Cluster Innovation Centre
University of Delhi

Unmasking the AI chatbot

Detecting AI generated content

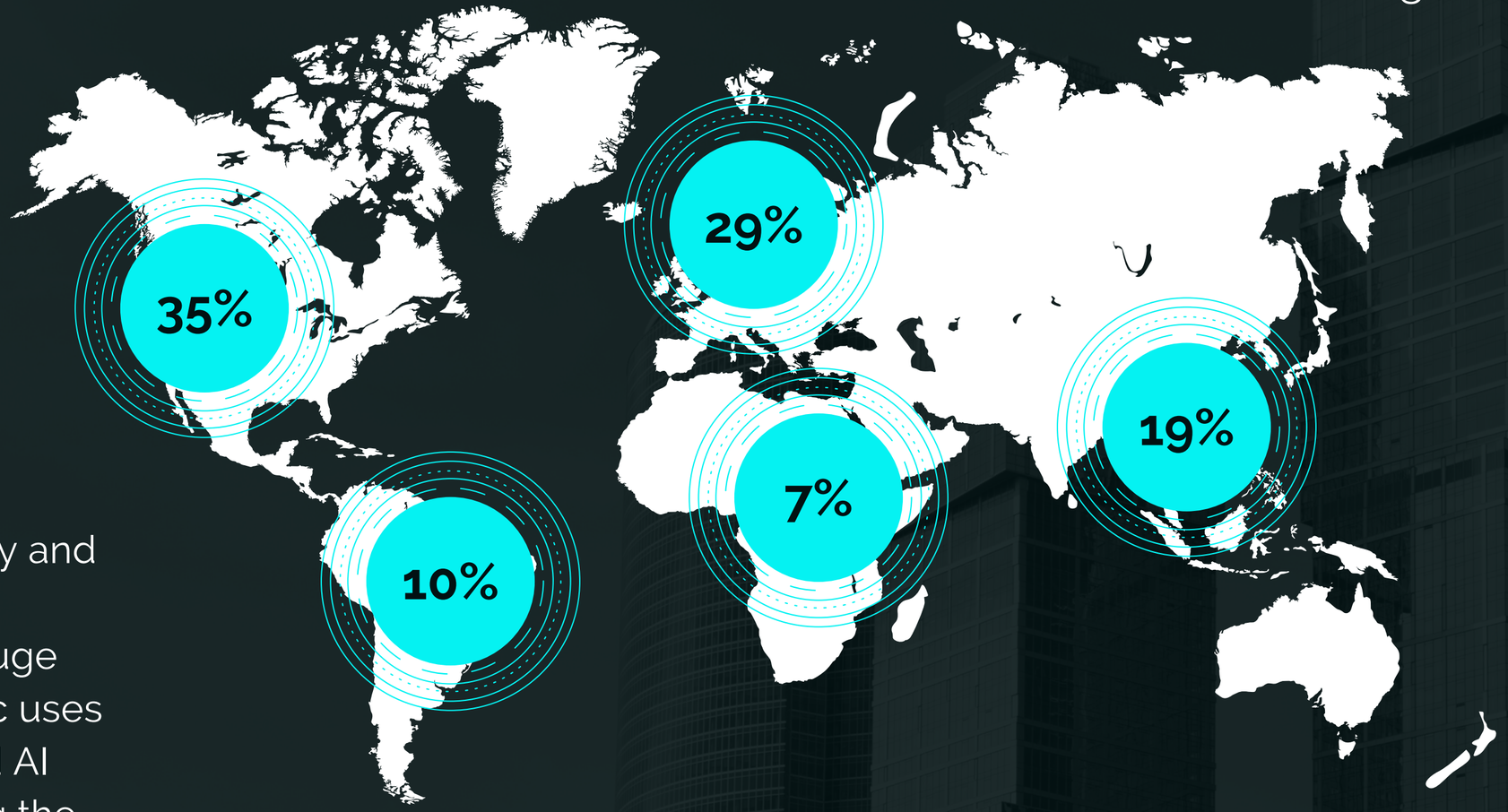
Saswat Susmoy Sahoo
Dev Mishra

Table Of Content

Need of the hour

With the advancing technology and mass adoption of the same, misutilization can become a huge problem. As the general public uses AI chatbots like ChatGPT, Bard AI etc, the chances of misutilising the resources increases.

Amount of fake news circulating



AI Chatbot users in the world

North A	South A	Africa	Asia	Europe
36%	24%	16%	28%	32%

** We don't verify or claim the data to be correct. Data produced by Google Bard AI

Our AI model

We started out designing this model using various Python Libraries like Numpy, Pandas, Scikit Learn, Tensorflow, Keras etc.

We used Numpy and Pandas for Data Handling purposes and Tensorflow with Keras for importing LSTM / Logistic Regression models. Scikit Learn was used to train and test our model and also to check our accuracy.

To implement concepts of OOPs we designed our GUI using PyQT5 which is based on Python Programming Language.

Numpy, Pandas, Scikit Learn, Keras etc

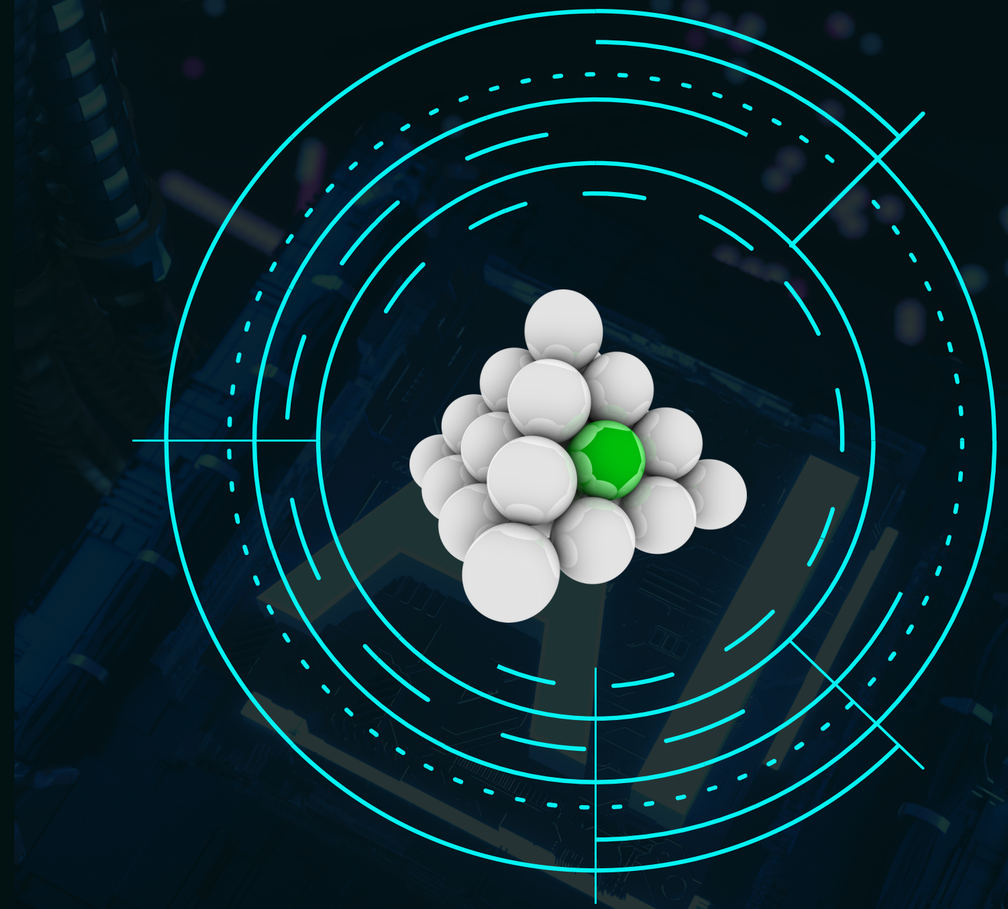
Model Used - LSTM / Logistic Regression**

** Logistic Regression isn't used in the working model rather it was used to compare accuracies in the early stage of developement of this project

Key Features

Our Model uses Deep Learning concepts of LSTM which produces quite accurate results in the fields of NLP and Cognitive Learning.

As our model is completely based out on Python, it will be easier and convenient for us to optimize our model further in future and also add more functionalities.



Data Used
8-10k

Accuracy
85-95 %**

** Accuracy above 90% sometimes denotes that the model is overfitting which might be in our case. So Accuracy statistics should not be considered as the sole basis to judge our project

Training and Evaluation

AI Generated

Apply page animations and transitions to your Canva presentation to emphasize

51%

Content Used

AI Generated Essays - 20%
ManaGPT prompt responses - 35%
Miscellaneous - 45%

Human generated

Apply page animations and transitions to your Canva presentation to emphasize

49%

Content Used

- Covid research abstracts 30%
- MacD consumer reviews 30%
- Medium Articles 10%
- US Prez speeches >1%
- BBC News RSS feed >1%
- Miscellaneous ~ 30%

Problems Faced

Data Handling

Due to our less exposure and knowledge in the field of Data Analytics, we found it difficult to handle such huge amount of data that we could access.

Cases of Overfitting and Underfitting

As we were starting out, we faced problems of Overfitting and Underfitting as we couldn't set our parameters to optimize our model and sometimes ample/shortage of data created the problem.

Less Knowledge

Due to less exposure and knowledge from our side in the ML field, we at times found it difficult to solve problems and understand concepts. We could not optimize the LSTM model to match our requirements properly.

Potential Application

We initiated this idea with the aim to detect AI generated content but as we entered the developement phase we realised the potential this small idea holds in various streams and fields.

Detecting Plagiarism

Versions of this model can be used to detect plagiarism in the content creating industry.

Detecting Fake News

This model can be trained on Real & Fake News data to enable it to detect fake news which can implemented in social media sites to prevent misinformation

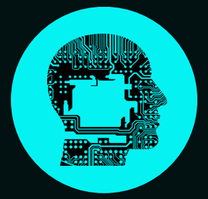
College Applications

A very accurate and precise version of this model can be used to detect fake/AI generated college essays or Letters of intent

Forensics

The model aids in identifying manipulated or tampered media, assisting in investigations and legal proceedings.

Future Work



Latest Technology

As AI algorithms improve, it becomes increasingly difficult to distinguish between AI-generated and human-created content.

We would like to explore various options to implement methods and integrate other technologies to keep our model updated as per industry standards.

Image Detection

We would like to explore the option to detect AI generated images in the future.

Codebase

```
from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QPushButton, QTextEdit, QLabel
from PyQt5.QtCore import Qt
import sys
import numpy as np
import pickle
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
from tensorflow.keras.models import load_model
```

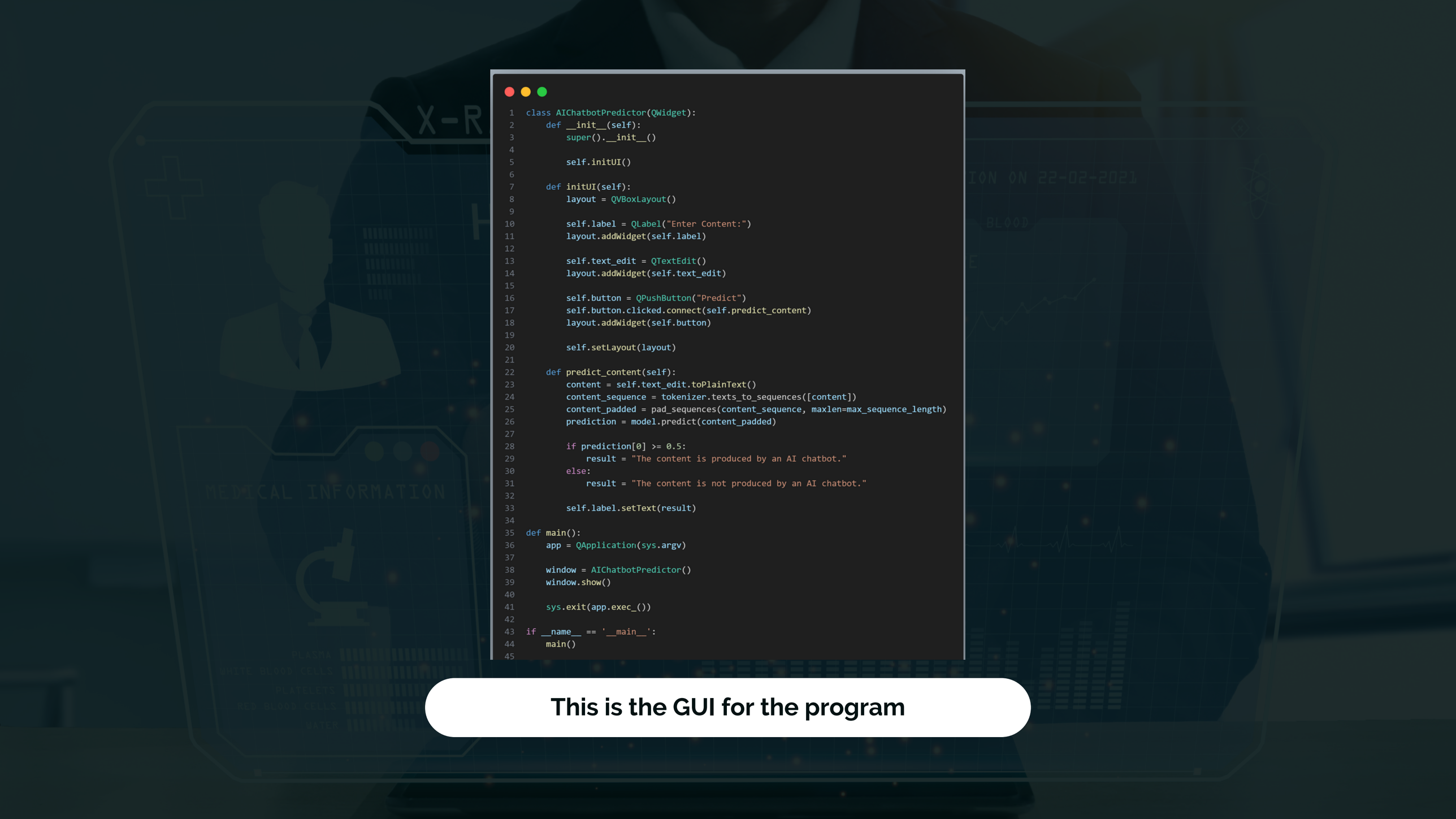
This contains all Libraries and Modules imported

```
df = pd.DataFrame(data)
loaded_model = load_model('model3_extended.h5')
# preprocessing
tokenizer = Tokenizer()
tokenizer.fit_on_texts(df['text'])
vocab_size = len(tokenizer.word_index) + 1

sequences = tokenizer.texts_to_sequences(df['text'])
max_sequence_length = max([len(seq) for seq in sequences])
padded_sequences = pad_sequences(sequences, maxlen=max_sequence_length)
```

This loads the data and load the trained model saved as "model3_extended". This also processes the data before training the model

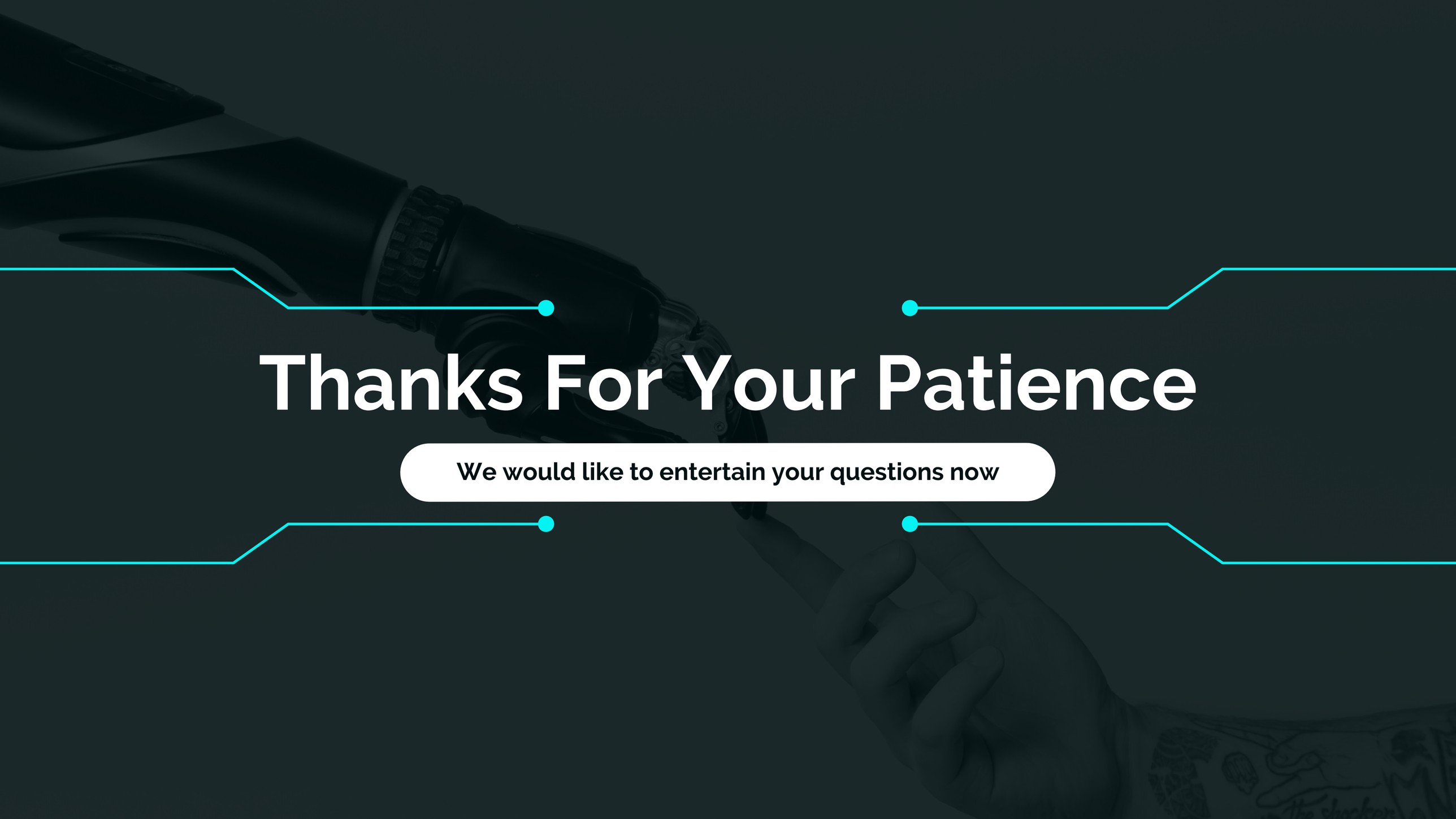
```
#dl model
model = Sequential()
model.add(Embedding(vocab_size, 100, input_length=max_sequence_length))
model.add(LSTM(128))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

This loads the Deep Learning model (LSTM) and gives various parameters to tune the same.



```python
class AIChatbotPredictor(QWidget):
    def __init__(self):
        super().__init__()

        self.initUI()

    def initUI(self):
        layout = QVBoxLayout()

        self.label = QLabel("Enter Content:")
        layout.addWidget(self.label)

        self.text_edit = QTextEdit()
        layout.addWidget(self.text_edit)

        self.button = QPushButton("Predict")
        self.button.clicked.connect(self.predict_content)
        layout.addWidget(self.button)

        self.setLayout(layout)

    def predict_content(self):
        content = self.text_edit.toPlainText()
        content_sequence = tokenizer.texts_to_sequences([content])
        content_padded = pad_sequences(content_sequence, maxlen=max_sequence_length)
        prediction = model.predict(content_padded)

        if prediction[0] >= 0.5:
            result = "The content is produced by an AI chatbot."
        else:
            result = "The content is not produced by an AI chatbot."

        self.label.setText(result)

def main():
    app = QApplication(sys.argv)

    window = AIChatbotPredictor()
    window.show()

    sys.exit(app.exec_())

if __name__ == '__main__':
    main()

```

This is the GUI for the program



Thanks For Your Patience

We would like to entertain your questions now