Customer Support Response Assistant

By: Darshak Desai
NUID: 002021808

Fine-Tuning a Large Language Model on a Large-Scale Customer Support Dataset

1. Introduction

Customer support systems play a critical role in maintaining customer satisfaction and operational efficiency. Many organizations rely on human agents to generate consistent, professional responses to common customer issues such as delayed orders, password resets, billing concerns, and subscription cancellations. However, manual handling of repetitive support queries can be time-consuming and inconsistent in tone and quality.

The objective of this project was to fine-tune a pre-trained Large Language Model (LLM) to generate high-quality, professional customer support responses. The project includes:

- Dataset preparation and preprocessing
- Fine-tuning on a domain-specific dataset
- Hyperparameter experimentation
- Baseline vs fine-tuned evaluation
- Error analysis
- A deployable inference interface

The final system demonstrates significant improvements over the baseline model and includes a production-style Gradio interface with controllable tone, decoding parameters, and export functionality.

2. Dataset Preparation

2.1 Dataset Selection

For this project, I selected the **Bitext Customer Support LLM Chatbot Training Dataset**, available on Hugging Face. This dataset contains approximately 26,000+ customer support question-response pairs across multiple categories, including:

- Account issues
- Billing inquiries
- Subscription management
- Order tracking
- Password resets

This dataset was chosen because:

- It contains realistic customer support interactions
- It is sufficiently large to support meaningful fine-tuning
- It covers diverse support intents
- It represents a real-world production use case

Using a large domain-specific dataset improves generalization and reduces overfitting compared to very small datasets.

2.2 Data Cleaning and Preprocessing

The dataset was cleaned and standardized by:

- Removing duplicate entries
- Normalizing whitespace
- Filtering extremely short or incomplete examples
- Converting text into consistent JSONL format

Each example was converted into an instruction-style format:

- `instruction`: High-level support instruction
- `input`: Customer message
- `output`: Ground-truth support response

Example format:

```
{
  "instruction": "You are a helpful professional customer
support agent...",
  "input": "My order is delayed and tracking hasn’t
updated.",
  "output": "We apologize for the delay. Please provide
your order ID..."
}
```

This instruction-based format improves alignment with instruction-tuned models like FLAN-T5.

2.3 Data Splitting

The dataset was split into:

- 90% Training
- 5% Validation
- 5% Test

This ensures:

- Proper hyperparameter tuning
- Reliable evaluation
- No data leakage

3. Model Selection

3.1 Base Model

The selected base model was:

google/flan-t5-small

This is an instruction-tuned sequence-to-sequence transformer model.

3.2 Justification

The model was chosen because:

- It is instruction-following (important for structured prompting)
- It is lightweight and CPU-friendly
- It allows rapid experimentation
- It supports controlled text generation

Although larger models may achieve better performance, FLAN-T5-small provides a strong balance between performance and reproducibility in limited hardware environments.

4. Fine-Tuning Setup

Fine-tuning was implemented using the Hugging Face Trainer API.

Key components:

- Tokenization with max input and output lengths
- Padding and truncation handling
- DataCollatorForSeq2Seq
- Checkpointing per epoch
- Saving best-performing model

The final model for each configuration was saved in:

“outputs/checkpoints/configX/final_model”

5. Hyperparameter Optimization

Three different configurations were tested:

Config	Epochs	Learning Rate	Batch Size	Avg Quality Score
config1	2	5E-05	8	**2.84**
config2	2	1E-04	8	2.78
config3	3	1E-04	4	2.70
baseline	—	—	—	1.21

5.1 Observations

- config1 achieved the highest average quality score.
- Lower learning rate (5e-5) generalized better on the large dataset.
- Increasing learning rate slightly reduced response stability.
- More epochs did not necessarily improve generalization.

This demonstrates that moderate learning rates can prevent overfitting on large datasets.

6. Model Evaluation

6.1 Baseline Comparison

The baseline model (pre-fine-tuned FLAN-T5-small) achieved:

Average Quality Score ≈ 1.21

The best fine-tuned model (config1) achieved:

Average Quality Score ≈ 2.84

This represents more than a 2× improvement over baseline.

6.2 Qualitative Improvements

Fine-tuned model:

- Generates more domain-relevant responses
- Provides actionable steps
- Maintains professional tone
- Avoids generic responses

Baseline model:

- Often overly generic
- Frequently asked for irrelevant details
- Produced inconsistent tone

7. Error Analysis

Despite improvements, some failure patterns remain:

1. Overly verbose responses
2. Requests unnecessary details
3. Slightly repetitive phrasing
4. Excessively formal tone in simple queries

These issues suggest:

- Further dataset balancing could help
- Intent classification routing could improve specificity
- Reinforcement Learning from Human Feedback (RLHF) could refine tone

8. Inference Pipeline

A production-style Gradio interface was implemented.

Features include:

- Model selection dropdown (baseline vs fine-tuned)
- Tone control (Neutral / Friendly / Strict Professional)
- Temperature slider
- Top-p slider
- Privacy-aware toggle ("Ask for details only if relevant")
- Chat history tracking
- Export chat logs to file
- Model caching for performance

This interface simulates a real-world deployment environment and allows live comparison between baseline and fine-tuned models.

9. Limitations

- Quality score is a heuristic metric
- No human evaluation study performed
- Model size limits complexity of responses
- Domain limited to customer support

10. Future Work

- Fine-tune larger models (e.g., FLAN-T5-base or LoRA on 7B models)
- Add intent classification routing
- Implement retrieval-augmented generation
- Add human evaluation metrics
- Deploy via Hugging Face Spaces

11. Conclusion

This project demonstrates that domain specific fine tuning significantly improves response quality for customer support applications. The fine-tuned model achieved more than double the quality score of the baseline model and was deployed through a flexible and production-style user interface.

The project satisfies all functional requirements, including dataset preparation, model selection, hyperparameter optimization, evaluation, error analysis and inference pipeline implementation.

12. References

- Hugging Face Transformers Documentation
- Bitext Customer Support LLM Chatbot Training Dataset
- Gradio Documentation