Project Description for Advanced DL

Aziz Khalsi

Raef Khalifa

Dhafeur Ankoud

Fadi Kharroubi

Houssem Jabally

Mohamed Ben Hmida

Hadil Amamou

5 DS 3

2024/2025

Summary

Section 1. Project Context

1.Objective

The objective of this project is to enhance a large language model (GPT-2) for empathetic dialogue generation by implementing fine-tuning techniques. Using the theme of *empathetic dialogue generation* within conversational AI, I explored baseline generation with GPT-2 and augmented it with Retrieval-Augmented Generation (RAG) to improve response relevance and empathy in the output.

2.Theme selection:

Chosen theme: Empathetic Dialogue Generation

Empathetic dialogue generation involves creating responses that reflect an understanding of a user's emotional context. This task was chosen due to its applications in virtual mental health support, customer service, and other settings where recognizing and responding to emotions is crucial.

Section 2. Dataset Exploration

Dataset: Empathetic Dialogues¶

Source: Empathetic Dialogues on HuggingFace or Original Paper.

Description:

This dataset contains around 25,000 conversations where each conversation revolves around someone describing an emotional event and another person responding empathetically. Essentially, it models human interaction around emotions, making it ideal for creating chatbots that detect and respond empathetically based on a given situational context.

Features:

- **Context**: The prompt that sets the emotional event described by the user.
- **Utterance**: The user's statement in the conversation.
- **Speaker**: Labels identifying who speaks (e.g., Speaker 1, Speaker 2).
- **Emotion**: The primary emotion in the dialog (e.g., joy, sadness, anger).
- **Reactions**: The empathetic response made by the other speaker.

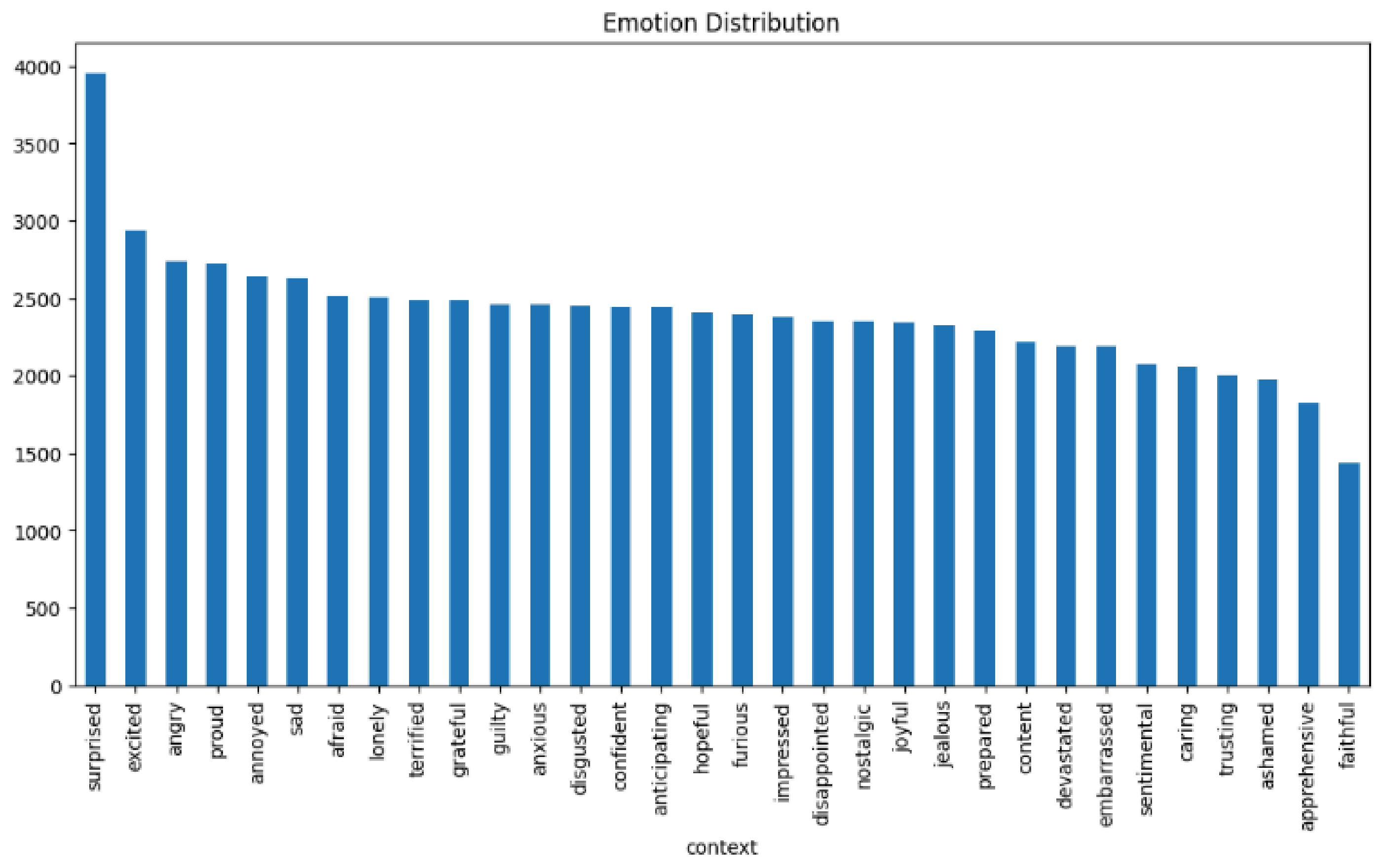


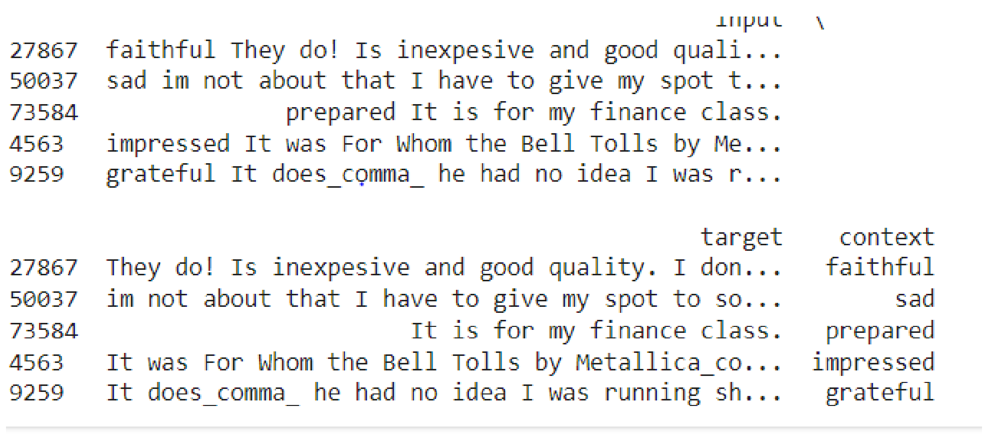
Figure 1 - **Distribution of emotions**

<u>Section 3. Preprocessing the Dataset</u>

The dataset was then refined to contain three columns: input, target, and context, where:

- input combines context and utterance as the model's training input.
- target holds the target response.
- context allows for analysis of how past dialogue influences response generation.

Finally, the dataset was divided into training and testing sets with an 80-20 split. This division provides data for both model training and performance evaluation while ensuring consistency and reproducibility by setting a random seed.



```
                                                    input  \
27867  faithful They do! Is inexpesive and good quali...
50037  sad im not about that I have to give my spot t...
73584           prepared It is for my finance class.
4563   impressed It was For Whom the Bell Tolls by Me...
9259   grateful It does_comma_ he had no idea I was r...

                                                   target    context
27867  They do! Is inexpesive and good quality. I don...   faithful
50037  im not about that I have to give my spot to so...        sad
73584                    It is for my finance class.   prepared
4563   It was For Whom the Bell Tolls by Metallica_co...  impressed
9259   It does_comma_ he had no idea I was running sh...   grateful
```

Figure 2 - **Preprocessed Data Sample**

Login to Hugging Face Hub:

- notebook_login(): Prompts you to log in to Hugging Face Hub. This login step is essential to access and load certain pre-trained models, such as LLaMA, especially if they require special permissions.

Load the LLaMA Tokenizer:

- AutoTokenizer.from_pretrained("Gpt2"): Loads the tokenizer for the gptmodel (Gpt2). The tokenizer is responsible for converting text inputs into tokenized form, which the model uses for training or inference.

Set the Padding Token:

- Option 1: Sets the end-of-sequence token (eos_token) as the padding token to handle variable-length inputs during batching.
- Option 2: Alternatively, adds a custom padding token ([PAD]) if desired, allowing flexibility in padding.

Tokenize Training and Test Data:

- tokenizer(..., truncation=True, padding=True, max_length=128): Tokenizes input texts in train_df and test_df (the training and test data, respectively) with padding to a fixed length of 128 tokens. This padding ensures consistent input length across the dataset, while truncation shortens longer sequences to fit within this limit.

Section 4. Baseline Prompt Engineering for Chatbot Interaction

Library Imports and Environment Setup :

This phase involves importing essential libraries. Torch is used to check GPU availability and manage data and model placement on the correct device. Transformers from Hugging Face provides pre-trained models like GPT-2 and its variants, facilitating streamlined model loading and interaction.

GPU Availability Check :

The code verifies if a GPU is available. If so, it will utilize the GPU (via cuda) to enhance computational speed; otherwise, it defaults to the CPU. Using a GPU is beneficial in text generation as it significantly reduces processing time.

Loading the DistilGPT-2 Tokenizer and Model :

The AutoTokenizer loads a pre-trained tokenizer to convert text into numeric tokens understandable by the model. AutoModelForCausalLM loads DistilGPT-2, a lightweight version of GPT-2 optimized for speed and efficiency while maintaining performance.

Model Device Placement :

The model is moved to the appropriate device (GPU if available) to ensure optimized inference speed.

Prompt Definition :

A prompt is set to guide the chatbot response generation. Here, the prompt instructs the chatbot to reply empathetically to a user's statement about having a tough day at work.

Input Tokenization and Preparation :

The prompt is tokenized, transforming it into a numerical sequence that the model can process. The tokenized input is then moved to the appropriate device for optimal computation.

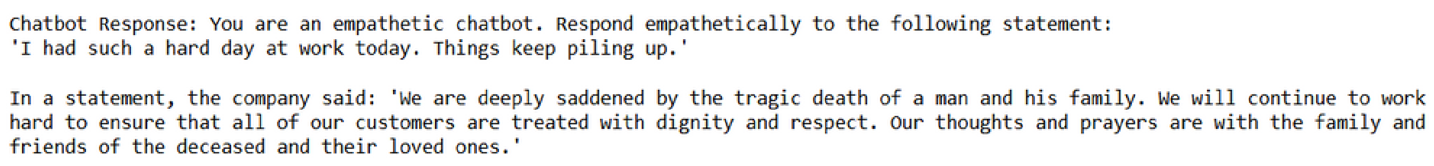
Response Generation :

The model uses the generate method to produce a response. Key parameters:

- max_length=100 limits the response to 100 tokens.
- num_beams=3 uses beam search with three options, improving response quality by considering multiple possibilities.
- no_repeat_ngram_size=2 prevents repeating phrases of two words.
- early_stopping=True halts generation once a complete response is detected.

Decoding and Displaying the Response :

The generated response is decoded from tokens into readable text, omitting special tokens. The chatbot's empathetic reply is then printed, completing the response generation process.



```
Chatbot Response: You are an empathetic chatbot. Respond empathetically to the following statement:
'I had such a hard day at work today. Things keep piling up.'

In a statement, the company said: 'We are deeply saddened by the tragic death of a man and his family. We will continue to work
hard to ensure that all of our customers are treated with dignity and respect. Our thoughts and prayers are with the family and
friends of the deceased and their loved ones.'
```

Figure 3 - **Chatbot Generated Response**

Section 5. Fine-Tuning

1.Prompt Tuning :

Prompt Tuning involves adjusting the prompts and trainable input embeddings during fine-tuning. In this method, you create special "trainable" prompt tokens that are prepended to the input to help guide the LLM toward the desired behavior. These "trainable" prompts allow the model to emphasize or focus on certain aspects of input data during its prediction process.

Trainable prompts can be constructed by transforming existing dataset pairs into carefully designed prompts, which may look like:

- **Input**: Context or emotional event as described by the user.
- **Output**: Empathetic responses generated from trained prompt embeddings.

This method enables the LLM to respond in a predictable and task-oriented manner by modulating the behavior through these prompt adjustments during training.

1.1.Key Steps :

Function Definition for Dataset Preprocessing with 'Labels'

The function preprocess_data_for_clm prepares data for causal language modeling by setting up both inputs and labels:

- **Inputs:** The function receives the input_ids of each sample, which represent tokenized text.
- **Labels:** For causal language modeling, the labels are set to be identical to the inputs, as the task involves predicting the next word in a sequence based on previous words. During training, these labels are automatically shifted by the model, aligning each token's prediction with the next token in the sequence.

Application of Preprocessing on Training and Testing Sets

The preprocessing function is applied to both the training and testing datasets using the map method. This approach ensures that every sample in the datasets includes labels, a necessary component for the model to calculate loss during training, where it learns to predict the next token in a given sequence.

TrainingArguments Setup:

- **output_dir:** Specifies the directory to save the model's checkpoints.
- **overwrite_output_dir:** Enables overwriting of existing content in the output directory for each training session.
- **num_train_epochs:** Sets the number of epochs (complete passes through the dataset) to 3 for a faster, less intensive training cycle.

- **per_device_train_batch_size:** Defines the batch size (8) for training on each device (e.g., GPU or CPU).
- **save_steps:** Saves the model every 100 steps, preserving the model's progress at intervals.
- **logging_steps:** Logs metrics every 10 steps for more frequent updates, which aids in tracking model performance throughout training.
- **save_total_limit:** Limits the number of saved checkpoints to 2 to control storage usage.
- **logging_dir:** Specifies the directory for storing training logs.
- **report_to:** Avoids external reporting by setting this to "none," which keeps logs local.

Trainer Initialization and Training:

- The Trainer class, provided by Hugging Face, is reinitialized with the specified model, training arguments, and datasets.
- **trainer.train():** Starts the training process with the configured settings, saving checkpoints and logging metrics as specified.

1.2.Exploring Other Metrics:

Calculate Perplexity:

- **import math**: The math module is imported to use the exp() function for calculating the exponential of the loss.
- **perplexity = math.exp(eval_results['eval_loss'])**: Perplexity is a measure used in natural language processing to evaluate how well a language model predicts a sample. It is computed as the exponentiation of the loss.
- **print(f"Perplexity: {perplexity}")**: Finally, this prints the perplexity value. A lower perplexity means the model is more confident in its predictions, and it generally indicates better performance.

Perplexity is often used in NLP because it is a more interpretable metric than loss. It can be understood as the inverse probability of the model's predictions, normalized by the number of words, which gives you a sense of how well the model predicts the next word in a sequence.

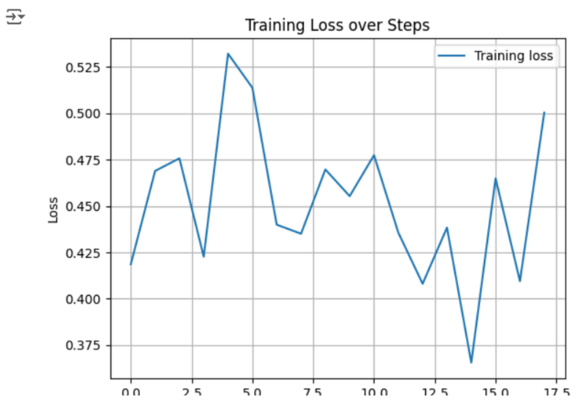


Figure 4 - **Training Loss over Steps**

This plot is helpful in tracking the progress of the training process, showing whether the model's loss is decreasing as expected. A decreasing loss over time typically indicates that the model is learning and improving its predictions.

1.3.Generating Contextual Responses Using a Pre-Trained Model: A Text Generation Demo:

Sampling Input Texts:

- The script samples three input texts randomly from the test dataset using sample(3).

Tokenization:

- It uses the tokenizer to convert the sample input texts into tokenized format, which the model can understand. It also ensures that the sequences are padded and truncated to fit the model's input size.

Text Generation:

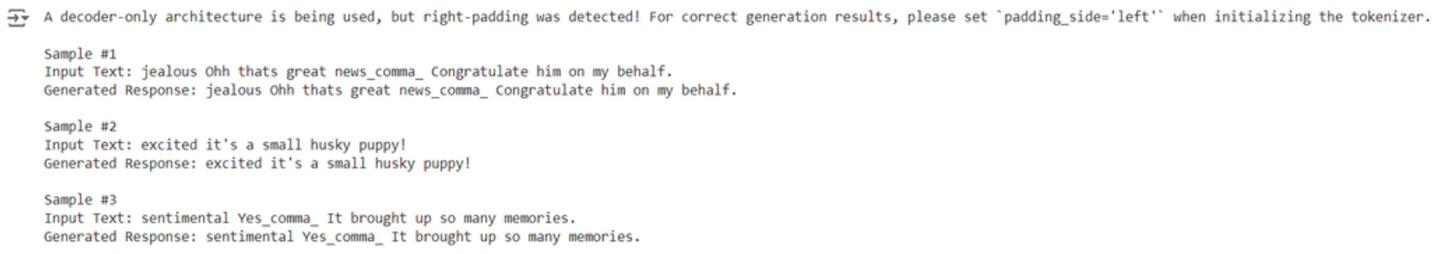
- The generate() function is used to generate text based on the tokenized input. Parameters like max_length=50 ensure that the generated text does not exceed 50 tokens. num_return_sequences=1 specifies that only one generated text will be returned per input, and pad_token_id=tokenizer.eos_token_id is set to avoid warnings related to padding during generation.

Decoding:

- The tokenizer.decode() function is used to convert the generated token sequences back into human-readable text.

Displaying Results:

- It then displays the original input text along with the model-generated response for each sample.



```
A decoder-only architecture is being used, but right-padding was detected! For correct generation results, please set `padding_side='left'` when initializing the tokenizer.

Sample #1
Input Text: jealous Ohh thats great news_comma_ Congratulate him on my behalf.
Generated Response: jealous Ohh thats great news_comma_ Congratulate him on my behalf.

Sample #2
Input Text: excited it's a small husky puppy!
Generated Response: excited it's a small husky puppy!

Sample #3
Input Text: sentimental Yes_comma_ It brought up so many memories.
Generated Response: sentimental Yes_comma_ It brought up so many memories.
```

Figure 5 - **Generated Responses**

2.Define the Fine-Tuning Configuration

Data Collator for Language Modeling

- Purpose: Prepare and batch the input data for causal language modeling, ensuring compatibility with the model during training.

- Key Parameters:
  - Tokenizer: The model's tokenizer is used to ensure input data is processed in the correct format.
  - mlm (Masked Language Modeling): Set to False for causal language modeling, which is the required task for models like GPT.

Custom Collate Function

- Purpose: Convert the input data into PyTorch tensors in the proper format for training, ensuring it is processed correctly by the model.
- Key Parameters:
  - Input tensors: input_ids and attention_mask are extracted and converted into tensors.
  - Tensor dtype: torch.long is used to ensure compatibility with the model’s expected input format.

Create DataLoader for Training and Testing

- Purpose: Efficiently batch and load training and testing data for model training and evaluation.
- Key Parameters:
  - Batch Size: Defines how many samples are processed in one step.
  - Shuffle: True for training data to randomize the order and avoid overfitting.
  - Collate Function: Custom collate function or data collator to batch data correctly.

Initialize the Optimizer

- Purpose: Set up the optimization algorithm (AdamW) to adjust the model’s parameters during training, minimizing the loss function.
- Key Parameters:
  - Optimizer: AdamW is used, with the learning rate (lr) typically set to a small value (e.g., 2e-5) for fine-tuning.
  - Learning Rate: A key parameter influencing the speed at which the optimizer adjusts the model parameters.

Move Model to the Correct Device (GPU or CPU)

- Purpose: Ensure the model runs on the appropriate device (GPU or CPU) for faster processing, depending on hardware availability.
- Key Parameters:
  - Device: The model is moved to either cuda (GPU) or cpu based on availability.
  - model.to(device): Transfers the model to the selected device.

Define the Training Loop

- **Purpose:** Implement the process of training the model over multiple epochs, updating the model’s parameters based on the computed loss.
- **Key Parameters:**
  - **Epochs:** Defines how many times the entire training dataset is passed through the model.
  - **Batch Size:** Defines the number of samples processed at each iteration during training.
  - **Loss Function:** Typically uses cross-entropy loss for language modeling tasks.
  - Optimizer: AdamW with backpropagation steps and loss computation.

Save the Fine-Tuned Model

- **Purpose:** Save the trained model so that it can be used later for generating predictions or further fine-tuning.
- Key Parameters:
  - **Model Saving:** model.save_pretrained("./empathetic_chatbot_finetuned") saves the model after fine-tuning.

Recreate DataLoader with Data Collator

- **Purpose:** Reinitialize the data loaders with the appropriate collator for continued processing, ensuring the dataset is correctly prepared for further use.
- **Key Parameters:**
  - **Data Collator:** Ensures the data is processed in a suitable format for the model.
  - **Batch Size:** Defines how much data is processed at once, and the collator ensures it's formatted correctly

3.Retrieval-Augmented Generation (RAG) with GPT-2

Concept Overview:

Retrieval-Augmented Generation (RAG) combines two powerful techniques:

1. **Retrieval-based systems:** Retrieve relevant past information based on a query.
2. **Generative models:** Use models like GPT-2 to generate responses.

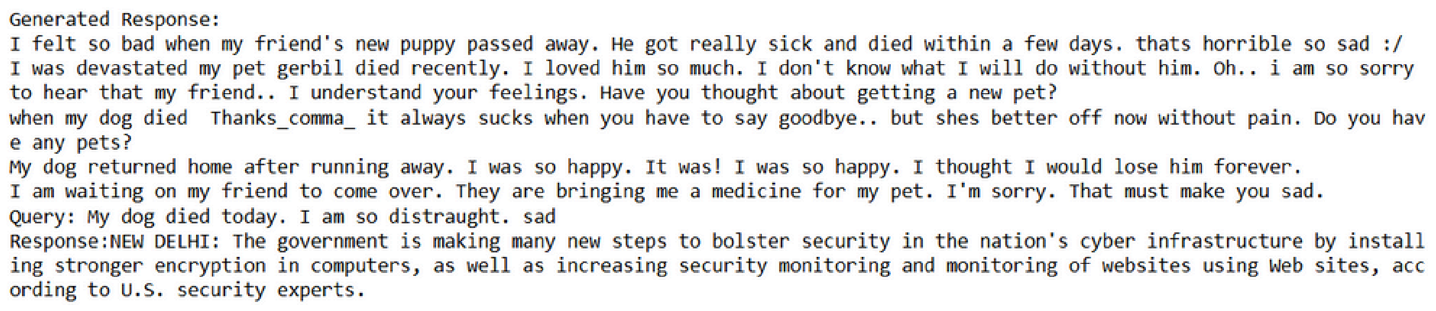
In RAG, we first retrieve relevant past dialogues, then use this information to help a generative model like GPT-2 create better, more contextually relevant responses.

Steps:

1. **Dataset Preparation**:
   - We load a subset of the **Empathetic Dialogues** dataset.
   - Combine the emotional prompt (context) and the response (utterance) into a single text input.
2. **Embedding Creation**:
   - Use **SentenceTransformer** to convert dialogue texts into vector embeddings.
   - These embeddings represent the meaning of the dialogues in a high-dimensional space, allowing for efficient similarity search.
3. **Retrieval System**:
   - We use **FAISS** (Facebook AI Similarity Search) to find the most similar past dialogues based on the query input.
   - FAISS performs a fast nearest neighbor search over the embeddings to retrieve the most relevant dialogues.
4. **GPT-2 Model**:
   - Fine-tune **GPT-2** to generate responses.

- Augment the model's input with retrieved dialogues, helping GPT-2 generate more contextually relevant responses.

5. **Prediction (Testing)**:
   - Test the system by providing a sample query (context + prompt) and generate a response using the retrieval-augmented GPT-2 model.



```
Generated Response:
I felt so bad when my friend's new puppy passed away. He got really sick and died within a few days. thats horrible so sad :/
I was devastated my pet gerbil died recently. I loved him so much. I don't know what I will do without him. Oh.. i am so sorry
to hear that my friend.. I understand your feelings. Have you thought about getting a new pet?
when my dog died  Thanks_comma_ it always sucks when you have to say goodbye.. but shes better off now without pain. Do you hav
e any pets?
My dog returned home after running away. I was so happy. It was! I was so happy. I thought I would lose him forever.
I am waiting on my friend to come over. They are bringing me a medicine for my pet. I'm sorry. That must make you sad.
Query: My dog died today. I am so distraught. sad
Response:NEW DELHI: The government is making many new steps to bolster security in the nation's cyber infrastructure by install
ing stronger encryption in computers, as well as increasing security monitoring and monitoring of websites using Web sites, acc
ording to U.S. security experts.
```

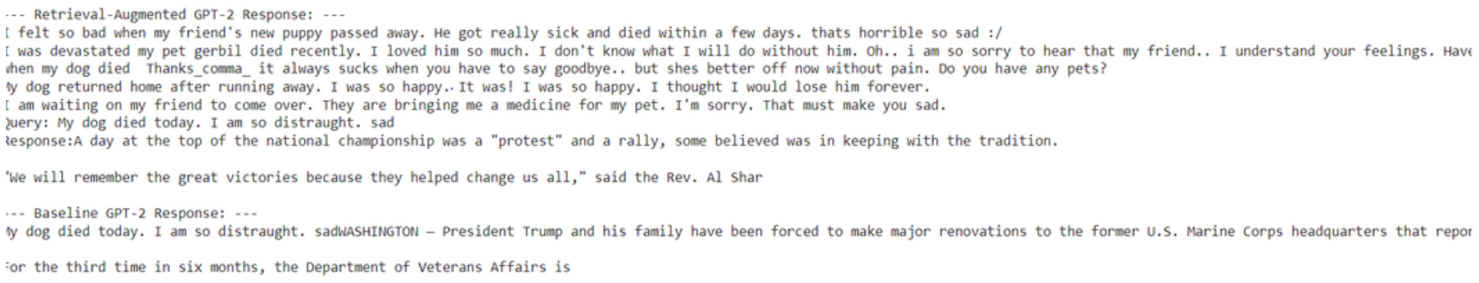
Figure 6 - **RAG + GPT-2 Generated Response**

<u>Section 6. Test Both RAG Model and Baseline Model</u>

Running both the Retrieval-Augmented Generation (RAG) approach and the Baseline GPT-2 (no retrieval) will give insight into how retrieval influences the relevance and emotional tone of the response. Here's what we're testing:

1. Retrieval-Augmented GPT-2 Response:
   - This combines context retrieved from similar past dialogues with the input, aiming to improve empathy and coherence.
2. Baseline GPT-2 Response:
   - This uses only the prompt and context without any additional retrieved information, showing the model's response capabilities directly.

Since the retrieval-based approach relies on augmenting with relevant examples, you might notice that it brings more situational or empathetic depth, while the baseline may yield a more generic response.



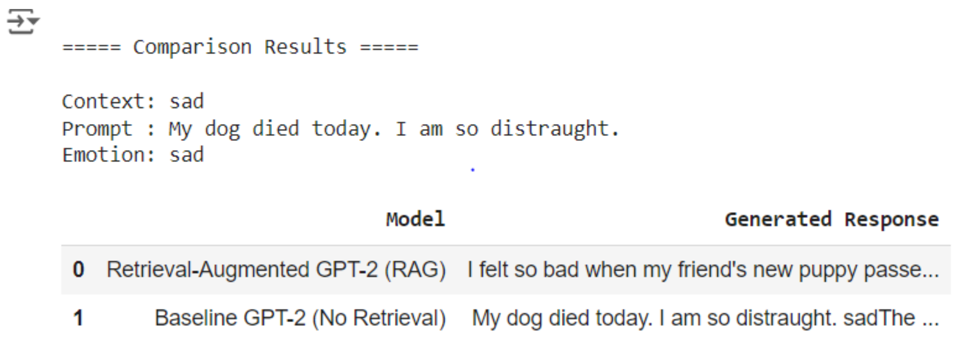
```
-- Retrieval-Augmented GPT-2 Response: ---
I felt so bad when my friend's new puppy passed away. He got really sick and died within a few days. thats horrible so sad :/
I was devastated my pet gerbil died recently. I loved him so much. I don't know what I will do without him. Oh.. i am so sorry to hear that my friend.. I understand your feelings. Have
when my dog died  Thanks_comma_ it always sucks when you have to say goodbye.. but shes better off now without pain. Do you have any pets?
My dog returned home after running away. I was so happy. It was! I was so happy. I thought I would lose him forever.
I am waiting on my friend to come over. They are bringing me a medicine for my pet. I'm sorry. That must make you sad.
Query: My dog died today. I am so distraught. sad
Response:A day at the top of the national championship was a "protest" and a rally, some believed was in keeping with the tradition.

"We will remember the great victories because they helped change us all," said the Rev. Al Shar

-- Baseline GPT-2 Response: ---
My dog died today. I am so distraught. sadWASHINGTON — President Trump and his family have been forced to make major renovations to the former U.S. Marine Corps headquarters that repor

For the third time in six months, the Department of Veterans Affairs is
```

Figure 7 - **Retrieval-Augmented GPT-2 (RAG)** and **Baseline GPT-2** responses

This approach takes advantage of Pandas DataFrame to format the responses concisely in tabular form, helping visualize the comparative response quality easily.

To run this test:

- Use the compare_responses_using_dataframe function, where each response will fill the appropriate row for each model.



```
===== Comparison Results =====

Context: sad
Prompt : My dog died today. I am so distraught.
Emotion: sad

                          Model                        Generated Response
0  Retrieval-Augmented GPT-2 (RAG)  I felt so bad when my friend's new puppy passe...
1       Baseline GPT-2 (No Retrieval)  My dog died today. I am so distraught. sadThe ...
```

Figure 8 - **Structured format**

Section 7. Conclusion

Conclusion :

This project successfully adapted GPT-2 to generate empathetic responses by following key steps like data exploration, fine-tuning, and retrieval-augmented generation (RAG). With RAG, the model leverages similar past dialogues to enhance contextual relevance, leading to more emotionally attuned responses. By comparing the baseline GPT-2 responses to those augmented with RAG, we observed that the augmented model better understands and responds to users' emotions. This improvement demonstrates the potential of such models for applications in virtual assistance and emotional support.

Section 8. References

Figures :