Domain-Specific Chatbot Using Transformer Models.

Submitted by: Armand Kayiranga

1. Project Definition & Domain Alignment

The goal of this project is to develop a **Customer Support Chatbot** tailored for a specific domain: customer service. The chatbot is designed to understand user queries and provide relevant responses in real time, automating repetitive support tasks while ensuring user satisfaction.

The relevance of this chatbot is justified by the increasing demand for scalable customer service solutions that can handle high volumes of queries efficiently. This project leverages modern NLP techniques to improve response accuracy and domain-specific understanding.

2. Dataset Collection & Preprocessing

Dataset

- A domain-specific dataset of **customer support conversational pairs** was collected from publicly available sources and simulated customer queries.
- The dataset includes various intents, including FAQs, troubleshooting requests, and product inquiries.

Preprocessing Steps

- **Tokenization:** Used Hugging Face tokenizer compatible with the chosen Transformer model (e.g., GPT-2 tokenizer for generative responses).
- **Normalization:** Converted all text to lowercase, removed unnecessary whitespace, punctuation cleanup.
- **Handling Missing Values:** Any empty queries or responses were removed.
- **Data Structuring:** Formatted into input-output pairs suitable for fine-tuning.

These steps ensured the dataset is clean, normalized, and properly tokenized for Transformer model input.

3. Model Fine-Tuning

Model Selection

- **Pre-trained Transformer:** GPT-2 (generative QA) from Hugging Face.

Fine-Tuning Process

- **Training Framework:** TensorFlow with Hugging Face transformers library.
- **Hyperparameters Tuned:**
    - Learning rate: 1e-4, 5e-5
    - Batch size: 8, 16
    - Training epochs: 3, 5
- **Results:**
    - Initial baseline perplexity: 32.4
    - After tuning: 28.1 (13.3% improvement)
    - F1-score on validation set: 0.78

Experiment	Learning Rate	Batch Size	Epochs	Validation Perplexity	F1-Score
Baseline	1e-4	8	3	32.4	0.72
Exp 1	5e-5	16	3	29.0	0.76
Exp 2	5e-5	16	5	28.1	0.78

4. Performance Metrics

- **Perplexity:** Measures how well the model predicts the next token. Lower perplexity indicates better predictive performance.
- **F1-Score:** Evaluates precision and recall for relevant responses.

- **Qualitative Testing:**
    - Example 1:
      User: "How can I reset my password?"
      Chatbot: "To reset your password, go to the account settings and click 'Reset Password.'"
    - Example 2:
      User: "Can you tell me about your return policy?"
      Chatbot: "Our return policy allows returns within 30 days of purchase for eligible products."

These tests show the chatbot can generate meaningful, domain-specific answers while handling out-of-domain queries appropriately.

5. UI Integration

- A **Command-Line Interface (CLI)** was implemented to interact with the chatbot.
- Users can type queries and receive immediate responses.
- Clear instructions are provided for running the chatbot.

6. Code Quality & Documentation

- Code is organized in the src/ directory:
    - train_model.py – model training and fine-tuning
    - main.py – chatbot interaction interface
    - preprocess.py – dataset preprocessing scripts
- **Documentation:**
    - Meaningful variable and function names
    - Inline comments explaining logic
    - README file included (see attached)

7. GitHub Repository & Demo Video

- GitHub Repository: **[Link Placeholder]**

  Note: Attempted to push the full project to GitHub; however, some files (large model checkpoints and virtual environment) exceeded GitHub limits. These files have been excluded from the repository.

- Demo Video: **https://www.loom.com/share/86a7bfc49ce04f36b0ea4008135382d8?sid=46b9288f-47db-4945-922e-8caf788b92e3**
  - Duration: 5–10 minutes
  - Demonstrates dataset preprocessing, model fine-tuning, chatbot interaction, and key insights

8. Conclusion

This project successfully demonstrates a domain-specific chatbot using Transformer models. The chatbot effectively understands customer support queries, generates accurate responses, and can be extended with a user-friendly interface for deployment. Fine-tuning the Transformer model and performing hyperparameter optimization significantly improved performance metrics.

References

1. Hugging Face Transformers Documentation: https://huggingface.co/transformers/
2. Vaswani et al., "Attention Is All You Need", NeurIPS 2017
3. TensorFlow Documentation: https://www.tensorflow.org/