Visually Impaired Using 'iNovel Algorithm', M. Arun , S.S. Salvadiswar, J.Sibidharan, In Eighth International Conference on Spoken Language Processing,2018.
[16] 'Beyond Human Recognition: A CNN-Based Framework for Handwritten Character Recognition',Li Chen, Song Wang, Wei Fan, Jun Sun, Satoshi Naoi Fujitsu Research & Development Center, Beijing, China,2017.
[17]'Normalization-Cooperated Gradient Feature Extraction for Handwritten Character Recognition', Cheng-Lin Liu, Senior Member, IEEE IEEE Transactions on pattern Analysis and Machine Intelligence, vol. 29, no. 8, 2017.
[18]'CNN Based Common Approach to Handwritten Character Recognition of Multiple Scripts',DurjoySenMaitra, Ujjwal Bhattacharya and Swapan K. Parui Computer Vision & Pattern Recognition Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata-108, India 16th International Conference on Document Analysis and Recognition (ICDAR),2017.
[19] 'A Review on Recognition of Online Handwriting in Different Scripts', Nilakhi Saikia1, S.R. Nirmala2 IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 8, Issue 2, Ver. I (Mar.-Apr. 2018), PP 19-29 e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197.
[20] 'Text to Speech Conversion Using Flite Algorithm', ZhonGxin , NY 298743, September 2017.
[21] 'Hidden semi-Markov model based speech synthesis', Zen, H., Tokuda, K., Masuko, T., Kobayashi, T. and Kitamura, T., 2017 In Eighth International Conference on Spoken Language Processing.
[22] 'Recognition of Tamil Handwritten Characters using Daubechies Wavelet Transforms and Feed-Forward Backpropagation Network', Jose, T.M. and Wahi, A., 2017 International Journal of Computer Applications, 64(8).
[23] 'Machine Learning Algorithms with ROC curve for Predicting and Diagnosing the Heart Disease', R.Kanan ,V.Vasanthi Springer, 2017.
[24] 'Handwritten Character Recognition Using Diagonal-Based Feature Extraction', K. Vijayalakshmi,S. Aparna, Gayatri Gopal and W. Jino Hans, In Proceedings of the International Workshop on Multilingual '13, pages 16:1–16:5, New York, NY, USA, 2016. ACM.
[25]'Sequence-to-Sequence Domain Adaptation Network for Robust Text Image Recognition ',Yaping Zhang, Shuai Nie, Wenju Liu, Xing Xu, In Proceedings of the International Workshop on Multilingual OCR, pages 16:1–16:5, New York, NY, USA, 2012. ACM,2017.