LOVELYBOT – THE CHATBOT ASSISTANT

[1] Melvin Paul Jacob, [2] Debanjan Dutt, [3] Dheeraj Sankar N, [4]Om Shinde, [5]Sayantan Bhattacharya

[1234]Student, Department of Mechatronics, Lovely Professional University, Jalandhar, India
[5]Assistant Professor, Department of Manufacturing, Lovely Professional University, Jalandhar, India

Abstract: When visiting a foreign place, people are unsure about the information they receive or are often confused about processing such. Thus, a chatbot is more reliable to access correct details. This led us to make a chatbot that is focused to work in an environment where new faces are common and would help them when they stumble into a quagmire. The chatbot is made to work in Lovely Professional University's Campus (that being so the name "LovelyBot") where it can help the freshers to ask any query to get familiar with the surrounding and can also help reduce anxiety. It allows for students to log in using face recognition and also it can scan documents. The simple tasks it does makes the process quick for new people.

IndexTerms **- Machine learning, Keras, Chatbot, Natural Language Processing, Face recognition, GUI**

I. INTRODUCTION

Artificial intelligence is rapidly evolving and is available to companies as well as academia. Universities are increasingly using AI to rationalize their experiences and satisfy their digitally engaged student's needs. This wave of AI transition involves Chatbots. To provide immediate answers to questions in an environment where answer times can make an impact, institutions are using them as complementary tools in addition to admissions tools. Chatbots is inherent to our daily life from ordering take-offs to signing in for planes. A chatbot is "a computer program that encourages dialogue with human users, especially through the internet". Customer assistance may be one of the most important components of online service customer satisfaction. However, the sector is shifting toward automated chat options to offer excellent services to an ever-growing target audience, with the increase in NLP technology. Like Facebook Messenger, Slack, Skype and WhatsApp, are widely used as message applications for chatbots. The fact that chatbots save time is one of the reasons for being so famous. Bots will respond immediately to requests; they maximize customer loyalty, thus increasing the likelihood that a lead may become a transaction or approve an admission in higher education. Fast contact is essential for every higher education institution to turn potential candidates into students.

Before the beginning of a new school year, students are excited about issues concerning lectures, tuition, accommodations and similar issues. It is certainly a time for divisions to enter, and it takes a lot of time and resources to react effectively to this flood of requests. Chatbots are essentially an answer to this problem. They are unbelievably comfortable, simple to use and programmed to respond automatedly to popular requests, prevent misunderstanding and pause. At any time of day or night, students can talk to them, which is especially beneficial to foreign candidates who live in various time zones. Another aim is to serve AI bots. They collect vast volumes of data during the time they work with students and candidates. This data provides insightful information about student behaviour that can or cannot be found on the website of an institution, making it an important asset for recruiting and marketing departments as all of this information provides an effective analysis.

Life-long learning in all areas of the business is becoming increasingly popular. To achieve this notion many LMS are utilized and the rivalry between solutions of this type is significantly increasing. The key benefit of such a program is the usage of AI. This enables training to be personalized, for instance, to construct customized educational pathways, identify strengths and weaknesses in knowledge fields between students and generate unique growth goals. The application of AI is not restricted to personalized training, though. Chatbots will become more and more important in the coming years. For instance, conventional search engines and social networking sites can be substituted. The advantages of bots include their easy engagement, their reaction speed and their ability to set them for users. The use of a bot makes interacting with services very easy and provides a global interface. In the so-called smart learning system - algorithms that replicate instructor behaviour, chatbots play an especially important role. Through the analysis of their responses, feedback and individualized development plan these applications may measure students' skills.

The fact is that education program administrators spend between 70 and 80% of their time on repeated chores, including polls, reminders and replies to the same mail-sending inquiries [1]. In current circumstances, chatbots are easier to automate such a procedure. They can handle simple and more complicated organizational problems. Chatbots are available to take care of various activities such as alert recalls and putting up new information notifications, continuous education or engagement with students, data gathering and performance assessments and personalized training.

This research aims to make a chatbot for university campuses to assist the students in problems faced by them. The chatbot can help the students get fast and accurate responses. The chatbot has an interactive graphic user interface constructed in Tkinter. The chatbot gives multiple choices i.e., either a guest or a student to open various options according to what the user has selected. Also, with the help of machine learning a chatbot is created which operates both verbally and non-verbally where an individual can speak with the chatbot or can type in their queries.

II. MATERIALS AND METHODS

This section provides information on various components of our proposed work. The component consists of the following:

1. GUI
2. Conversational Chatbot
3. Document Scanner

4. Face Detector

The following are the main libraries used in this project:

- **NLTK (Natural Language Toolkit)**- This toolkit is one of the most popular NLP libraries that contains packages to make machines understand human language and respond to it with an appropriate response. Tokenization, Stemming, Lemmatization, Punctuation, Character Count, Word Count are some of the modules present in this library. It is a very helpful in-text pre-processing. [2]
- **Keras** - Keras is an application programming interface (API) for people, not robots. It provides reliable basic APIs; minimizes the number of user activities required for typical use cases and provides transparent and operating error notices. Keras meets best practices for cognition load reduction. Keras is a platform that is built over TensorFlow 2.0, which can be extended to massive clusters of graphics cards. [3]
- **TensorFlow** - TensorFlow is an platform for machine learning. It features a broad and flexible ecosystem of tools, modules and community resources, allowing academics to push the latest developments in ML and developing ML powered applications simply and deployed.[4]
- **Tkinter** - The Tkinter module in Python's basic toolkit GUI. On several platforms of Unix as well as on Windows machines, Tk and Tkinter are accessible. [5]
- **Pytesseract** - Python-tesseract is a python optical character recognition (OCR) tool. That is, the language contained in the images will be recognized and read.

II.1. GUI

The GUI for the application was designed in Tkinter. It consists of mainly four output screens, the login, the map, the chatbot interface and document input as shown in figure 1.

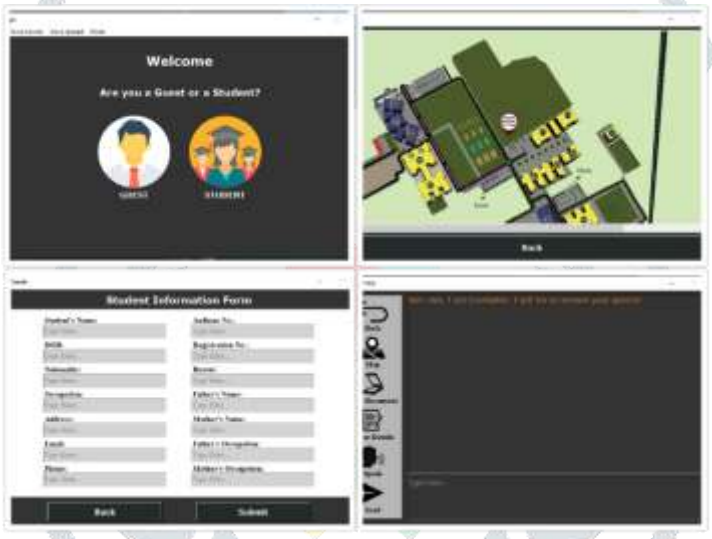

Figure 1 GUI Interface

1. The login page, this window lets you select the audience, i.e., student or guest. The student can log in with their university id and password. The guest portion is clickable and will be redirected to a guest page.
2. The interactive main page, this page is the main menu of the app it lets you upload the scanned documents, enter student details show the map and lets you interact with the bot verbally or textually.
3. The scrollable map, is a map that shows the whole university which both students and guests would be able to refer to.
4. The document input screen allows users to input relevant information into the college database directly.

II.2. Conversational Chatbot

The proposed conversational chatbot will be trained on the dataset. We use a **Domain Specific Retrieval-Based NLP Chatbot** model where multiple hidden layers are used in DNN (deep neural networks) with ReLu activation function in the neurons of the hidden dense layers to classify which category or tag the user's message belongs to and then we will give a random response from the list of responses. It's commonly used in the industry to make goal-oriented chatbots where we can modify the atmosphere and flow of the chatbot to drive our consumers with the finest experience. Figure 2 given below shows the relationships between the various components of a conversational chatbot.

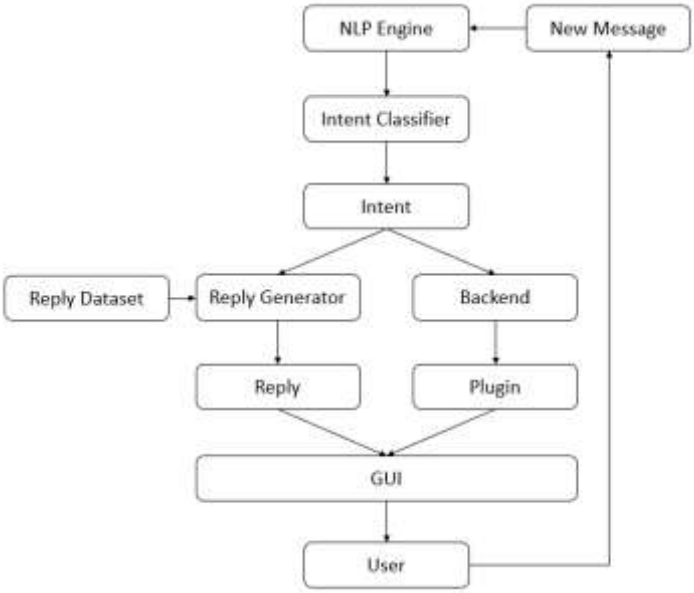

Figure 2 Chatbot Schematic diagram

II.2.1. Dataset

The dataset we are going to use is 'intents.json.' This is a JavaScript Object Notation (JSON) file that contains the questions we might get from the user and the answers we want to return to the user. The file contains different categories of question and answers that are separated into different tags, for example, greetings, goodbye, etc. The questions that the user might ask are referred to as the patterns, and answers to those questions are referred to as the responses. If a tag has a context component it refers to a tag to which that tag is associated. A dataset sample is shown in figure 3.

```
{"intents": [
        {"tag": "greeting",
         "patterns": ["Hi", "Hi there", "Is anyone there ?"
         "responses": ["Hello, thanks for visiting", "Good
         "context": ""
        },
        {"tag": "Weather",
         "patterns": ["what is the weather today ?", "How
         "responses": ["The weather is "],
         "context": ""
        },
        {"tag": "name",
         "patterns": ["what is your name ?", "what's your
         "responses": ["My name is LovelyBot"],
         "context": ""
        },
```

Figure 3 Dataset Sample

II.2.2. Text Pre-Processing

Before applying the raw text to a machine learning or a deep learning model, we need to prepare data for modelling in a few ways: by separating words, managing punctuation and instances, and more. Cleaning up text data in the NLP is a particular task. We perform the following for this conversational chatbot we are building:

a. **Removing Contractions-** Expanding contractions may be helpful at this stage, as our word tokenizer can break words like "don't" into "do not". It is not difficult to correct this tokenization at a later date but to do so beforehand makes it easier and simpler.
   Example: I'd -> I would,
   don't -> do not

b. **Tokenization-** is the process of separating a given text into constitute words—i.e. separating a phrase, a sentence, a paragraph, or a whole text document into small parts, such as particular words or terms. Each of these smaller groups is referred to as a token. Tokenization can be achieved manually by separating based on blank space or using dedicated software in libraries such as NLTK.
   Example: "Great! I won a prize." changes to ['Great', '!', 'I', 'won', 'a', ' prize ', '.'].

c. **Removing stop words and special characters-** A stop word is a widely used word (such as "the", "a", "an", "in") that the search engine has been configured to ignore, both when indexing search entries and when retrieving them as a result for a search query.
   We wouldn't want these terms to take up too much space in our database or take up precious computation cost. For this, we can easily delete them by storing a list of terms that you consider stopping words. NLTK in python includes a list of stop words stored in 16 different languages. The same can be said about special characters (such as '!', "", '#', '$').

d. **Replacing all integers with their textual representation–** In this step integers like 4, 5 are converted to their textual representation that is, four and five respectively, this allows the system to have a normalized input.

e. **Lemmatization-** Lemmatization is a common technique of standardization in pre-processing text. Another text normalization technique close to this one is called stemming. This is also achieved alongside manual tokenization to provide usable tokens. It helps get words to their dictionary form. Lemmatization is extended by default to nouns. It is more effective because it uses more educated research to construct groups of terms with similar meanings depending on the context, making it more complicated and time-consuming. This is used where the contextual information needs to be preserved. For example, words like connect, connected connection, etc. are changed to their root form, i.e. connect.

f. **Vocabulary building-** Our next step is to create a vocabulary, which is a collection of words in a given dataset, which is also known as a bag of words. It is called a word pack since all details regarding the order or arrangement of words in the document is ignored and the model is involved only with whether the recognized words appear in the document, not where they appear in the document. It has two purposes: a dictionary of known words and a measurement of the existence of known words.
The idea behind the word bag is that documents are identical if they have similar information. Even, from its contents alone, we can learn much about the context of the text. This is going to be very useful during dataset vectorization.

g. **Data Vectorization–** This step utilizes the bag of words and converts each word from each response to an array of ones and zeros the word in the bag of words is converted into one if it is present in the response and zero if not.
For example, if our bag of words contains the words [training is, never, great], and we want to vectorize the sentence "training is great," we will have the following vector: [1, 1, 0, 1].
This array is prepared for each response and is accompanied by the corresponding tag.

After training the model, when the user inputs the text or sentence, the text undergoes the same pre-processing techniques before it is computed by the model. In case the input is verbal it is first converted into text.

II.2.3. Model Architecture

Once we have our training data ready, now we will build a deep neural network (dnn) that has three dense layers with dropouts. We use the Keras sequential API for this. A dense layer feeds all output from the preceding layer to all of its neurons, each neuron delivering one output toward the next layer. In the proposed model the dense layers have 848, 720, 71 neurons respectively. Here 71 is the number of tags or classes. The model was compiled using an SGD optimiser. Stochastic gradient descent (SGD) is an iterative process for optimising objective function with sufficient smoothness characteristics. In figure 4 a model representation is shown.

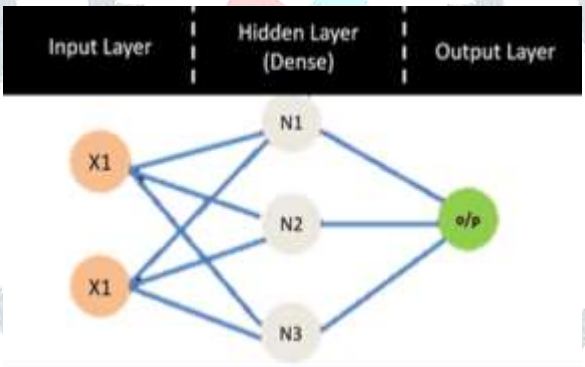

Figure 4 Model Representation

Additionally, dropouts are also provided after the first and second dense layer. Dropout is an easy way to avoid the over-fitting of neural networks. Since the outputs of the layer under dropout are randomly subsampled, effectively reducing the capability or thinning the network while training. As such, a larger network, e.g. more nodes or neurons, might be needed when using a drop-out. The following is the final model summary. Model architecture is shown in figure 5 given below.

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_9 (Dense)              (None, 848)               219632
_________________________________________________________________
dropout_6 (Dropout)          (None, 848)               0
_________________________________________________________________
dense_10 (Dense)             (None, 720)               611280
_________________________________________________________________
dropout_7 (Dropout)          (None, 720)               0
_________________________________________________________________
dense_11 (Dense)             (None, 71)                51191
=================================================================
Total params: 882,103
Trainable params: 882,103
Non-trainable params: 0
_________________________________________________________________
```

Figure 5 Model Architecture

II.2.4. Hyperparameter optimization

Machine learning algorithms have been commonly used in several applications and areas. To fit a machine learning model into various problems, its hyper-parameters must be adjusted. Hyperparameters can include items like learning rate, number of epochs, or the number of hidden layers. Selecting the right hyper-parameter configuration for machine learning models

has a significant effect on the efficiency of the model. Deep knowledge of machine learning algorithms and effective hyperparameter optimization techniques are also needed. Although there are many automated optimization strategies available they have different strengths and disadvantages when applied to different types of problems.[6] In this paper, optimizing the hyper-parameters of the proposed model is done using the Hyperband tuning algorithm in Keras-tuner and the before mentioned architecture is a result of that optimization.

II.2.5. Training and Evaluation

After training the model for 400 epochs, we achieved a 100% training accuracy on our model and a loss of 0.00895. The loss is measured based on training and its interpretation is dependent on how well the model is doing. It is the sum of the mistakes made in the training sets for each case. Loss value means how badly or well the model behaves after each iteration of optimization. Greater loss value suggests poor model behaviour and accuracy. Ideally, it needs to be zero as shown in the graph in figure 6.

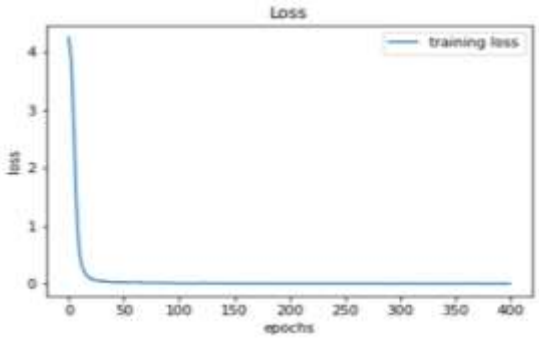

Figure 6 Loss vs Epochs

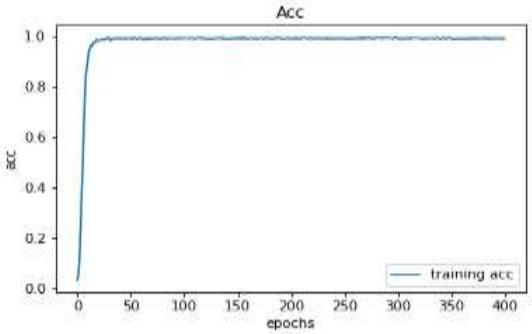

Figure 7 Accuracy vs Epochs

The accuracy metric is used to calculate the output of the algorithm in an interpretable manner. The accuracy of the model is generally determined by the model parameters and is measured in the percentage form. It is a measure of how accurate the prediction of your model is relative to the actual data. It must be as near as possible to 100% as shown in the graph in figure 7.

II.3. Face Recognition

The face of this system is the principal entity. The system's initial response is to take pictures of the face. The image is then transformed to the grayscale image from the RGB picture, which reduces the image data by three times. In this way, the image is processed using the Haar Cascade method in which the face is recognized by this process. A green box is used to mark the face when the face is identified. The cycle of differentiating the face of non-face objects is completed. The word Haar represents, Haar wavelet which is a box-math function, the same concept as the Fourier function. At first, the processing of pictures was only achieved by looking at the RGB value per pixel. Viola and Jones later established the Haar- feature extraction method which improved the face detection process. Digital image attributes used in object recognition are Haar features. Figure 8 depicts steps involved in face detection.

Figure 8 Steps in face detection [7]

The four main steps in the Haar cascade are Haar features extraction, integral images concept, Adaboost training and cascading of the classifier. Haar- function stores images in frames, where many pixels are in a row. Each box is analyzed and create a difference in value that showed the dark and luminous areas. Those values are used for image processing. Haar features extraction uses this stored data. The value of this feature is calculated by subtracting the pixel value by pixels in the black region on the white side. Integral images definition is used to simplify the process of measuring the value of a function. An integral pixel value is the pixel quantity at the top and the pixel at the left. The image is integrated into smaller pixel processes since it begins from the left top to the right bottom, this just means that the addition of all pixels is calculated in a given rectangle using four values. The pixels that represent the edges of the rectangle in the picture given in figure 9 these properties are in the integrated picture.

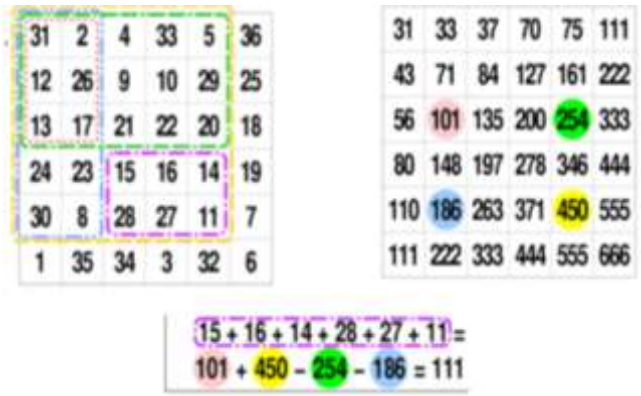

Figure 9 Integral Images concept [8]

Haar Cascades uses the AdaBoost learning algorithm to pick from a broad range of a limited number of relevant features to provide classifiers with an effective outcome. Figure 10 shows Haar features.

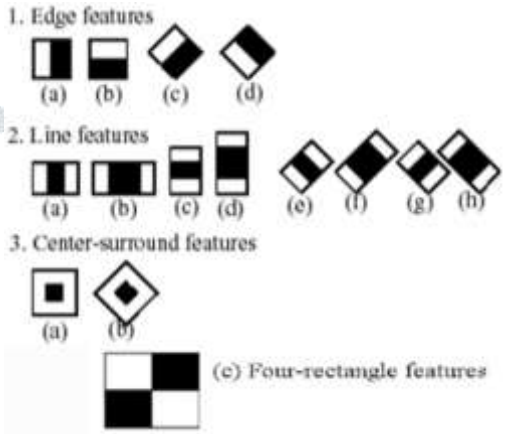

Figure 101 Haar Features [9]

Classifier cascading consists predominantly of stages consisting of a common classification for each stage. It is useful because it eliminates the need to connect all features directly to the frame. Instead, the functions are grouped into separate sub-window, and the classifier determines whether the sub-window is at each point of view. In case it does not, the sub-window and functions in that window will be discarded [10]. If the pictorial regions actually pass the filter, the next step entails a more sophisticated filter. Only regions effectively passing through all filters are considered a face contest. In other words, the regions of the photographs would be detected. The reason for the multi-stage rating is that the non-face windows are removed quickly and safely. The classifier is used to refuse the sub-windows to produce more wrong constructive areas. After multiple processing phases, the number of false positives has dropped dramatically. Figure 11 shows a cascade structure for Haar classifiers. [11]

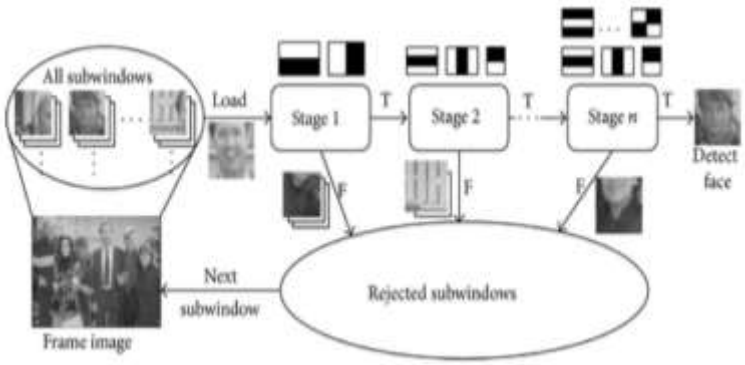

Figure 11 Cascade structure for Haar classifiers [12]

The face area is separated for facial recognition after detection of the face. Facial recognition is an innovative form of face identification. The local binary pattern histogram is used to characterize the face after the image/video algorithm is retrieved from the face. The qualified images are matched to the effects of the streaming video detection by the local binary pattern histogram (LBPH) algorithm. The histogram is based on the images used in this algorithm. The local binary pattern is an easy and efficient texturizer that marks the threshold of pixels for each pixel group and considers the output to be the binary number.

LBPH uses 4 parameters which are:

1. Radius: The central pixel radius is the radius for building the local circular binary pattern. Normally the radius is 1.
2. Neighbours: survey figures show the circular binary distribution of the local area. The more sample points you have the higher the system cost. Normally it's eight.
3. Grid X: the cell counts horizontal figure. The thinner the grid, the larger the dimension of the corresponding vector. Normally it's eight.
4. Grid Y: cell number vertical orientation. The thinner the grid, the larger the dimension of the corresponding vector.

Normally it's eight. Figure 12 shows a circular LBP.

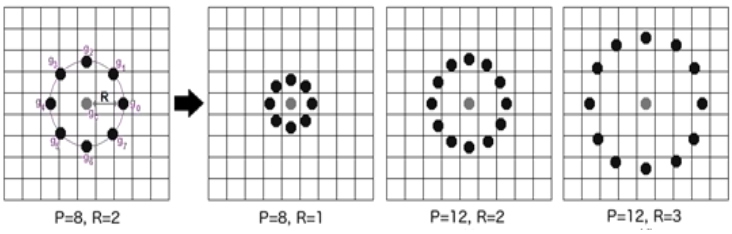

Figure 12 Circular LBP [13]

Until we can actually recognize faces, we need to train the algorithm on a faces dataset that we need to classify and each of these faces gets a unique ID so that the algorithm uses this knowledge is used to identify an input image and to deliver data to you. The same individual's photographs must be equally identified. LBPH produces the intermediary representation that best resembles the real, by emphasizing the features of the facial. To do this, the algorithm uses the sliding window definition based on the radius and surrounding parameters. Figure 13 shows us the LBP operation being applied.

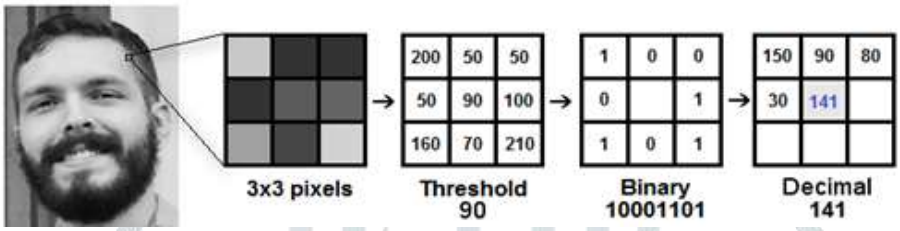

Figure 13 Applying the LBP operation [13]

Suppose we have a facial image in grayscale. To get part of this image as a pixel window, the LBP technique uses a different number of radii and neighbours. Let the radius be one so we've got 8 neighbours in a 3x3 slot. This can also be depicted as a 3x3 matrix containing pixels varying from 0 to 255 which is the strength or intensity of each pixel. We take the matrix's central value that should be used as the threshold. The new values for the 8 neighbours are specified with this value. We set a new binary value for each neighbour as per the threshold, 1 for values equal to or higher than the threshold and 0 for values below the threshold [10]. The matrix only has binary values, since the centre value is disregarded and these values will be transformed into a line matrix, e.g. 10001101. It can be achieved in the form of clockwise or anti-clockwise but the end result would be the same.

This binary value is transformed to a decimal component, which is a pixel from the original image and is set to the central value of the matrix. We have a new image at the end of LBP which better reflects the characteristics of the original image [14]. The vertical axis in the histogram will be labelled with the pixel value of the adjacent points and the horizontal axis will be indicated with the number for the neighbour.

Using the image that was generated in the last stage, the image is divided into multiple grids using Grid X and Grid Y parameters and the histogram of each region is drawn. Since the image is in grayscale, each point in each grid would contain 256 values from 0 to 255 representing each pixel intensity, thus we can get a histogram of each grid. Then, we need to combine each histogram to make a new bigger histogram. Then, each histogram must be combined, to create a new, larger histogram. Let there be 16x16 grids, then the final histograms, we'll have will be 16x16x256=65.536 locations. In figure 14 the final histogram shows the features of the original picture.

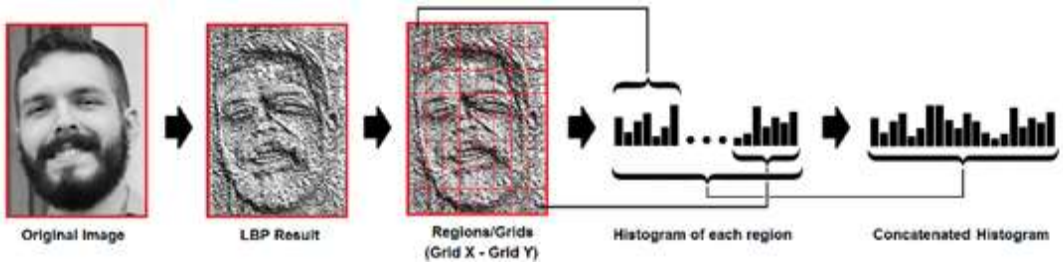

Figure 14 Extracting the Histograms [13]

The training dataset generates a separate histogram to display each image. Such moves are performed to the image input again and a histogram representing the images is constructed. Both histograms need to be compared and an image with the nearest histogram is returned to find the image identical to that of the input. The picture ID with the closest histogram is the algorithm performance. The algorithm also gives the measured distance, which can be used as a confidence measure. More confidences suggest that the two histograms are different. Then we can build trust to determine immediately if the algorithm recognizes the image correctly or not. If your trust exceeds the specified threshold, we may presume the algorithm has recognized the image or the entity successfully.

II.4. Document Scanner

In this proposed module, the system will be able to scan the user's documents and extract information from them. The document scanning and information extraction involve the following steps:

Detecting Document

- Detecting Edges - First, we convert the RGB image into a greyscale image. Then the image is blurred to remove noise. Then Canny Edge Detection is applied to the object to find the edges.
- Finding Contour or Outline- The contour of the largest object with 4 edges is detected from the image using OpenCV functions.
- Perspective Transform - Perspective transform is applied to obtain the top-down view of the document.

Scanning Document

- Feature detection & Aligning Forms - This is achieved using the ORB function in OpenCV. ORB or Oriented FAST and Rotated BRIEF is essentially a fusion of the feature from accelerated segment test (FAST) keypoint detector and the BRIEF descriptor with several adjustments to improve performance. Using quick to find key points first from the document provided by the user and the sample document. Then the features of both documents are matched and they are aligned accordingly.
- Region of Interest (ROI) extraction - The ROIs are extracted from the document using predefined locations. The program will crop out ROI from the document and few examples are given below in figure 15.

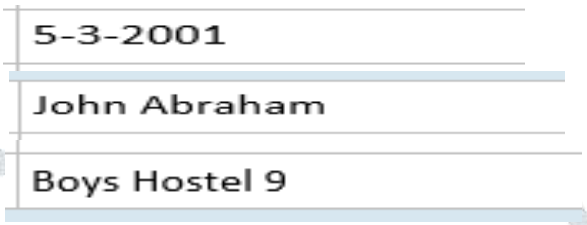

Figure 15 Region of Interests

- Text Detection and Saving - The ROIs are then fed into the OCR (Optical Character Recognition) module to detect text. This extracted information is then saved to the database. An example of a document to be scanned is given below in figure 16. The final output is given below in figure 17. It is evident from the output that text detection is not perfect and is prone to defects.

Figure 16 Sample Document

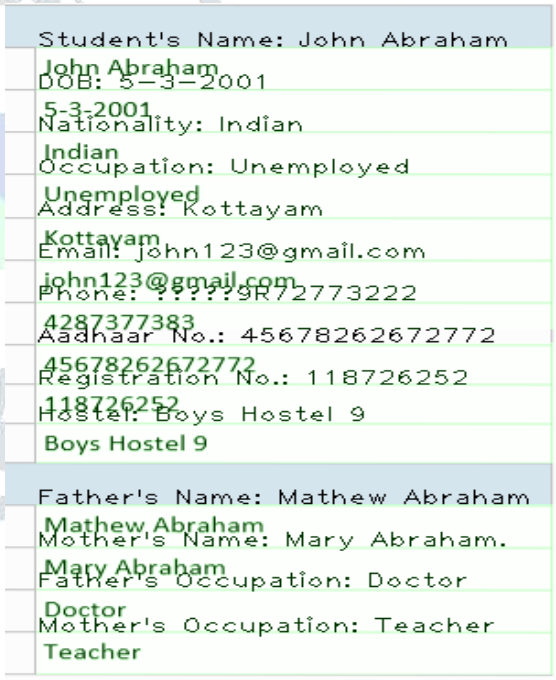

Figure 17 Output

III. CHALLENGES AND SOLUTIONS

Chatbot

The chatbot is domain-specific and can only answer a limited number of questions within its domain. This can be solved by providing a larger dataset containing lots of information. Another defect is that the chatbot's responses are predefined and cannot be changed, it selects a response from a list of appropriate ones, this issue can be solved by using a generative chatbot model instead of a retrieval based one but the issue is that the accuracy of the output decreases.

Document Scanner

We are using the Pytesseract OCR tool for our document scanner and the problem with a camera-based document scanner is that it is of low accuracy. So, the accuracy of a document scanner can be improved by increasing the existing accuracy of the OCR engine which can be achieved by:

1. Checking the quality of the image source
2. Scaling the image to the Right Size

3. Enhancing the contrast of the image
4. Removing noise from the images
5. Deskewing and analyzing page layout
6. Analyzing character edge
7. Using databases and thesaurus

Face Recognition

Motion: - For face detection we need to get a stationary head location in an image sequence which is not always possible because for a person moving in a crowd it is difficult to separate the person from the complicated background with the added difficulty of that person being in motion. When something is in motion each frame has differential intensities of pixels so to fix that we use the Gaussian smoothing method wherein we establish a base frame of reference that never changes and then process every other consequent frame with the base frame. This technique helps to smooth out the high-frequency noise which could hamper our face detection algorithm

Lighting Condition: - Variation in lighting condition poses another obstacle in image detection as it changes how the light is reflected. This is a greater challenge when compared to detecting two different faces under the same lighting condition and differentiating the same face in different lighting conditions. The lighting conditions are very critical to image detection as the same person with the same facial expression and pose can appear differently if the lighting conditions change.[10] A graphic technique used as a non-linear procedure called Gamma correction is used to control the overall exposure of a given image. Images that are not well-balanced look either overexposed or underexposed which is not a particularly good thing for image detection, thus improving detection rates.

IV. FUTURE SCOPE

As of now, our chatbot is an application. Keeping the enhancement of our chatbot in mind we plan to implement

1. The next goals to build these chatbots should be to accommodate a variety of languages to meet a broader user base.
2. From time to time, users have multiple problems that they would like to address in a single message. This changes the formulation of the intent classification problem into a multi-label classification problem. Different methods could be considered to address this issue.[15]
3. There would also be the option to request a human agent in this list to prevent users from getting frustrated with the bot when the bot is unable to answer the question or the user is unable to clarify the request.[15]
4. Create a database and add users to it.
5. Building a physical model, through which students can interact with hardware like raspberry pi, touch screen, keyboard, speakers and a body. A physical model design is shown in figure 18, which is our proposed design.
6. Manual training as hard as the system offers a huge range of training courses and regularly updates the knowledge on them and adding new topics. The change to new AI technologies, which include automated self-training on particular panels, typical replies and queries, will enable you to grow chatbot functionality for all the courses inside the system.

Figure 18 Proposed Robot Frame.

IV. RESULTS AND DISCUSSION

We have implemented a domain-specific retrieval-based NLP Chatbot which can answer questions with relatively high accuracy. The Chatbot takes in the input then processes it which is then fed into the model which determines the category with the highest probability and then picks a random answer from the database to the corresponding category and finally displays it in the output.

A document scanner is implemented through a camera using OpenCV techniques. Though it is very easy for a human being to read information off from a paper it is quite different for a machine to process that information for it to be useful, so the system does this in two main steps: first, it detects the document which includes detecting edges, finding contours and perspective transform. The second step is scanning the document which includes feature detection, aligning forms, ROI extraction, text detection and saving. The only downside is its inflexibility and low accuracy.

Finally, we have our face detection feature which follows four main steps namely Haar features extraction, Integral images concept, Adaboost training and cascading of the classifier. Due to the algorithm used the detection rate is average.

V. CONCLUSIONS

The primary goal of this paper is to propose an application. This concept arose from thinking about the challenges that university students face. A wide range of literature review is conducted to ensure continuity in all chores, where an application is deemed the best model to fulfil the requirements. Research on the application establishes a connection to learn more about the

hottest temporal innovations and incompatible algorithms such as artificial intelligence, machine learning, python, and natural language processing (NLP).

In this paper, we propose an application that can interact verbally, visually and textually with its user. There are three main features to our proposed application which includes an interactive chatbot, a document scanner and a face detector that is integrated into a GUI. All in all, the chatbot is designed to help students have instant access to information about the college, it can also guide the student to perform various first aid operations in case of emergencies and can also display a map of the college to direct students to their destination. The document scanner provided allows the students to upload documents to the college website without standing in long queues or needing to interact with the office staff. Face recognition is implemented to let the students log in faster for ease of use.

REFERENCES

[1] Fuller, R. Using a learning management system to support blended professional learning at Polytech High School. Doctoral dissertation, University of Delaware. (2016).
[2] nltk.org
[3] keras.io
[4] tensorflow.org
[5] docs.python.org
[6] Li Yang, Abdallah Shami, On hyperparameter optimization of machine learning algorithms: Theory and practice, Neurocomputing, Volume 415, 2020
[7] Melvin Paul Jacob, Comparison of Popular Face Detection and Recognition Techniques, International Research Journal of Modernization in Engineering Technology and Science Volume:03 Issue:04 April-2021, e-ISSN: 2582-5208
[8] Lienhart, R. and Maydt, J. (2002) An Extended Set of Haar-Like Features for Rapid Object Detection. 2002 International Conference on Image Processing, Vol. 1, Rochester, 22-25 September 2002, I-900-I-903.
[9] levelup.gitconnected.com
[10] Parul Pandey, "Face Detection with Python using OpenCV", www.datacamp.com
[11] Souhail Guennouni, Ali Ahaitouf, Anass Mansouri. "Face Detection: Comparing Haar-like combined with Cascade Classifiers and Edge Orientation Matching "
[12] Kim, Mooseop & Lee, Deok Gyu & Kim, Ki-Young. (2015). System Architecture for Real-Time Face Detection on Analog Video Camera. International Journal of Distributed Sensor Networks. 2015. 1-11. 10.1155/2015/251386.
[13] face recognition how lbph works, towardsdatascience.com
[14] Pratyushnair, "Facial Recognition Using Open-CV", medium.com
[15] Peters, Florian, Master thesis: Design and implementation of a chatbot in the context of customer support, matheo.uliege.be