

JESTMASTER : PROJECT 2

REPORT

Name: Sahrudayi Caroline P.
NETID: SXP230055

System Description:

JestMaster is a personalized joke assistant that specializes in delivering hilarious one-liners and joke punchlines tailored to the user's preferences. The system utilizes various Machine Learning (ML)/ Deep Learning(DL) and Natural Language Processing (NLP) techniques to understand user input, generate humorous responses, and maintain user profiles.

Key Components:

User Input Processing:

- JestMaster preprocesses user input using NLP techniques to clean and tokenize the text. This ensures that the input is in a suitable format for further processing.
- JestMaster also used **Named Entity Recognition(NER)** in spacy to determine what and where a proper noun might be, for the purpose of parsing a name from a sentence

Joke Punchline Generation:

- Jokes are generated using a **sequence-to-sequence LSTM (Long Short-Term Memory)** neural network architecture. The model consists of an encoder and a decoder.
- The encoder processes the user's input sequence and generates hidden states that represent the context of the input.
- The decoder takes these hidden states and generates a sequence of words representing the punchline or joke response.
- The model is trained on a corpus of jokes and punchlines along with a corpus on casual conversations to learn the relationship between input prompts and humorous outputs besides being able to carry a casual conversation.

User Profile Management:

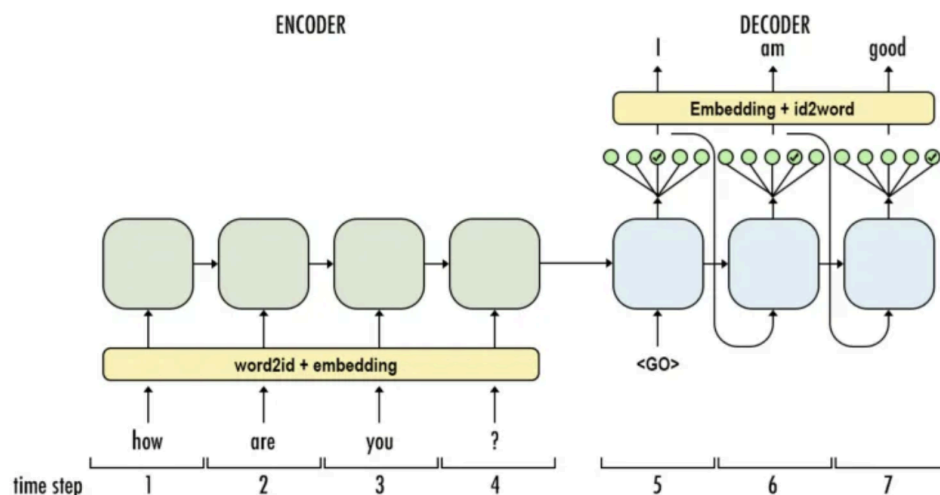
- JestMaster maintains user profiles to personalize the joke delivery experience.
- When a new user interacts with JestMaster, a user profile is created, storing information such as the user's name, preferred joke types, and dislikes.
- User profiles are stored in text files in a specified directory, allowing JestMaster to access and update user information as needed.

Conversational History Logging:

- JestMaster logs conversations with users, storing both user prompts and JestMaster's responses.
- This logging enables JestMaster to maintain a history of interactions with each user, facilitating personalized interactions and improving joke recommendations over time.

Techniques Used:

- **Sequence-to-Sequence LSTM Model:** The core of JestMaster's joke generation capability is based on a sequence-to-sequence LSTM model. This architecture allows JestMaster to learn the patterns and structures of jokes, casual conversations and generate coherent and humorous responses.
- **Seq2Seq Model**
- The Sequence to Sequence model (seq2seq) consists of two RNNs — an encoder and a decoder. The encoder reads the input sequence, word by word and emits a context (a function of final hidden state of encoder), which would ideally capture the essence (semantic summary) of the input sequence. Based on this context, the decoder generates the output sequence, one word at a time while looking at the context and the previous word during each timestep. This is a ridiculous oversimplification, but it gives you an idea of what happens in seq2seq. (“Chat Bot using Seq2Seq Model!. Many undergraduate students are facing... | by Vvs Manideep | Medium”)



- **NLP Techniques:** User input is preprocessed using NLP techniques such as tokenization and cleaning to prepare it for input into the LSTM model. Regex is used to further clean the data. NER techniques have been employed by using Spacy.

Datasets and Cleaning:

I used the jokes Question-Answer along with a conversations dataset from Kaggle. Links mentioned below

<https://www.kaggle.com/datasets/jiروز/qa-jokes/data>

<https://www.kaggle.com/datasets/kreeshrajani/3k-conversations-dataset-for-chatbot>

About qa jokes dataset:

This dataset contains 38,269 jokes of the question-answer form, obtained from the r/Jokes subreddit. The dataset contains a csv file, where a row contains a question ("Why did the chicken cross the road"), the corresponding answer ("To get to the other side") and a unique ID.

The data comes from the end of 2016 all the way to 2008. The entries with a higher ID correspond to the ones submitted earlier.

An example of what one might do with the data is build a sequence-to-sequence model where the input is a question and the output is an answer. Then, given a question, the model should generate a funny answer. This is what I did as the final project for my fall 2016 machine learning class. The project page can be viewed [here](#).

Disclaimer: The dataset contains jokes that some may find inappropriate. (Kaggle)

About Conversations dataset:

About Dataset

This dataset is used for research or training of natural language processing (NLP) models. The dataset may include various types of conversations such as casual or formal discussions, interviews, customer service interactions, or social media conversations.

Application

- Chatbots and virtual assistants: Conversation datasets are used to train chatbots and virtual assistants to interact with users in a more human-like manner.
- Customer service: Conversation datasets can be used to train customer service chatbots, allowing companies to provide 24/7 customer support without human intervention(Kaggle)

Preparing and cleaning:

- The qa jokes dataset had some offensive content as it's source is a subreddit called r/jokes. I had censored offensive words.
- I referred to other datasets and manually added about 200 rows to the conversations dataset to make the dataset more varied and robust.
- **Duplicate Removal:**
Duplicate questions and their corresponding answers are removed from the dataset. This step ensures that each question is unique and avoids redundancy in the training data.
- **Text Cleaning:**
The `clean_sent` function is applied to each question and answer to preprocess the text. This function:
 - Converts the text to lowercase to ensure uniformity.
 - Expands contractions to replace common contractions with their full forms.
For example, "won't" becomes "will not".

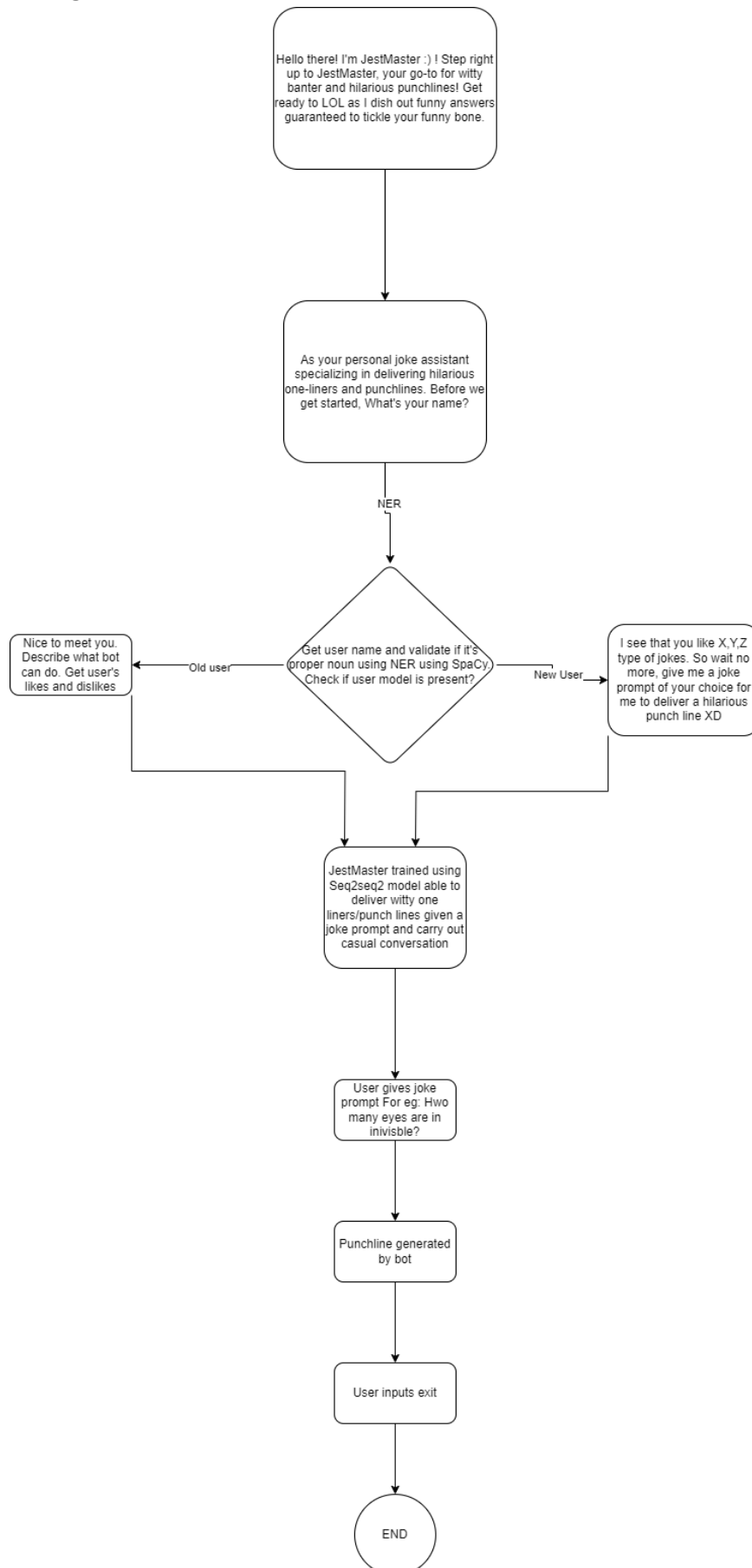
- Removes punctuation marks using regular expressions, leaving only alphanumeric characters and spaces.

Adding Special Tokens:

- Each answer is preprocessed by adding special tokens '<SOS>' (Start of Sentence) at the beginning and '<EOS>' (End of Sentence) at the end. These tokens help the model learn when to start and stop generating responses.
- Special tokens '<PAD>' (Padding) and '<OUT>' (Out of Vocabulary) are added to the vocabulary dictionary to handle padding and out-of-vocabulary words, respectively.
- Padding Sequences: The encoded input sequences (questions) and decoded input sequences (answers) are padded to ensure uniform length. The `pad_sequences` function pads each sequence with zeros to a maximum length of 30 tokens.
- Padding is applied at the end of sequences ('post') to maintain the integrity of the input data. For the decoded input sequences (final output), the first token (representing '<SOS>') is removed before padding to exclude it from the training data.

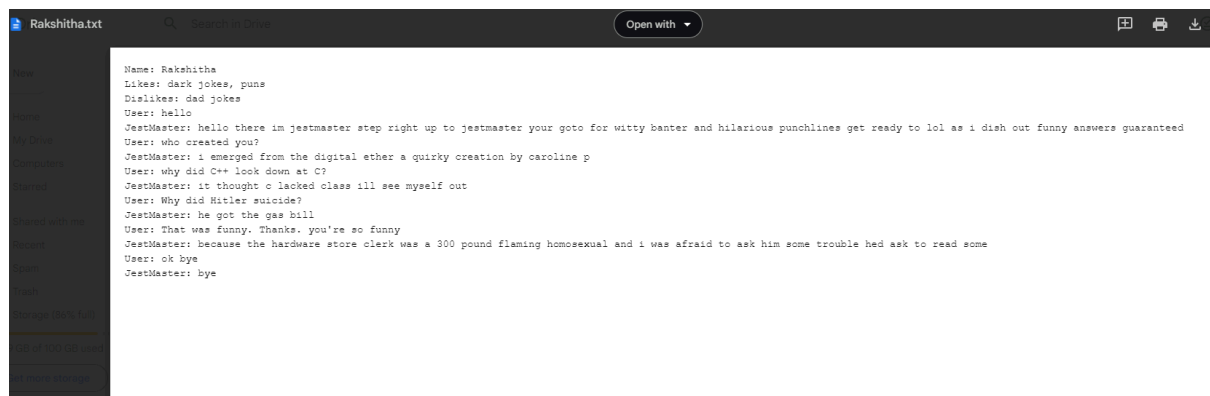
These preprocessing steps are essential for preparing the dataset in a format suitable for training a sequence-to-sequence model. They help standardize the data, handle vocabulary management, and ensure consistency in input and output sequences.

Dialog Tree:



User: hello
JestMaster: hello there im jestmaster step right up to jestmaster your goto for witty banter and hilarious punchlines get ready to lol as i dish out funny answers guaranteed
User: how's it going?
JestMaster: im doing well how about you
User: I'm good thanks
JestMaster: I see
User: what can you do?
JestMaster: ah the ageold question im here to sprinkle a dash of humor into your day with punchy punchlines and witty banter so ask away and brace yourself for
User: what's hitler's favorite board game?
JestMaster:Gas Who?
User: what do you call an iron cat?
JestMaster: a feline
User: Bye
JestMaster: see you again! Bye!

Appendix:



```
{
  "Name": "Rakshitha",
  "Likes": "dark jokes, puns",
  "Dislikes": "dad jokes",
  "User": "hello",
  "JestMaster": "hello there im jestmaster step right up to jestmaster your goto for witty banter and hilarious punchlines get ready to lol as i dish out funny answers guaranteed",
  "User": "who created you?",
  "JestMaster": "i emerged from the digital ether a quirky creation by caroline p",
  "User": "why did C++ look down at C?",
  "JestMaster": "it thought c lacked class ill see myself out",
  "User": "Why did Hitler suicide?",
  "JestMaster": "he got the gas bill",
  "User": "That was funny. Thanks. you're so funny",
  "JestMaster": "because the hardware store clerk was a 300 pound flaming homosexual and i was afraid to ask him some trouble hed ask to read some",
  "User": "ok bye",
  "JestMaster": "bye"
}
```

User Model - Rakshitha.txt

Evaluations and Analysis:

Strengths:

- JestMaster engages users in personalized conversations by addressing them by name, asking about their preferences, and tailoring joke delivery based on their likes and dislikes. This personalized interaction enhances user engagement and satisfaction.
- It effectively generates humorous responses and punchlines using a sequence-to-sequence LSTM model. It understands the context of user prompts and delivers relevant and witty jokes, contributing to a fun user experience.
- JestMaster maintains a log of conversations with users, enabling it to remember past interactions and maintain continuity in conversations.

Weaknesses:

- While JestMaster delivers humorous responses, its repertoire of jokes may be limited. Since it relies on a pre-existing dataset of jokes, it may struggle to provide novel or diverse responses, leading to repetitive interactions with users over time.
- I noticed that the chatbot generates only somewhat relevant results sometimes. The dataset is ~40k rows and I made the assumption to train the model to overfit over the dataset I considered. The results can be improved using a larger dataset and training a more robust model which would need additional processing power.
- The chatbot does not incorporate mechanisms for continuous learning or adaptation based on user feedback. As a result, it may not improve its performance or adapt to changing user preferences over time, potentially leading to stagnation in user engagement.

Survey:

Survey on Chatbot Experience

Thank you for taking the time to provide feedback on your experience with my chatbot, JestMaster! Please rate the following statements based on your interaction with the chatbot:

| | Keerthi | Amulya | Rakshitha |
|------------------------------|--|----------------------------------|---|
| Overall Satisfaction | 4 | 5 | 5 |
| Ability to hold Conversation | 5 | 4 | 5 |
| Ease of use | 5 | 5 | 5 |
| Any additional Comments | I liked the witty personality of the bot | Can handle user feed back better | I loved the creativity and customisation of the bot |