CREATE A CHATBOT USING PYTHON

PHASE 3 : DEVELOPMENT PART 1

SUBMISSION DOCUMENT.....

Project : Create Chatbot Using Python.

Introduction:

- Creating a chatbot using Python involves developing a program that can simulate human conversation.
- You'll need to decide on the type of chatbot, such as rule-based or AI-powered, select the right tools and libraries, design the conversation flow, write the code, test it thoroughly, and then deploy it.
- Continuous maintenance and improvement are crucial for keeping your chatbot effective.
- Python's versatility and the availability of NLP and ML libraries make it a powerful choice for chatbot development, offering the potential to enhance user experiences, automate tasks, and provide valuable support in various applications.

Content for phase 3 in project:

In this sphase 3 , we begin building our project by loading and preprocessing the dataset.

Data Source :

A good data source for creating a chatbot should contain accurate ,complete ,and easy accessible one for users.

Dataset Link: https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot

hi, how are you doing? i'm fine. how about yourself?
i'm fine. how about yourself? i'm pretty good. thanks for asking.
i'm pretty good. thanks for asking. no problem. so how have you been?
no problem. so how have you been? i've been great. what about you?
i've been great. what about you? i've been good. i'm in school right now.
i've been good. i'm in school right now. what school do you go to?
what school do you go to? i go to pcc.
i go to pcc. do you like it there?
do you like it there? it's okay. it's a really big campus.
it's okay. it's a really big campus. good luck with school.
good luck with school. thank you very much.
how's it going? i'm doing well. how about you?
i'm doing well. how about you? never better, thanks.
never better, thanks. so how have you been lately?
so how have you been lately? i've actually been pretty good. you?
i've actually been pretty good. you? i'm actually in school right now.
i'm actually in school right now. which school do you attend?
which school do you attend? i'm attending pcc right now.
i'm attending pcc right now. are you enjoying it there?
are you enjoying it there? it's not bad. there are a lot of people there.
it's not bad. there are a lot of people there. good luck with that.
good luck with that. thanks.
how are you doing today? i'm doing great. what about you?
i'm doing great. what about you? i'm absolutely lovely, thank you.
i'm absolutely lovely, thank you. everything's been good with you?
everything's been good with you? i haven't been better. how about yourself?
i haven't been better. how about yourself? i started school recently.
i started school recently. where are you going to school?
where are you going to school? i'm going to pe.....

.....

Preprocessing your dataset is a crucial step when creating a chatbot using Python. Proper data preprocessing can improve the accuracy and effectiveness of your chatbot. Here are some essential steps for preprocessing your dataset:

- Data Collection:

Gather your chatbot training data. This could be in the form of conversation logs, customer support interactions, or any other relevant text data.

- Data Cleaning:

Remove any irrelevant or unnecessary information from your dataset. Handle or remove special characters, HTML tags, and any other noise in the text. Correct spelling and grammatical errors if necessary.Standardize text, such as converting all characters to lowercase.

- Tokenization:

Tokenization involves splitting the text into individual words or tokens. Python's NLTK or spaCy libraries are helpful for this task.

- Stop Word Removal:

Remove common stop words (e.g., "the," "is," "and") to reduce the dimensionality of the dataset and improve model performance.

- Stemming or Lemmatization:

Reducing words to their base or root form can improve model accuracy. You can use NLTK or spaCy for this purpose.

- Handling Contract :

Expand contractions (e.g., "can't" to "cannot") to ensure consistent language usage.

- Handling Synonyms:

Replace synonyms with a consistent term. This reduces the variation in the data.

- Remove Duplicates:

Eliminate duplicate conversations or messages from the dataset, as they can skew the model.

- Data Formatting:

Format the data into a structure that your chatbot model can understand. Typically, it's a list of input-output pairs or a context-response structure.

- Data Splitting:

Split the data into training, validation, and test sets. This allows you to train, tune, and evaluate your chatbot model effectively.

- Encoding Text:

Convert the text data into a numerical format that machine learning models can work with. Common methods include one-hot encoding, word embeddings (e.g., Word2Vec, FastText, or GloVe), or using deep learning models like transformers (e.g., BERT).

- Handling Imbalanced Data:

If you have a class imbalance (e.g., some intents or responses are more frequent than others), you may need to oversample or undersample to balance the data.

Install necessary libraries:

You will need the Transformers library from Hugging Face and the Torch library for PyTorch.

```
import nltk
from nltk.tokenize import word_tokenize

nltk.download('punkt')

# Provided conversation dataset
conversation = [
    ("hi, how are you doing?", "i'm fine. how about yourself?"),
    ("i'm fine. how about yourself?", "i'm pretty good. thanks for asking."),
    ("i'm pretty good. thanks for asking.", "no problem. so how have you been?"),
    ("no problem. so how have you been?", "i've been great. what about you?"),
    ("i've been great. what about you?", "i've been good. i'm in school right now."),
    ("i've been good. i'm in school right now.", "what school do you go to?"),
    ("what school do you go to?", "i go to pcc."),
    ("i go to pcc.", "do you like it there?)…….
https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot
]

# Create input-output pairs
input_data = [line[0] for line in conversation]
output_data = [line[1] for line in conversation]

# Tokenization
def tokenize_text(text):
    return word_tokenize(text)

input_data = [tokenize_text(line) for line in input_data]
output_data = [tokenize_text(line) for line in output_data]
```

```
# Remove stopwords
stop_words = set(stopwords.words('english'))
filtered_tokens = [word for word in tokens if word.lower() not in stop_words]

# Stemming
stemmer = PorterStemmer()
stemmed_tokens = [stemmer.stem(word) for word in filtered_tokens]

# Print the preprocessed input and output pairs
for input_line, output_line in zip(input_data, output_data):
    print("Input:", input_line)
    print("Output:", output_line)
    print()
```

- In summary, preprocessing the dataset when creating a chatbot is a crucial step to enhance the chatbot's performance and ensure a smooth conversational experience.
- This process involves cleaning the data by removing noise and irrelevant information, standardizing text, and handling issues like contractions and special characters.
- It also includes reducing the dimensionality of the data through techniques like removing stopwords and stemming, making it more manageable for machine learning models.
- Data formatting into input-output pairs or context-response structures is essential for training the chatbot.
- Overall, a well-preprocessed dataset is the foundation for a chatbot that can provide accurate, consistent, and user-friendly responses while also facilitating easier maintenance and updates as new data becomes available.

CONCLUSION AND FUTURE WORK(Phase 3) :

Project Conclusion :

- In the Phase 3 conclusion, In the early stages of our project, we loaded and preprocessed the dataset, setting a strong foundation for our chatbot's development. This step involved cleaning and structuring the data, enabling our chatbot to deliver accurate and consistent responses
- Future Work: We will discuss potential avenues for future work, building the project by performing different activities like feature engineering, model training, evaluation etc as per the instructions in the project.