Project 2 - TextToSQL System

1. **Platform**: Python, Salesforce-CodeT5-base , Lora, PEFT, Spider Text-to-SQL database

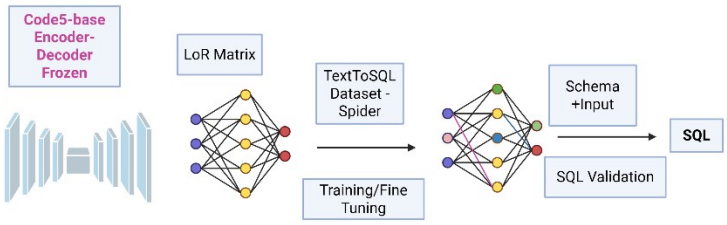

Figure 1: LoRA Supervised Fine Tuning of CodeT5-base model using Spider dataset

2. **Model Selection**
   SalesForce-CodeT5-base is a transformer-based encoder-decoder model based on Google's Text-to-Text Transfer Transformer T5 model. T5 was trained on a custom dataset called Colossal Clean Crawled Corpus for NLP tasks like QA, summarization, translation, and sentiment analysis.

   CodeT5 is finetuned on T5 for Code-specific tasks, including Code summarization, Code generation, clone detection, and translation using CodeSearchNet that contains code from multiple languages such as Java, Python, and JavaScript.

   The Code aware model was chosen for easier fine-tuning for text to text-to-SQL task as compared to a generic language model.

3. **Dataset**
   Spider is the industry standard Text to SQL dataset and has been used in many TextToSQL projects already. It has 10k records. The queries are performance optimised.

4. **Training**
   CodeT5-base was supervised fine-tuned on a randomly sampled subset of 20% of the Spider dataset, consisting of ~2k records due to limited GPU RAM, and was tested on 200 questions. Gradient accumulation was applied to address this issue, but that caused the performance to degrade severely hence was not pursued.

The Low-Ranked Adaptation (LoRA) technique was used for fine-tuning by introducing new, smaller matrices to train, keeping the original large model of 220 million parameters frozen. Mixed precision training was employed, keeping 16-bit precision for faster training and inference, and 32-bit for critical tasks like log loss scaling to prevent underflow/overflow.

Dropout is added to the LoRA layers to prevent overfitting. The training was done for 3 epochs, and the model was saved in cloud storage for inference.

5. **Model Performance**
   The test performance steadily increased as training progressed, culminating at 85% for precision. Recall and F1.

6. **Inference and SQL Validation**
   SQLLite was used to set up the required database tables, and the schema was generated to be passed in the prompt.

   The saved model is loaded and run with user questions. The SQL returned is validated against the database, and the error is returned to the user.

7. **Scopes of Improvements**
   More elaborate prompt engineering, longer fine-tuning on a larger dataset, can easily achieve near-perfect performance.