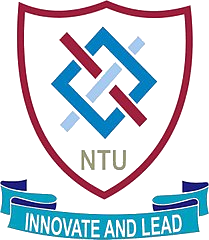
**Project Report**



**AI VOICE ASSISTANT**

**Group Details**

Abdul Rehman

19-NTU-CS-1133

M. Awais Younas

19-NTU-CS-1151

Muhammad Ali Asad

19-NTU-CS-1539

**Department of Computer Science,**

National Textile University, Faisalabad

**Title: Voice Assistant using NLP**

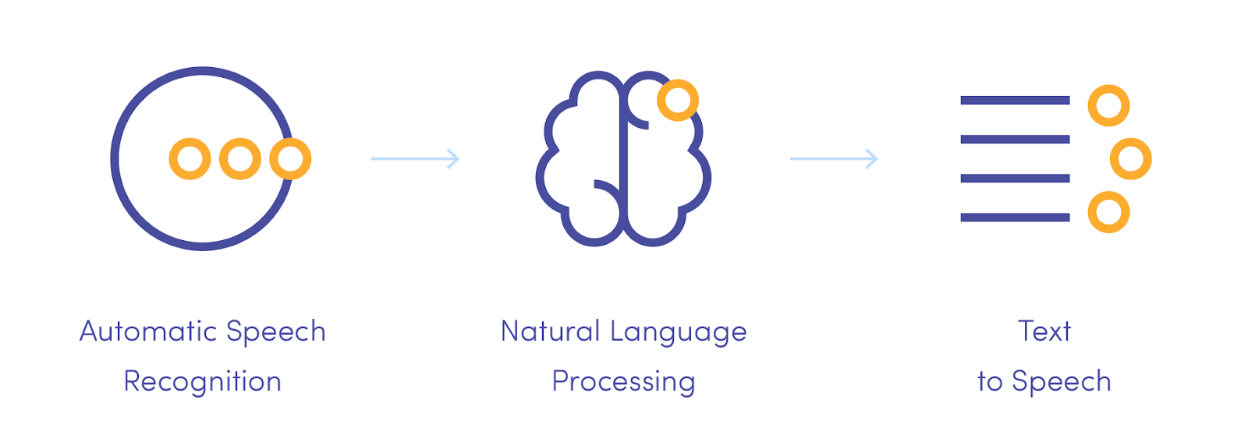
Summary:

The NLP strives to build machines that understand and respond to text or voice data with text or speech of their own. NLP combines computational linguistics with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data.

Introduction to Topic:

Python is used to create and run algorithms that enable the model to perform NLP tasks such as text classification, sentiment analysis, and language translation.

The project aims to build a highly accurate and efficient NLP model using machine learning techniques. The model is trained on a large corpus of text data to learn the patterns and relationships between words and phrases. Once trained, Jarvis ML can be used to perform a variety of NLP tasks, such as answering questions, generating text, and translating text from one language to another.



Tools and Technologies:

* **Python**

Python is a high-level, interpreted programming language that is widely used for web development, scientific computing, data analysis, artificial intelligence, and other areas.

* **NLP**

NLP stands for Natural Language Processing, which is a field of artificial intelligence and computer science that deals with the interactions between computers and human languages. NLP applications include speech recognition, machine translation, sentiment analysis, text summarization, question answering, and more.

* **Pyttsx3**

pyttsx3 is a Python library for text-to-speech (TTS) conversion. It uses the native speech engines on the user's computer to perform TTS, and supports multiple languages. It is easy to use and can be integrated into a variety of applications, such as those built using the Python programming language.

* **Natural Language Understanding:**

This AI voice assistant can understand the intent behind a user's speech and respond appropriately, making it easy for users to ask for information or perform tasks.

* **Voice Synthesis:**

This AI assistant can generate synthetic speech, allowing it to speak back to the user, providing information or completing tasks.

* **Integration with other services:**

The AI assistant can be integrated with other services like Wikipedia, Google Search and more, making it a one-stop solution for many of the user's needs.

* **VS Code**

Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft for Windows, Linux, and macOS. It supports many programming languages and has a variety of features such as debugging, source control integration, and an extensive extension marketplace.

**NLP Application Areas:**

Here are a few applications of NLP :-

* Voice Assistant
* Chatbots
* Language translation
* Search results
* Text analytics

**Implementation**

1. First step is to import the above-described libraries

from Task import NonInputExecution

from Task import InputExecution

from Speak import Say

from Listen import Listen

import random

import json

import torch

from Brain import NeuralNet

from NeuralNetwork import bag\_of\_words, tokenize

2. Next step is to input the cv library for the yolo functions and other related libraries

from pickletools import optimize

from random import shuffle

import numpy as np

import json

import torch

import torch.nn as nn

from torch.utils.data import Dataset, DataLoader

from NeuralNetwork import bag\_of\_words, tokenize, stem

from Brain import NeuralNet

3. After importing libraries we have set up to start the Chatbot using listen.py command

import speech\_recognition as sr

def Listen():

    r = sr.Recognizer()

    with sr.Microphone() as source:

        print("Listening")

        r.pause\_threshold = 1

        audio = r.listen(source, 0, 3)

    try:

        print("Recognizing ...")

        query = r.recognize\_google(audio, language="en-in")

        print(f"You Said : {query}")

    except:

        return ""

    query = str(query)

    return query.lower()

Listen()

4) After this the Assistant starts and listens

**Results and Evaluation**:

**Accuracy:**

**This measures how well the chat assistant responds to user inputs, including recognizing the intent and generating an appropriate response.**

**User Satisfaction:**

**This involves collecting feedback from users about their experience with the chat assistant, including the ease of use and the quality of responses.**

**Task Completion Rate:**

**This measures the proportion of tasks that are completed successfully by the chat assistant.**

**Response Time:**

**This measures the amount of time it takes for the chat assistant to generate a response after a user input.**

**Engagement:**

**This measures the amount of interaction between users and the chat assistant, including the number of messages exchanged and the duration of the interaction.**