Arizona State University

Group: 22
Project: Text2SQL Demo

Jinal Vyas
Nishtha Kuberbhai Chaudhary
Omkar Shelar
Rebecca Pires Dos Santos
Vrishir Bhaskar Iyer

Presentation Outline

- Problem Definition
- System Architecture & Algorithms
- Datasets (Description, Size and Preprocessing)
- Evaluation (Metrics, Experiments, Preliminary Findings)
- UI Interfaces or Data Visualization (Preliminary Designs)
- Project Plan: Tasks, Deadlines, Division of Work

Problem Definition

Existing large language models like ChatGPT and Gemini, while powerful, are not optimized for schema-aware Text-to-SQL translation and often produce semantically invalid queries. They also pose issues of cost, privacy, and limited customization.

Solution: To address this, we develop a custom Text2SQL pipeline and fine-tune an LLM for domain-specific SQL generation. A traditional ML model is used as a baseline to evaluate and highlight the performance gains achieved through LLM fine-tuning.

Dataset Description

- Spider 1.0
  - https://yale-lily.github.io/spider
  - Complex and cross-domain text-to-SQL dataset.
  - 10,181 questions and 5,693 SQL queries on 200 databases.
  - The queries are unique and complex.
  - Multiple tables which cover 138 different domains.

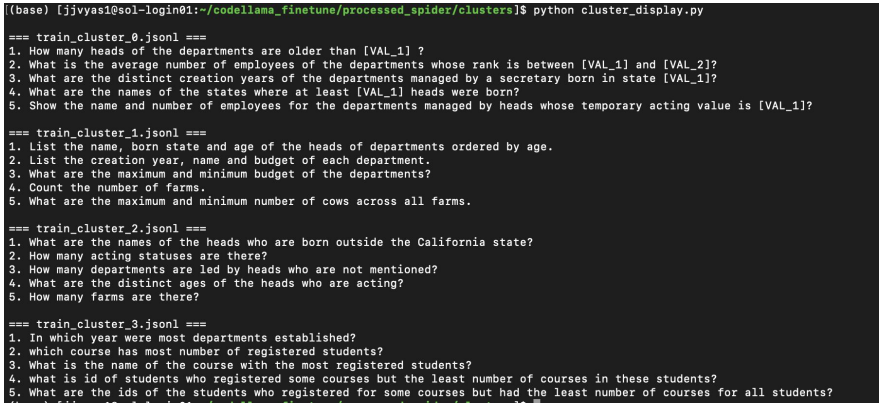
- BIRD
  - https://bird-bench.github.io/
  - Targets real-world conditions: larger, messier databases, dirty or ambiguous values, need for external knowledge, and SQL efficiency
  - BIRD stresses robustness, execution, and scalability beyond Spider 1.0.

System Architecture for Data-Segmentation Field

- We followed two approaches to better our previously fine-tuned LLM model
  - Natural Language Query based clustering
  - SQL Query based clustering
- Both were done against train data samples (disjoint from test data samples) to allow for schema generalization
- We feed the DB schema as well as the question for CodeLlama to better understand the DB structure
- Reasons to cluster:
  - SPIDER has diverse SQL patterns so it helps to create “specialists” models to help with similar queries.

Fine-Tuning based on Natural Language

1. **Clustering:**
   a. Extract question from prompt
   b. Embed using **MiniLM**
   c. Run K-Means (**k = 4**)
   d. Create clusters and centroids
2. **Cluster-Specific Fine-Tuning:**
   a. Fine-tune one LoRA per cluster

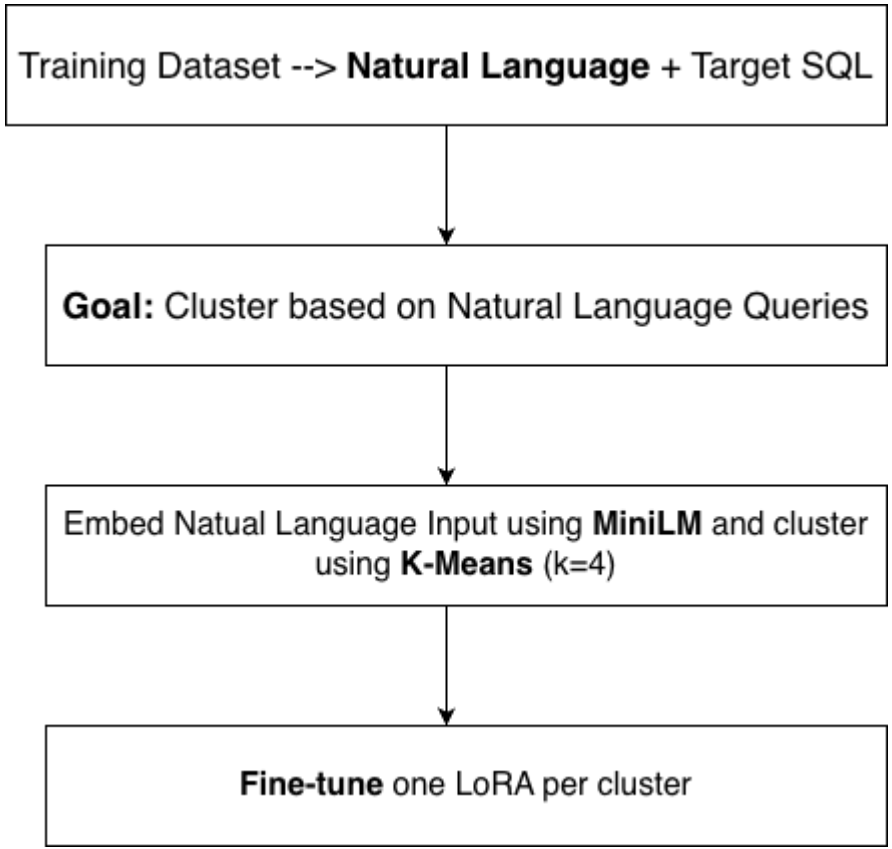


3. **Inference time Router:** Selecting the Right cluster/LoRA

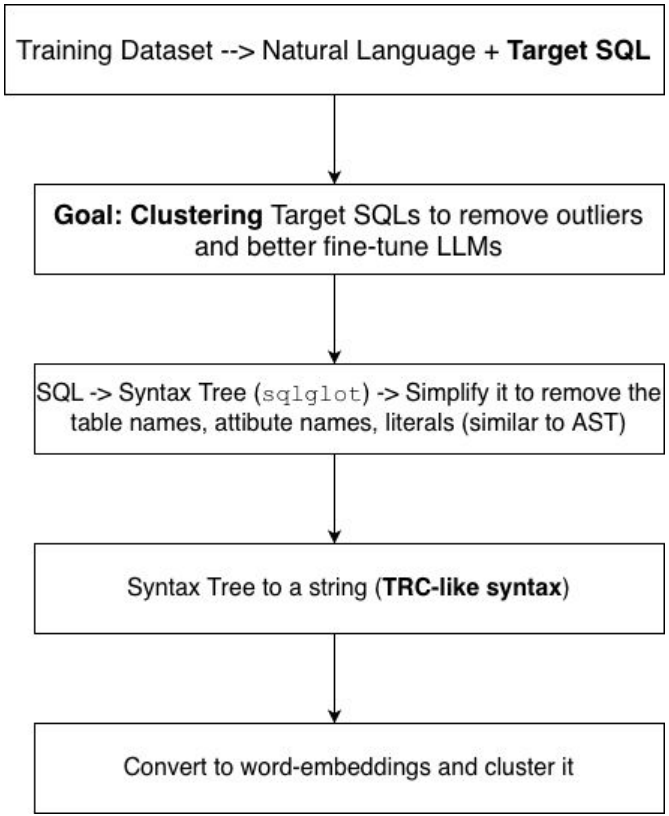
   Question → nearest centroid → pick LoRA + model → SQL

4. Use the correct LoRA adaptor with CodeLlama to get SQL

Fine-Tuning based on Natural Language



Fine-Tuning based on Target SQL



- select T2.first_name , T2.last_name from candidates AS T1 join people AS T2 on T1.candidate_id = T2.person_id

  AND (IDENT_1 . IDENT_4 EQ IDENT_2 . IDENT_3) PROJ PROJ_COL PROJ_COL

- select T2.Name , count(*) from debate_people AS T1 join people AS T2 on T1.Affirmative = T2.People_ID group by T2.Name

  AND (IDENT_4 . IDENT_6 EQ IDENT_2 . IDENT_7) GROUPBY IDENT_2 . IDENT_1 PROJ PROJ_COL AGG_*

Fine-Tuning based on Target SQL (contd.)

- Algorithms used for clustering:
  - HDBScan – Too much concentration of clusters

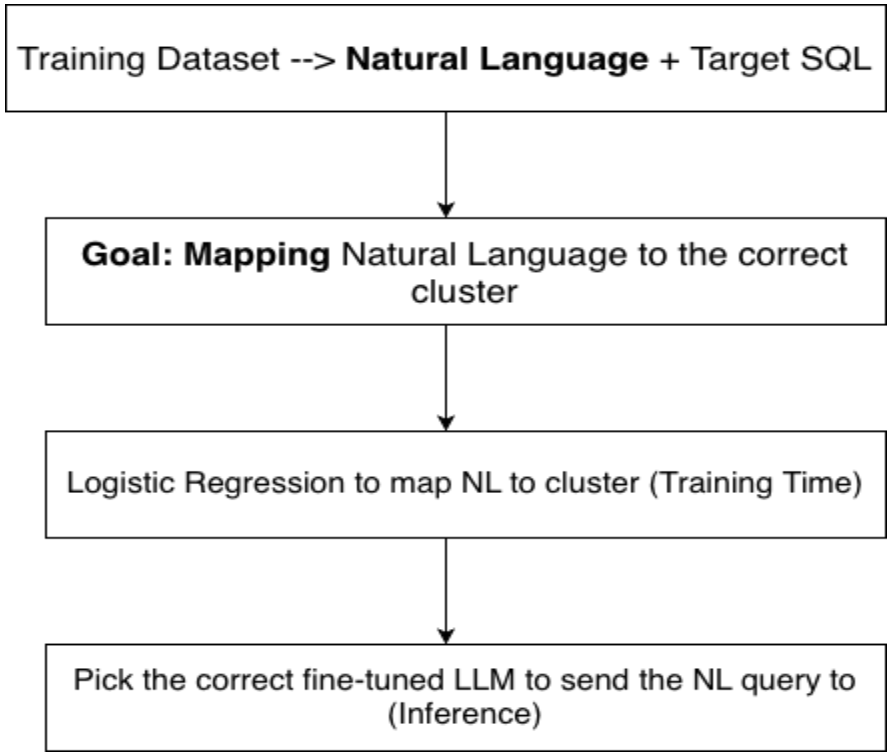

```
{
  "num_clusters_excluding_noise": 3,
  "num_points_total": 6492,
  "num_outliers": 73,
  "points_per_cluster": {
    "-1": 73,
    "0": 10,
    "1": 6401,
    "2": 8
  }
}
```

Fine-Tuning based on Target SQL (contd.)

- Algorithms used for clustering:
  - Hierarchical (agglomerative) Clustering

```
{
  "num_clusters_excluding_noise": 3,
  "num_points_total": 6492,
  "points_per_cluster": {
    "0": 1537,              # Simple projections / aggregations
    "1": 3223,              # Joins and complex selection
    "2": 1732               # Grouping and Aggregations
  }
}
```

Fine-Tuning based on Target SQL (contd.)



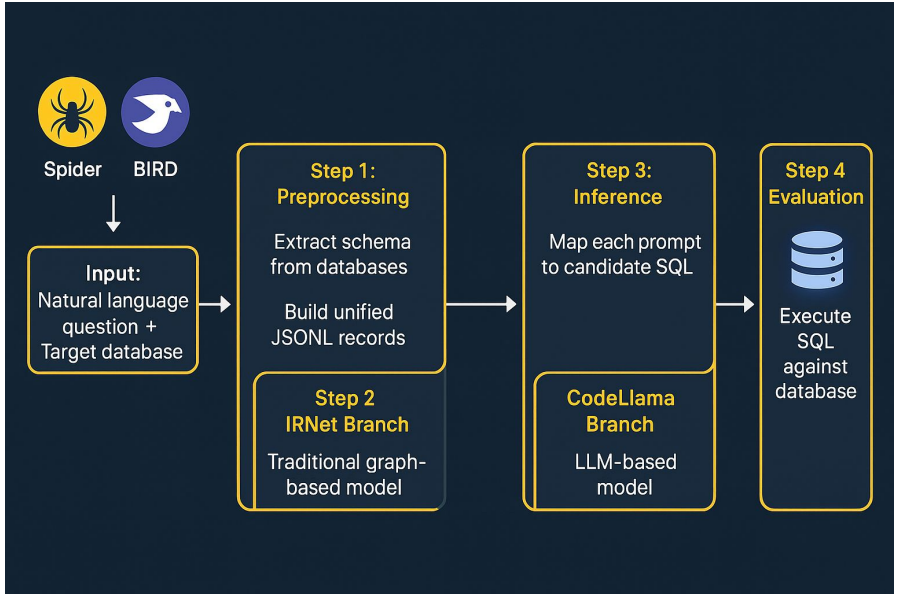
Validation accuracy: 0.7662
Training:Validation Split => 80:20

	precision	recall	f1-score
0	0.71	0.67	0.69
1	0.82	0.87	0.84
2	0.71	0.66	0.68
accuracy			0.77
macro avg	0.74	0.73	0.74
weighted avg	0.76	0.77	0.76

System Architecture for LLM-Spider and BIRD

Phase 1 – CodeLlama with Spider 1.0

- **Goal:** Establish a baseline Text-to-SQL performance on a standard academic benchmark.
- **Dataset:** Spider 1.0 – 200 databases, 138 domains, complex cross-domain SQL queries.
- **Pipeline:**
  - Preprocess Spider → serialize schema + question into a unified prompt.
  - Fine-tune **CodeLlama-7B** using LoRA on train split.
  - Generate SQL for dev split; evaluate with **EX.**
- **Outcome:**
  - Demonstrates how well CodeLlama learns structured SQL on a *clean, curated* benchmark.
  - Builds our reference point for later comparison with BIRD.



Phase 2 – CodeLlama with BIRD

Goal: Test CodeLlama on a **large, realistic** benchmark.

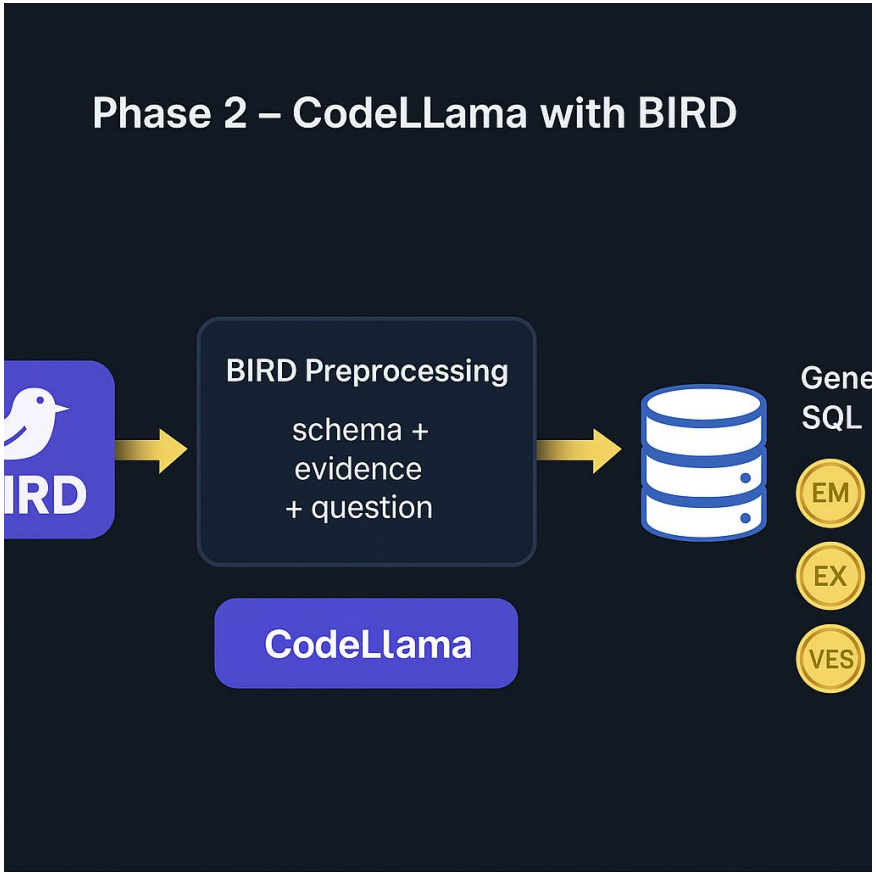
Dataset: BIRD – 12,751 question–SQL pairs, 95 large databases, 33.4 GB across 37+ professional domains (finance, healthcare, education, etc.).

Key Changes vs Spider:

- Much **larger schemas and table counts** → heavier prompts.
- Additional fields like **evidence** and more complex value reasoning

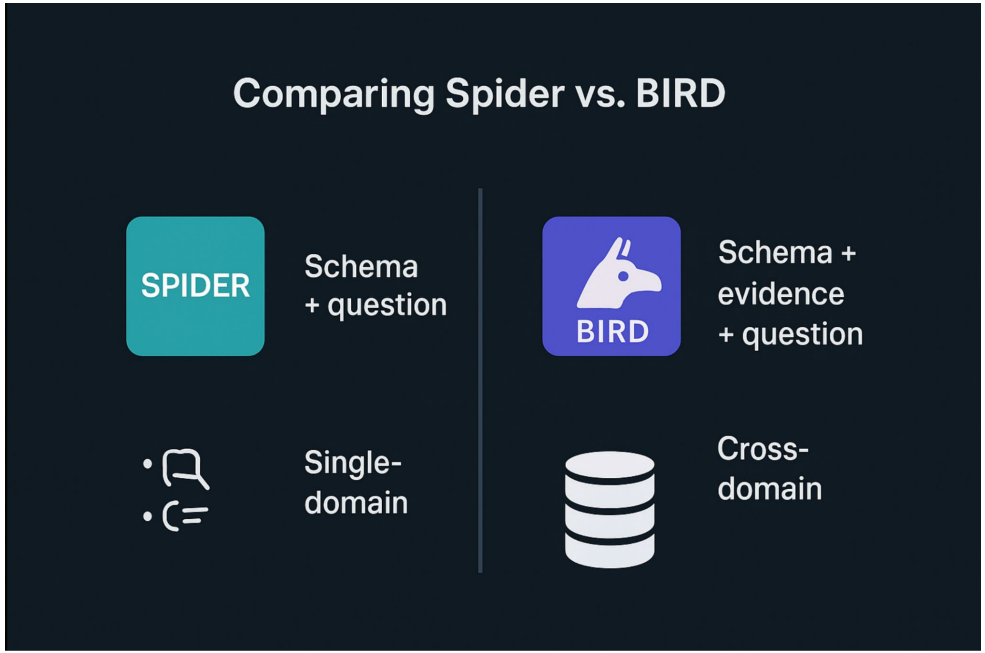
Pipeline Adjustments

- New `preprocess_bird.py` to build prompts from `train.json`, `dev.json`, and `/train_databases`, `/dev_databases`.
- Careful schema pruning + delexicalization to fit context window.
- Evaluate with **EX** (Execution Accuracy).



Phase 3 – Comparing CodeLlama on Spider vs BIRD

- **Task Setup:**
  - Same base model (**CodeLlama-7B**) and similar fine-tuning strategy.
  - Different data characteristics and evaluation metrics.
- **Spider vs BIRD:**
  - **Spider:** smaller, cleaner schemas; strong focus on cross-domain generalization; evaluated mainly with **EX**.
  - **BIRD:** larger, real-world databases; more complex value reasoning and evidence; evaluated with **EX.**
- **Key Insights:**
  - CodeLlama achieves **higher EX on Spider**, showing it handles curated academic benchmarks well.
  - On BIRD, **EX highlights partial correctness** revealing where the model struggles with scale and value reasoning.
  - Together, the two datasets show the trade-off between **benchmark performance** and **real-world readiness**.



Evaluation Metrics

Setup

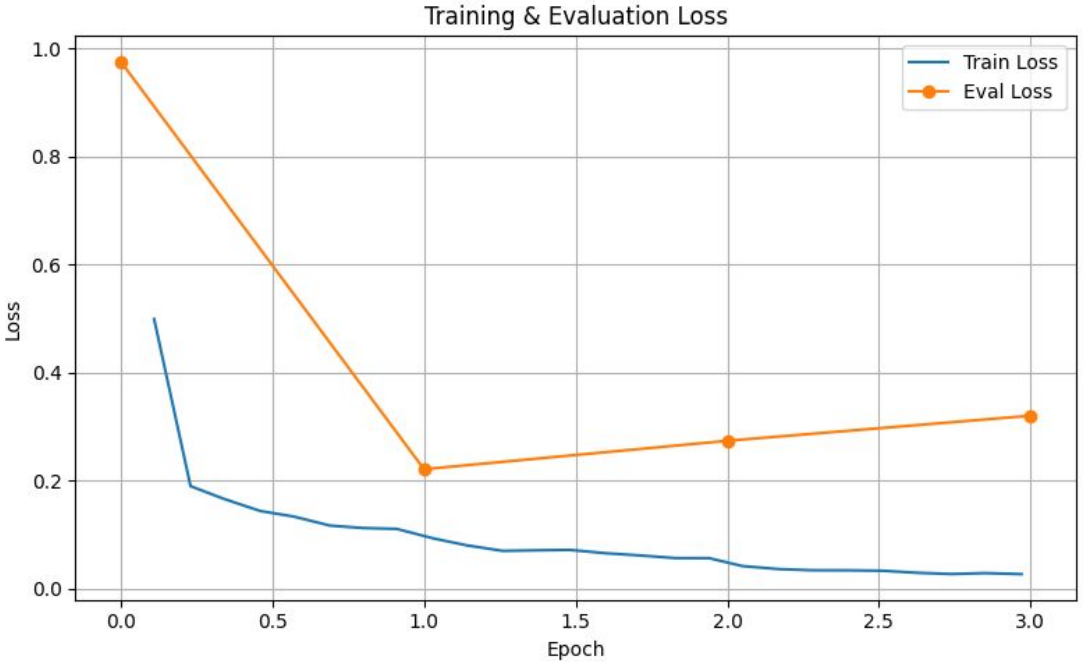
- **Base Model:** IR-NET (Microsoft Research), CodeLlama-7B-hf (Meta)
- **Fine-Tuning:** LoRA adapters
- **Datasets:** Spider 1.0 and BIRD
- **Metrics: Execution Accuracy (EX)** → Compares results after executing both queries on real databases.

Metric	Execution Accuracy (EX) Score
Spider 1.0 (With ML)	52.7%
Spider 1.0 (LLM Fine-tuned)	69.3%
Data Segmented Fine-tuned Spider 1.0	68.9%
BIRD (LLM Fine-tuned)	39.60%

Training & Evaluation Loss

The training loss (blue) drops quickly, showing strong convergence, while evaluation loss (orange) decreases initially then rises slightly indicating mild overfitting after epoch 1.

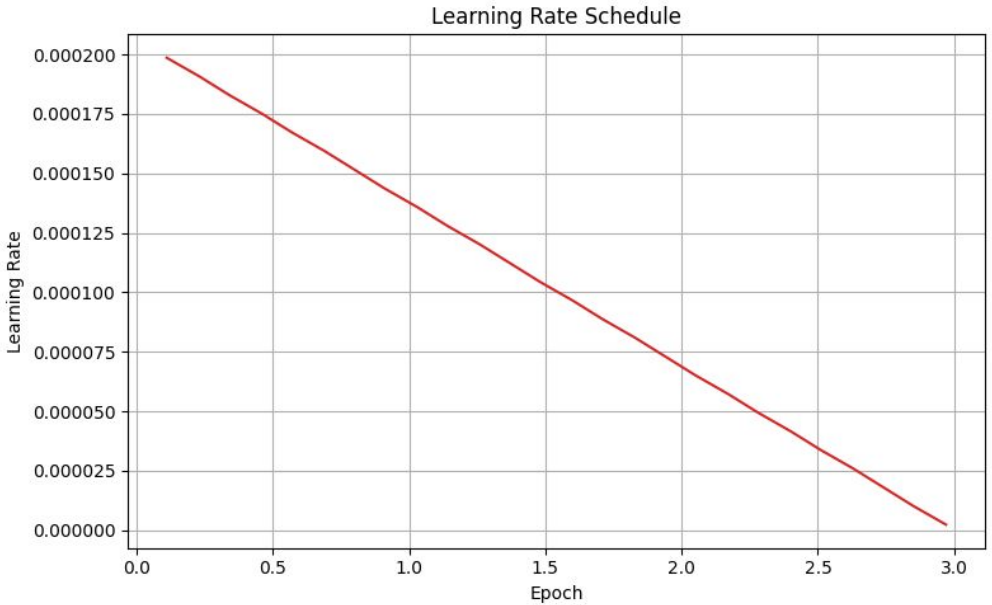
This trend suggests the model learned efficiently, with best generalization likely around epoch 1 and 2.



Learning Rate Schedule

The learning rate (red) decays linearly from 2×10^{-4} to 0 across 3 epochs, confirming the linear scheduler worked as intended.

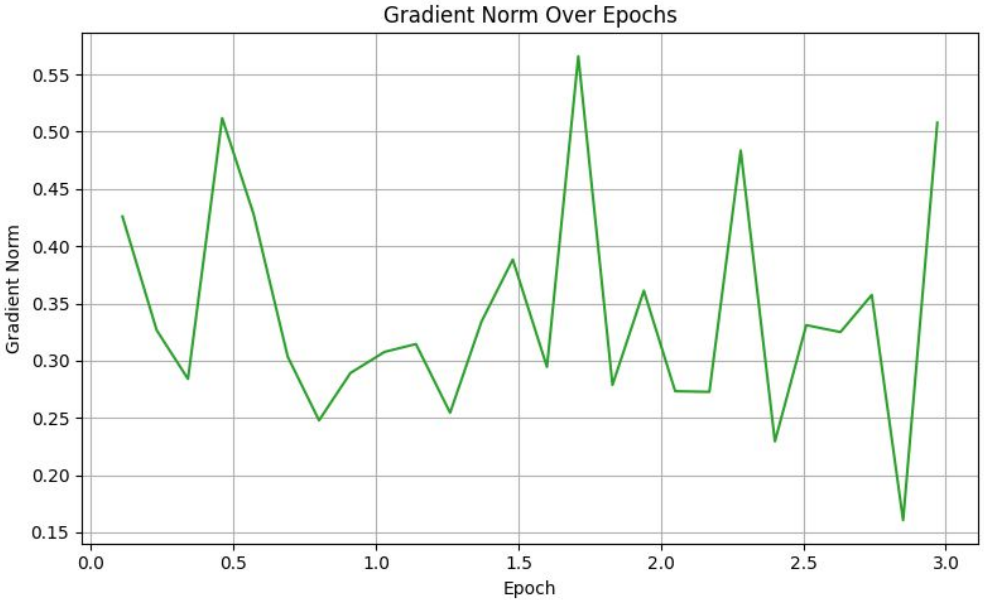
This smooth decay helps stabilize training by allowing large updates early and fine-tuned adjustments later.



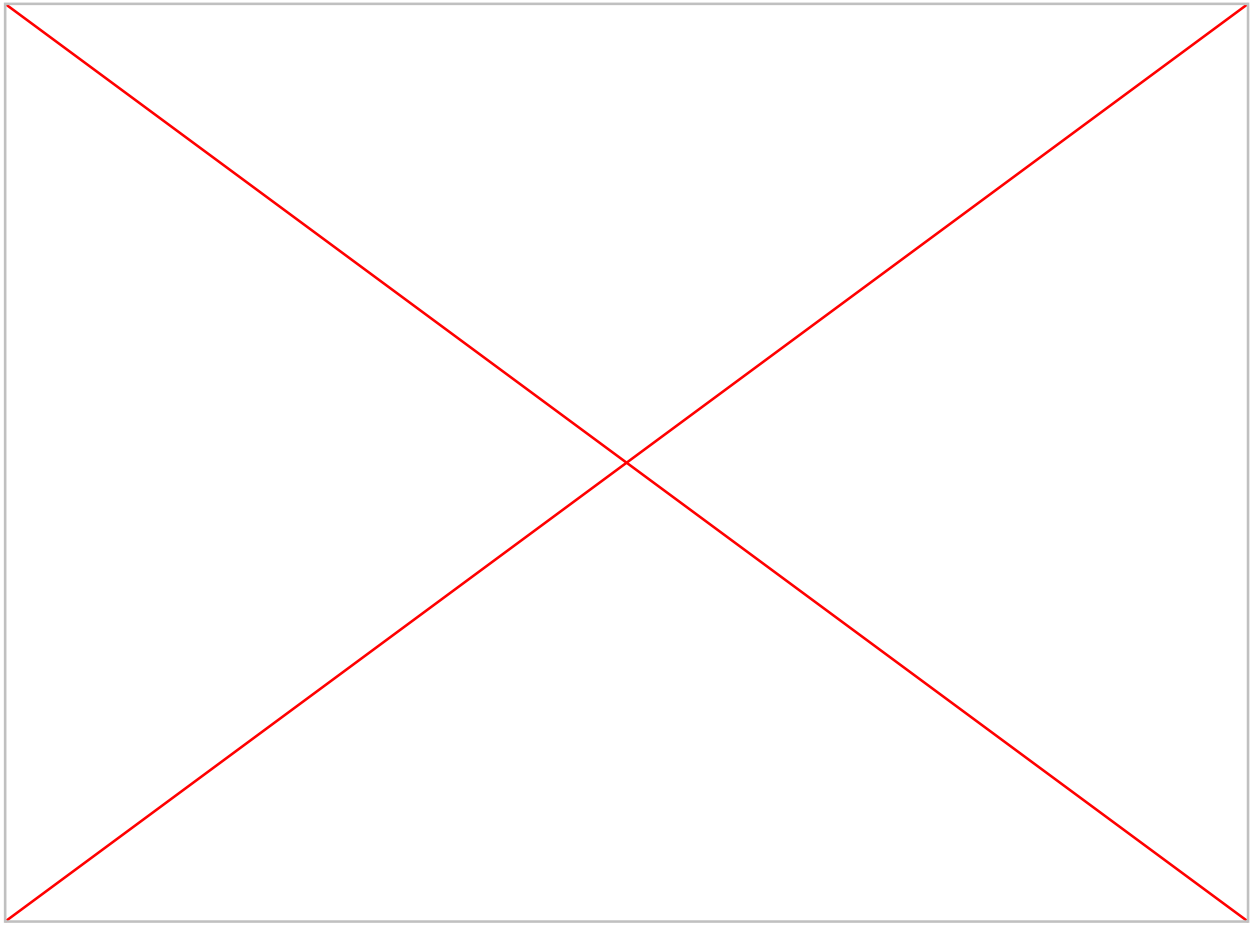
Gradient Norm Over Epochs

Gradient norms fluctuate moderately (≈ 0.2 – 0.55), indicating healthy parameter updates without exploding or vanishing gradients.

This stability shows that our optimizer, LoRA configuration, and learning rate schedule were well-balanced.



UI Interface Demo



Future Work and Enhancements

1. Hybrid Prompt-plus-Finetune Strategies

- We can explore hybrid approaches that **combine prompt engineering with fine-tuning** to further enhance model performance.
- Through this, we hope to identify a balanced method that maximizes performance while maintaining computational efficiency.

2. Experimenting with close-ended LLMs through APIs

- Several LLMs like GPT-4, Deepseek are expected to perform much better given their extremely large number of parameters, however due to their close-ended access, we restricted them for applying to a research project where exploration needs to be significantly conducted.

Project Plan: Tasks, Deadlines, Division of the Work

Task	Deadline	Responsible
Data Preparation	October 21st	All
Traditional ML Model Training	October 27th	Rebecca Pires dos Santos, Nishtha Chaudhary, Omkar Shelar
LLM Fine Tuning	November 3rd	Jinal Vyas and Vrishir Iyer
Evaluation and Comparison	November 21st	All
Improvement and Research on Advancement Areas	December 5th	All

All contributed in all areas. Responsible refers to the team member responsible for initiating the task and making sure it is completed in the planned time.

References

[1] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. R. Radev, “Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, Oct.–Nov. 2018.

[2] V. Zhong, C. Xiong, and R. Socher, “Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning,” arXiv preprint arXiv:1709.00103, Sep. 2017.

[3] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch, “Natural Language Interfaces to Databases – An Introduction,” arXiv preprint cmp-lg/9503016, Mar. 1995.

[4] X. Xu, C. Liu, and D. Song, “SQLNet: Generating Structured Queries from Natural Language without Reinforcement Learning,” arXiv preprint arXiv:1711.04436, Nov. 2017.

[5] T. Yu, M. Yasunaga, K. Yang, R. Zhang, D. Wang, Z. Li, and D. R. Radev, “SyntaxSQLNet: Syntax Tree Networks for Complex and Cross-Domain Text-to-SQL Task,” arXiv preprint arXiv:1810.05237, Oct. 2018.

[6] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J.-G. Lou, T. Liu, and D. Zhang, “Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation,” arXiv preprint arXiv:1905.08205, May 2019.

[7] B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson, “RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers,” arXiv preprint arXiv:1911.04942, Nov. 2019.

[8] X. V. Lin, R. Socher, and C. Xiong, “Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing,” arXiv preprint arXiv:2012.12627, Dec. 2020.

[9] R. Cao, L. Chen, Z. Chen, Y. Zhao, S. Zhu, and K. Yu, “LGESQL: Line Graph Enhanced Text-to-SQL Model with Mixed Local and Non-Local Relations,” in Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, Aug. 2021, pp. 2541–2555.

[10] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, Oct. 2014, pp. 1532–1543.

[11] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, “ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators,” arXiv preprint arXiv:2003.10555, Mar. 2020.

[12] T. Yu, R. Zhang, K. Yu, D. Zhang, Z. Wang, W. Zhang, Y. Zhao, D. Lin, M. Sun, D. Zhou, et al., “Spider 2.0: A Large-Scale Cross-Domain Text-to-SQL and Database Grounding Dataset,” arXiv preprint arXiv:2209.12391, 2022.

[13] L. Shi, Z. Tang, N. Zhang, X. Zhang, and Z. Yang, “A Survey on Employing Large Language Models for Text-to-SQL Tasks,” ACM Computing Surveys, vol. 58, no. 2, article 54, Sep. 2025, pp. 1–37, doi: 10.1145/3737873. [Online]. Available: https://doi.org/10.1145/3737873.

[14] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. Advances in Neural Information Processing Systems 36 (2024) [Online]. Available: https://arxiv.org/abs/2305.03111

Codebase

- https://github.com/santoreb/Text_to_SQL