Fine-tuning Language Models for Text-to-SQL Generation

-Ashwin Nair

Project Overview & Motivation

Objective

Fine-tune language models to convert natural language questions into SQL queries with explanations using Gretel's synthetic dataset.

Why Text-to-SQL?

- **Democratizes data access** Non-technical users can query databases
- Reduces development time Faster analytics and reporting
- Improves accuracy Structured generation vs manual SQL writing

Key Challenge

Generate syntactically correct SQL queries that match the semantic intent of natural language questions.

Dataset & Domain Focus

Gretel Synthetic Text-to-SQL Dataset

- Total Size: 100,000 examples across 85+ domains
- **Selected Domains**: Financial services, healthcare, finance, insurance (4,318 examples)
- Why Domain Filtering? Focus on business-critical sectors with similar query patterns

Data Structure

Input: Schema + Natural Language Question

Output: SQL Query + Explanation

Example

```
Schema: CREATE TABLE employees (emp_id INT, name VARCHAR(100), department VARCHAR(50)) Question: "Find all employees in Engineering" Output: SQL: SELECT * FROM employees WHERE department = 'Engineering'; Explanation: Filters employees table...
```

Model Selection & Architecture

Selected Models	
Model	Rationale
CodeT5-Small	Pre-trained on code, specialized for programming tasks
FLAN-T5-Small	Instruction-tuned, strong general language understanding

CodeT5-Small -60million param

FLAN-T5-Small-80 Million param

Training Configuration

- Train/Val/Test Split: 70%/10%/20% (3,022/432/864 examples)
- **Training Setup**: 3 epochs, batch size 4, gradient accumulation,val_steps=50, AdamW Optimizer
- **Hardware**: T4 GPU with mixed precision (FP16)

Enhanced Prompting Strategy

Convert this natural language question to SQL using the given database schema.

Database Schema: [SCHEMA]

Question: [QUESTION]

Please generate a SQL query with explanation:

Evaluation Methodology

Multi-Dimensional Evaluation Framework					
Metric	Weight	Purpose			
Syntax Correctness	20%	Can SQL be parsed?			
Structural Similarity	40%	Do components match reference?			
Exact Match	10%	Perfect query match			
Explanation BLEU	30%	Quality of natural language explanation			

Combined Score Formula

Score = 0.2×Syntax + 0.4×Structural + 0.1×Exact + 0.3×BLEU

Structural Analysis

- Extracts tables, functions, conditions from SQL
- Captures semantic correctness beyond syntax

Loss Reduction Comparison:						
Model	Initial Training Loss	Final Training Loss	Reduction			
CodeT5	0.797	0.338	58% reduction			
FLAN-T5	14.958	0.782	95% reduction			
Model	Initial Validation Loss	Final Validation Loss	Reduction			
CodeT5	0.640	0.312	51% reduction			
FLAN-T5	10.292	0.613	94% reduction			

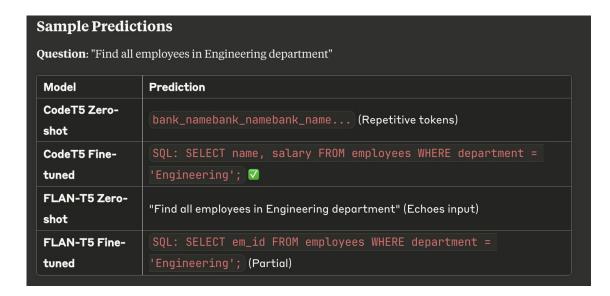
Results & Performance Comparison

Zero-Shot vs Fine-Tuned Performance							
Model	Syntax Correctness	Structural Similarity	Exact Match	Combined Score			
CodeT5 (Zero-shot)	0.5%	20.3%	0.0%	8.2%			
CodeT5 (Fine- tuned)	99.3%	63.8%	10.0%	54.9%			
FLAN-T5 (Zero- shot)	4.9%	20.3%	0.0%	9.1%			
FLAN-T5 (Fine- tuned)	88.9%	34.9%	0.0%	38.0%			

Key Insights

- CodeT5 Superior: 44.7% improvement vs 28.9% for FLAN-T5
- Syntax Mastery: Both models achieved near-perfect syntax after fine-tuning
- Structural Understanding: CodeT5 better captures SQL query structure
- Domain Specialization: Code pre-training provides significant advantage

Qualitative Analysis & Examples



Observations

- Zero-shot models fail completely on code generation tasks
- CodeT5 produces more accurate column selections and query structure
- FLAN-T5 struggles with proper SQL syntax and column naming
- Fine-tuning essential for any reasonable performance

Challenges and Lessons

Started from one domain and shifted to 4-5 for bigger dataset

Shifted from t5-small to flant5-small

Improved the prompt for better result

Moving away from plain BLEU to custom logic+BLEU

High-Impact, Near-Term Improvements

- 1) Advanced Model Architectures
- Larger Models: CodeT5-Base/Large, StarCoder, CodeLlama
- 2) Advanced Prompting & Context
- 3) More advanced logic for evaluation

Conclusions:

Code-Specialized Models Significantly Outperform General Language Models

Fine-tuning is Absolutely Essential for Text-to-SQL Tasks

Start with a smaller model and eventually scaling up