Automated Email Routing using LLMs: Forzen-gpt, LoRA Fine-Tuning, and BERT Classification

Mattia Lomuto VR543327 Carlo Stasi VR543606
Natural Language Processing - LLM Project 2025-26

February 9, 2026

1 Introduction

The objective of this project is to automate the routing of customer support tickets based on their content. We aim to classify emails into five distinct departments: *Technical Support, Customer Service, Billing and Payments, Sales and Pre-Sales,* and *General Inquiry.* The project compares three approaches: **zero-shot and few-shot prompting strategies** applied to both frozen models and **parameter-efficient fine-tuned (LoRA)** architectures, contrasted with a traditional **discriminative BERT-based classifier**.

2 Dataset and Preprocessing

We utilized the `Tobi-Bueck/customer-support-tickets` dataset from Hugging Face, filtering for English emails only.

3 Methodology

3.1 Agent 1: Prompting (Frozen GPT-2)

We employed `gpt2` (124M) and `distilgpt-2` (82M) in a frozen state. No gradient updates were performed. To guide the generation, we designed two specific prompt templates.With Frozen GPT-2, since there is no training involved, the only way to improve accuracy is by experimenting with different prompts.

3.1.1 Zero-Shot Prompt

In the Zero-Shot setting, the model is provided only with the instruction and the target email, relying entirely on its pre-training. This two prompt-type was used, obviusly, both in Frozen GPT-2 and LoRa.

```
Instruction: Act as an email routing assistant for a company. Your task is to
read the following email and assign it to exactly ONE of these departments: -
Technical Support
- Customer Service
- Billing and Payments
- Sales and Pre-Sales
- General Inquiry
Instructions: - Base your decision only on the content of the email. - Output
ONLY the department name. Email: <Insert Email Here>

Department: (to predict)
```

3.1.2 Few-Shot Prompt

In the Few-Shot setting, we delineate the desired output format by providing context examples before the query.

```
Instruction: Classify each email into exactly ONE of these departments: Technical
Support, Customer Service, Billing and Payments, Sales and Pre-Sales, General
Inquiry.

Examples:
Email: My internet connection keeps dropping. Can you help?
Department: Technical Support

Email: I want to purchase your premium plan. What are the options?
Department: Sales and Pre-Sales

Email: My credit card was charged twice for the same order.
Department: Billing and Payments

Email: How do I reset my account password?
Department: Customer Service

Current Task:
Email:
Object: (given the object)
Body: (given the body)
Department: (to predict)
```

Extended Few-Shot Limitations We also explored a denser prompting strategy, increasing the number of examples to three per category ($k = 15$). Contrary to expectations, this yielded no performance gains. We attribute this decrease of performance to two factors:

- **Context Window Saturation:** The extended prompt length approached the model's 1024-token limit, likely causing the truncation of the target email or the dilution of initial instructions (vanishing gradient effect on the prompt).
- **Information Overload:** For smaller architectures (124M parameters), an excessively articulated prompt may introduce noise rather than clear patterns, overwhelming the model's attention mechanism.

3.2 Agent 2: LoRA Fine-Tuning (Generative)

For the generative fine-tuning task, we utilized both Distil-GPT-2 and GPT-2. To manage computational resources efficiently, we applied **Low-Rank Adaptation (LoRA)**

- **Configuration:** Rank $r = 16$, Alpha $\alpha = 32$, Dropout $= 0.1$.
- **Target Modules:** Weights were injected into the attention mechanisms (`c_attn`).
- **Training:** The model was trained for 3 epochs using mixed precision (FP16) to optimize memory usage on the GPU

3.3 Class Imbalance and Mitigation

Initial exploratory analysis revealed a severe class imbalance, with *Technical Support* dominating the dataset (6,476 samples) compared to underrepresented classes like *General Inquiry* (340 samples). Preliminary experiments showed that training on this raw distribution yielded a deceptively high baseline accuracy ($\approx 60\%$). We attribute this to the model overfitting the majority

class prior probability (*Technical Support*), effectively ignoring minority classes. To mitigate this bias and force the model to learn semantic features rather than frequency heuristics, we implemented a hybrid balancing strategy:

- **Undersampling:** The majority class (*Technical Support*) was randomly downsampled to approximately 2,500 samples to reduce its dominance.
- **Data Augmentation (Random Word Swapping):** To upsample minority classes (e.g., *General Inquiry*, *Sales*) to the target of $\approx 3,000$ samples without causing overfitting via exact duplication, we applied a stochastic word-swapping technique. For each augmented sample, two random tokens in the email body were swapped. This introduced regularization noise, preventing the model from memorizing exact positional patterns while preserving the overall semantic context.

It is important to note that while this balancing strategy led to a decrease in overall accuracy compared to the imbalanced baseline (due to the removal of the majority class bias), it was crucial for improving the recall of minority departments.Upon evaluating this strategy using metrics such as accuracy and F1-score (see Table 1.), the performance on the minority (undersampled) classes was found to be null. This confirmed that the model was heavily biased, systematically defaulting to the majority class. However, by applying this technique, we successfully recovered a measurable predictive signal from these features, effectively mitigating the model's tendency to ignore the underrepresented data. However, by applying this technique, we successfully recovered a measurable predictive signal from these features, effectively mitigating the model's tendency to ignore the underrepresented data.

Table 1: Impact of Balancing Strategy on Minority Classe

(a) Imbalanced

Department	**Pre.**	**Rec.**	**F1**	**Sup.**
Billing and Payments	0.70	0.50	0.58	291
Customer Service	0.36	0.09	0.15	395
General Inquiry	0.00	0.00	0.00	29
Sales and Pre-Sales	0.00	0.00	0.00	93
Technical Support	0.59	0.94	0.73	849

(b) Hybrid Balancing Strategy

Department	**Pre.**	**Rec.**	**F1**	**Sup.**
Billing and Payments	0.90	0.67	0.77	291
Customer Service	0.32	0.07	0.12	395
General Inquiry	0.06	0.10	0.07	29
Sales and Pre-Sales	0.09	0.67	0.17	93
Technical Support	0.76	0.57	0.65	849

3.4 Agent 3: DistilBERT Classifier (Discriminative)

We fine-tuned a `distilbert-base-uncased` model. Unlike the generative approach, this model uses a classification head on top of the encoder to output a probability distribution over the 5 class labels directly.

Implementation details

- **Tokenization**: We used a max sequence length of 512 tokens with dynamic padding which is the max value declared for BERT classifier.
- **Training**: The model was fine-tuned for 6 epochs with a learing rate of $2e^{-5}$ and a batch size of 16.
- **Regularization**: We employed the `load_best_model_at_end` strategy, monitoring the accuracy on validation to prevent overfitting. As observed duting training, the model achieved convergence at Epoch 6, reaching a validation accuracy of approximately 83%.

Considerations on metric selections: Accuracy vs. Validation loss

During the fine-tuning of the discriminative agent (DistilBERT), we faced a decision regarding the model selection strategy. Specifically, we observed a slight divergence between the *Validation Loss* and the *Validation Accuracy* in the final epochs of training.

As shown in our training logs, the model achieved its lowest Validation Loss at Epoch 4 (0.585), with an accuracy of 80.3%. However, by continuing training to Epoch 6, the Validation Loss increased marginally to 0.593 (+0.008), while the Accuracy improved significantly to nearly 83% (+2.6 percentage points).

We chose to prioritize **Accuracy** as the primary metric for the `load_best_model_at_end` strategy. This decision is grounded in three key considerations:

1. **Project Requirements:** The primary objective of the assignment is to evaluate the agents based on their ability to correctly route emails. As explicitly stated in the project specifications, the comparison between models must be performed in terms of accuracy on the test set.

2. **Operational Effectiveness:** In a real-world classification scenario, the discrete decision (which department receives the ticket) is more critical than the probability calibration (measured by Loss Value). A model might have a slightly higher loss because it is "less confident" about some correct predictions or "overly confident" about a few wrong ones, yet still classify a larger total number of emails correctly.

3. **Magnitude of Improvement:** The increase in validation loss observed at Epoch 6 was negligible (< 0.01), suggesting that the model was not suffering from detrimental overfitting. Conversely, the gain in accuracy was substantial. Sacrificing nearly 3% of correct classifications to save a marginal amount of loss would have resulted in a suboptimal agent.

Consequently, the final model checkpoint used for testing corresponds to the epoch with the highest validation accuracy, ensuring the best possible performance on the ticket routing task.

4 Experimental Setup

The experiments were conducted on a local and cloude workstation:

- **Hardware:** NVIDIA GeForce RTX 5070 and v5-e 1-TPU of Google Colab.
- **Environment:** WSL on Visual Studio Code and Google Colab.
- **Optimization:** FP16 precision and Gradient Accumulation were used to fit the 355M parameter model within VRAM constraints.

5 Results and Comparison

The performance of the three agents is summarized in Table 2.

Table 2: Performance comparison: Accuracy, Memory and Training Time

Model name	**Accuracy**	**Mem (MB)**	**Time (min)**
GPT2_prompting	18.59	864.64	2.58
DistilGPT2_prompting	**25.47**	863.35	1.50
GPT2_loRa	63.61	1607.89	10.54
DistilGPT2_lora	**65.48**	1547.01	9.42
GPT2_loRa_balanced	48.40	1756.45	15.12
DistilGPT2_lora_balanced	**50.03**	1568.06	14.06
Distil_BERT	**82.92**	2031.00	8.56
Distil_BERT_balanced	69.58	2138.64	8.24

5.1 Analysis

- **Prompting:** The frozen model struggled with specific domain rules, often hallucinating categories or failing to adhere to the strict output format.
- **LoRA Fine-tuning:** With minimal trainable parameters ($< 1\%$), LoRA achieved decent baseline accuracy. However, the **hybrid balancing strategy** was the critical factor: while it slightly reduced global accuracy, it successfully recovered predictive signal for minority classes, significantly boosting **Recall** where the imbalanced model failed completely.
- **DistilBERT:** As expected for classification tasks, the discriminative model provided high efficiency and accuracy but lacks the generative flexibility of GPT-2.

6 Conclusion

This project demonstrated that Parameter-Efficient Fine-Tuning (LoRA) bridges the gap between large frozen models and specialized classifiers. While DistilBERT is strictly more efficient for classification, the LoRA-tuned GPT-2 Medium demonstrated that generative models can be adapted to routing tasks but with some adjustament, if the dataset had been balanced,maybe the model would have performed better.

References

[1] NLP Course, "LLM Project Definition," Univ. of Verona, 2025.

[2] Hugging Face, "Tobi-Bueck/customer-support-tickets".

[3] Hu et al., "LoRA: Low-Rank Adaptation of Large Language Models".