CSE 5525: Assignment 3

Krish Patel

1 Data Statistics and Processing (8pt)

Instructions: Use Table 1 and Table 2 to describe the data statistics before and after any pre-processing respectively. Use the T5 tokenizer to report the statistics. For the statistics after pre-processing, if you did different pre-processing for different models, you need to indicate them separately. The gray text in each row is there to guide you and should be removed in your submitted report. Depending on your pre-processing, some numbers may be identical across tables.

Statistics Name	Train	Dev
Number of examples	4225	466
Mean sentence length	10.96	10.91
Mean SQL query length	60.90	58.90
Vocabulary size (natural language)	868	444
Vocabulary size (SQL)	644	393

Table 1: Data statistics before any pre-processing. Statistics computed using simple whitespace tokenization for word counts and word sets for vocabulary sizes.

Statistics Name	Train	Dev
Number of examples	4225	466
Mean sentence length	23.10	23.07
Mean SQL query length	217.37	211.05
Vocabulary size (natural language)	796	470
Vocabulary size (SQL)	556	396

Table 2: Data statistics after pre-processing with T5 tokenizer. Natural language inputs are prefixed with "translate English to SQL:" before tokenization. The high token-to-word ratio for SQL queries arond 3.6x is due to T5's subword tokenization splitting SQL identifiers—for example, `flight_1.flight_id` tokenizes to 8 tokens: `['_flight', '_', '1.', 'f', 'light', '_', 'i', 'd']`—and keywords like `SELECT` split into 3 tokens `['_', 'SEL', 'ECT']`. This occurs as T5 was pre-trained on natural language rather than SQL queries.

2 T5 Fine-tuning and Training From Scratch (8pt)

Instructions: Use Table 3 and Table 4 to describe your data processing steps (if any) and the implementation details, respectively for the fine-tuned T5 model, and the T5 model trained from scratch. The gray text in each row is there to guide you and should be removed in the submitted report. Be clear enough that we can replicate your approach in PyTorch using only your descriptions.

Design choice	Description
Data processing	Natural language inputs are prefixed with "translate English to SQL:" before tokenization. Encoder inputs and decoder outputs are shortened to a maximum of 1024 tokens to fit memory and execution limits. The decoder starts with a beginning-of-sequence token, `<extra_id_0>`, and decoder targets include an EOS token at the end during training. No additional preprocessing is done to maintain SQL structure.
Tokenization	Inputs to the encoder and decoder are tokenized using the default T5 tokenizer. Natural language inputs are prefixed with "translate English to SQL:" before tokenization. Encoder inputs are truncated to 1024 tokens. The decoder has `<extra_id_0>` as the beginning-of-sequence token, and decoder targtes are appended with EOS token. Padding is applied at the batch level using pad_sequence with `PAD_IDX=0`.
Architecture	Entire pre-trained T5-small model was finetuned without freezing any layers. So, no architectural modification are done in fine-tuning.
Hyperparameters	Learning rate: 1e-3 (0.001), Batch size: 16 (train), 8 (test), weight decay: 0.0, Optimizer: default AdamW, Scheduler: Cosine with 0 warmup epochs, Max epochs: 20, patience ephocs: 5, Generation: Beam search with num_beams=4, max_length=512, early_stopping=True, Loss: Cross-entropy on non-padded tokens.

Table 3: Details of the best-performing T5 model configurations (fine-tuned)

Design choice	Description
Data processing	Same as fine-tuning (Table 3).
Tokenization	Same as fine-tuning (Table 3).
Hyperparameters	Same as fine-tuning (Table 3) except Learning Rate: 1e-4 (0.0001), and initial weight for scratch were randomly initialized using T5Config from google-t5/t5-small. Additional hyperparameters such as repetition_penalty=1.2, no_repeat_ngram_size=3 were applied in test inference to reduce repititive outputs.

Table 4: Details of the best-performing T5 model configurations (from scratch)

3 Large Language Model (LLM) Prompting (14pt)

3.1 In-Context Learning (ICL)

Instructions: Provide in Table 5 the instruction prompt(s) that you used for ICL.

If the prompt you used for zero- and few-shot prompting is identical, except for the examples, there's no need to repeat it. If you made small modifications between zero- and few-shot, please provide them separately. For all entries, you need to specify the corresponding values of k.

Shot	Prompt
k=0,1,3	Prompt 0 (p=0): Convert English questions to SQL queries. Prompt 1 (p=1): Convert English questions to SQL queries for a flight database. Use proper SQL syntax with correct table joins and aliases. Output only the SQL query. Prompt 2 (p=2): Convert English questions to SQL queries for a flight database. Use proper SQL syntax with SELECT, FROM, WHERE, and JOIN clauses. Always use table aliases like `flight_1`, `city_1`, `airport_service_1`, etc. Join tables correctly through their foreign key relationships. End all queries with a semicolon. Schema:(First 500 characters of the flight_database.schema) (included when k1 for any p=0,1,2) (1st example question) (1st example answer) (2nd example question) (2nd example answer) [till kth example]

Table 5: Instruction prompts used for zero- and/or few-shot prompting.

Example selections: Please provide a clear, detailed, and succinct description of how you selected the examples when $k > 0$.

For few shot prompting for k example selection, I chose first k data points from the training set, as I wanted ability to debug and get consistent performance.

3.2 Best Prompt and Ablation Study

Instructions: Report the best prompt you used in Table 6. If the best prompt you used is the same as the one specified in Table 5, you can just copy the best prompt and label it. If it is different (e.g. you designed another prompt that is better), you should clearly describe how you created it and what are the methods you used in the caption.

You will also need to clearly and succintly describe in Table 7 the ablation experiments that you performed by removing different parts of the prompt. For that, you need to first highlight the parts of the prompt that you ablated for each experiment in a distinct color[1], as shows the placeholder example in Table 5, and second, provide the description by referring to the highlighted part. When reporting your results in Table 8, you will need to refer to your ablations variants.

Prompt (best p=1, k=3)
Convert English questions to SQL queries for a flight database. Use proper SQL syntax with correct table joins and aliases. Output only the SQL query. (1st example question) (1st example answer) (2nd example question) (2nd example answer) (3rd example question) (3rd example answer)

Table 6: The best prompt corresponds to Prompt Type 1 from Table 5 with $k = 3$ (three-shot). This configuration provides both explicit formatting guidance (green) and sufficient examples (blue) to demonstrate the database knowledge and query patterns.

Color	**Description**
ForestGreen	Removing the format guidance to evaluate whther the model can learn proper formatting conventions solely from 3 examples. This evaluates the sufficiency of few-shot learning without much explicit instructions. Corresponding to Prompt p=0 with k=3 from 5
Blue	Removing all 3 training examples while keeping both tasks description and formatting guidance. This evaluates the model's ability to generate queries with instructions alone, without any demonstrations. Corresponding to Prompt p=1 and k=0 from 5. It is expected that this degrades performance as flight database has complex schema and hence need demonstration for proper references.

Table 7: Ablation variants. Put a clear description of the ablation experiments you did. If it helps to describe more clearly, you can also provide an example of the relevant part of the prompt before and after the ablation.

[1] `https://www.overleaf.com/learn/latex/Using_colors_in_LaTeX`

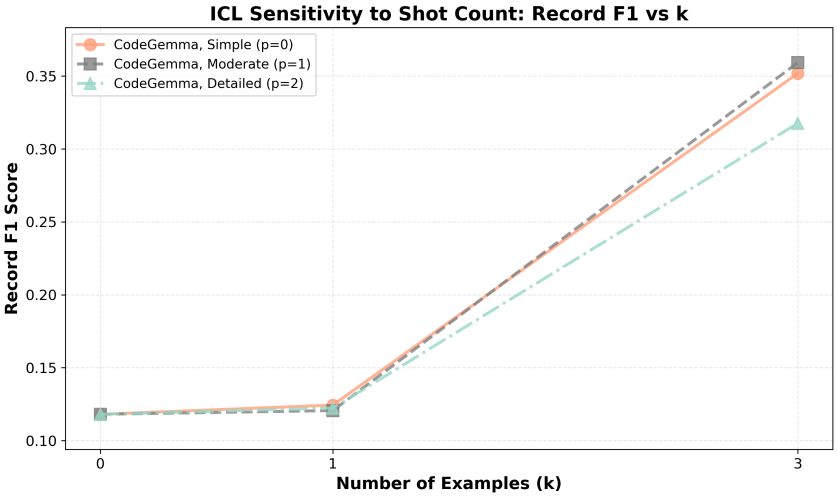
4 Results and Analysis (20pt)

Quantitative Results: Use Table 8 to report your test and development set results. Your test results should match with the results on gradescope. For the development set, you should also report results from experiments you conducted to arrive at your final configuration. When reporting experiments, you should replace "variant" with brief and meaningful descriptions of whatever hyperparameter or setting that you varied. For ICL, you should specify what is the parameter k used, and what the full model corresponds to. For T5, the full model refers to the best model you described in Section 2. For T5, if you experimented with different design choices, you can add rows specifying the variants and what you tried. The text in gray is only for example purpose, and should be removed and replaced with your own choices. You may add more rows if needed.

System	Query EM	F1 score
Dev Results		
CodeGemma LLM Prompting		
Full model (k=3, p=1)	00.00	00.36
Zero-shot, simple (k=0, p=0)	00.00	00.12
Zero-shot, moderate (k=0, p=1)	00.00	00.12
Zero-shot, detailed (k=0, p=2)	00.00	00.12
1-shot, simple (k=1, p=0)	00.00	00.12
1-shot, moderate (k=1, p=1)	00.00	00.12
1-shot, detailed (k=1, p=2)	00.00	00.12
3-shot, simple (k=3, p=0)	00.00	0.23
3-shot, moderate (k=3, p=1)	00.00	00.36
3-shot, detailed (k=3, p=2)	00.00	00.32
Ablation Studies (from best: k=3, p=1)		
No formatting instructions	00.00	00.35
No examples)	00.00	00.12
T5 Fine-tuned		
Full model (lr=1e-3)	00.03	00.80
Variant (lr=1e-4)	00.02	00.69
Variant (lr=5e-5)	00.02	00.58
T5 From Scratch		
Full model (lr=1e-4)	00.01	00.48
Variant (lr=1e-3)	00.00	00.12
Variant (lr=5e-5)	00.00	00.31
Test Results		
ICL	00.00	00.30
T5 fine-tuning	00.00	00.73
T5 from scratch	00.00	00.36

Table 8: Development and test results for all models. LLM prompting shows performance across shot counts (k) and prompt types (p). T5 models show variants with different learning rates. Best configurations are labeled as "Full model".

ICL sensitivity to k: For ICL, please provide a plot of the Record F1 on the development set that the model achieved with different values of k. The x-axis should be k, and the y-axis the Record F1. The prompts and examples used for this plot should correspond to the ones you described in Subsection 3.1.


ICL Sensitivity to Shot Count: Record F1 vs k
CodeGemma, Simple (p=0)
CodeGemma, Moderate (p=1)
CodeGemma, Detailed (p=2)
Record F1 Score
0.35
0.30
0.25
0.20
0.15
0.10
0
1
3
Number of Examples (k)

Qualitative Error Analysis: Conduct a detailed error analysis for each of the three models. Identify common error types and discuss possible reasons for these errors in Table 9.

You must identify at least three classes of errors for the queries, and use examples to illustrate them. It must be clear what model makes the errors you are analyzing. If you identified the same type of error for different models, you don't need to duplicate the descriptions, but you need to clearly specify an example for each of the model, indicate the statistics for each model, and specify to which model each statistics correspond to. You may add more rows to the table.

Error Type	Models	Example	Description	Statistics
Wrong Table/Column Names	All models	**Question:** "what flights are available tomorrow from denver to philadelphia" **ICL k=0 Generated:** `SELECT * FROM flights WHERE departure_city = 'Denver'` **Expected:** Use `flight` table (not `flights`) with joins through `airport_service` and `city`	Model hallucinates not related or non existent table names (`flights` instead of `flight`) or column names (`departure_city`). Dramatic few-shot learning: zero-shot 85.4%, one-shot 1.3%, three-shot 0.9%. All T5 models achieve 0.0% errors regardless of training differences (fine-tuned or scratch, all learning rates). Shows schema names are surface-level patterns and can easily be learned through supervised training.	ICL CodeGemma k=0, p=1: 398/466 (85.4%) ICL k=1, p=1: 6/466 (1.3%) ICL k=3, p=1: 4/466 (0.9%) T5 fine-tuned (lr=1e-3): 0/466 (0.0%) T5 fine-tuned (lr=1e-4): 0/466 (0.0%) T5 scratch (lr=1e-3): 0/466 (0.0%) T5 scratch (lr=1e-4): 0/466 (0.0%)

Missing or Incorrect Joins	All models	**Question:** "what airlines fly between toronto and pittsburgh"
		ICL k=3 Generated:
		`SELECT DISTINCT flight_1.flight_id FROM flight flight_1, city city_1 WHERE city_1.city_name = 'PITTSBURGH'`
		Expected: Must join `flight` to `airport_service` to `city` for both cities `airport_service` to `city` for both cities, plus `days` table **Question:** "show me the ground transportation schedule in philadelphia in the morning on wednesday"
		T5 FT Generated:
		`SELECT DISTINCT ground_service_1.transport_type FROM ground_service ground_service_1, city city_1 WHERE city_1.city_name = 'PHILADELPHIA'`
		Missing: Day constraint (Wednesday) and time constraint (morning) - no joins to temporal tables
			Most prevalent error (75.5-94.8%). Counterintuitively, ICL worsens with examples: 75.5% at k=0 to 94.8% at k=3. T5 fine-tuned: 91.6-92.7% for both lr. T5 scratch (lr=1e-3): 92.7%, but (lr=1e-4): 0.0%—not correct, the undertrained model avoids joins entirely. Multiple joins still remains a problem across all approaches.	ICL k=0, p=1: 352/466 (75.5%)
				ICL k=1, p=1: 439/466 (94.2%)
				ICL k=3, p=1: 442/466 (94.8%)
				T5 fine-tuned (lr=1e-3): 427/466 (91.6%)
				T5 fine-tuned (lr=1e-4): 432/466 (92.7%)
				T5 scratch (lr=1e-3): 432/466 (92.7%)
				T5 scratch (lr=1e-4): 0/466 (0.0%)

Incomplete WHERE Clauses	All models	**Question:** "list all arrivals to baltimore on thursday morning arriving before 9am"
		ICL k=0 Generated:
		`SELECT * FROM Arrivals WHERE AirportCity = 'Baltimore' AND ArrivalTime < '09:00:00'`
		Expected: Must include `days` table join for Thursday and `departure_time < 900` constraint **Question:** "what is the cost of united airlines flight 415 from chicago to kansas city thursday night"
		T5 FT Generated:
		`...WHERE flight_1.airline_code = 'UA' AND flight_1.flight_number = 4...`
		Missing: Night time constraint (`departure_time NOT BETWEEN 601 AND 1759`), incorrect flight number (4 vs 415)
			Removes constraints from question. ICL: 26.8% errors at k=0, improves to 12.9-14.4% at k≥1. T5 fine-tuned: 9.0-9.2% (both lr). T5 scratch (lr=1e-3): 8.6%, but (lr=1e-4): 34.1%. Shows constraint solvability requires sufficient training signal; undertrained model learned partial query structures but not complete constraint handling.	ICL k=0, p=1: 125/466 (26.8%)
				ICL k=1, p=1: 60/466 (12.9%)
				ICL k=3, p=1: 67/466 (14.4%)
				T5 fine-tuned (lr=1e-3): 42/466 (9.0%)
				T5 fine-tuned (lr=1e-4): 43/466 (9.2%)
				T5 scratch (lr=1e-3): 40/466 (8.6%)
				T5 scratch (lr=1e-4): 159/466 (34.1%)

Truncated Queries	All models	**Question:** "show me flights from boston to atlanta and return flights from atlanta to boston" **ICL k=3 Generated:** `...WHERE f.departure_city = 'Atlanta' AND f.arrival_city = 'Boston' AND f.departure_date_time >=` [query cuts off mid-clause, no closing semicolon] **Expected:** Complete query with proper closing and all clauses finished	Early termination due to token limits (`max_new_tokens=250` for ICL, `max_length=512` for T5). Significant: ICL k=0 shows 0.0% (simpler queries fit), k≥1 shows 24.9-25.1%. T5: 99.8-100% across all seven configurations. It is a technical limitation, not modeling error—easily fixable by increasing generation limits.	ICL k=0, p=1: 0/466 (0.0%) ICL k=1, p=1: 116/466 (24.9%) ICL k=3, p=1: 117/466 (25.1%) T5 fine-tuned (lr=1e-3): 465/466 (99.8%) T5 fine-tuned (lr=1e-4): 466/466 (100.0%) T5 scratch (lr=1e-3): 466/466 (100.0%) T5 scratch (lr=1e-4): 466/466 (100.0%)

Table 9: Qualitative error analysis on development set (466 examples) across all configurations: ICL CodeGemma (k=0,1,3), T5 fine-tuned (lr=1e-3, 1e-4), T5 scratch (lr=1e-3, 1e-4). Zero-value statistics are highly informative: 0.0% truncation at k=0 reveals simpler zero-shot queries, 0.0% missing joins in undertrained T5 (lr=1e-4) indicates join avoidance rather than correct generation. The core unsolved challenge is join composition (75-95%