International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

SFT for Improved Text-to-SQL Translation

[1]Puneet Kumar Ojha, [2]Abhishek Gautam, [3]Ankit Agrahari, [4]Parikshit Singh

Submitted: 23/12/2023 **Revised**: 29/01/2024 **Accepted**: 07/02/2024

Abstract: Large Language Models (LLMs) have proved significant proficiency when comes to code generation especially in Structured Query Language (SQL) for databases and recent successful Text-to-SQL method involves fine-tuning pre-trained LLMs for SQL generation tasks. Transforming natural language text into SQL queries, has been attempted to solve with various learning techniques including Few-shot learning[1], fine tuning. In this paper we propose Supervised fine-tuning (SFT) as a better alternative for learning technique for text-to-SQL generation task using Code-Llama that pushes state of art accuracy on spider test suite to 89.6% on dev set which represent first instance of surpassing the earlier best-in-class with 5.5% higher score and 86.8% of exact match accuracy on dev set. Furthermore, we demonstrate that properly prompted LLM along with SFT provides far fewer hallucinations and much more robust LLM that can be used as a general tool for any text-to-SQL generation use case.

Keywords: *SQL, LLM, SFT, Language, Data Analytics, Translation.*

1. Introduction

Automatic SQL generation from natural language has been one of the most crucial needs to enhance database accessibility without the knowledge of data definition or querying methods. With advancement in LLM's conversational chatbots have bloomed and come up with easier ways to access the database and provide better data analytics.

Several training and optimization techniques have been demonstrated for achieving decent performance in text-to-SQL generation. RESDSQL[2] for example utilizing a distinct approach for connecting database schemas and dissecting the structure of queries, employing an improved encoding process with ranking and a decoding framework aware of skeleton structure, this was primarily achieved with the encoder-decoder model T5 by fine tuning the model in two stages cross encoder training followed by seq2seq training. PICARD[3] applied an innovative method involving progressive parsing to restrict auto-regressive decoding, while RASAT[4] merged self-attention mechanisms aware of database schemas with controlled auto-regressive decoders within the model's framework.

The development of massive LLMs such as GPT-3 [5], PaLM [6], ChatGPT [7], GPT-4 [8], and PaLM-2[9], each with billions of parameters, has led to significant strides in zero-shot and few-shot learning techniques, particularly in-context learning[10]. These approaches, especially few-shot prompting, are advantageous over fine-tuning because they require less computational power, are less likely to overfit training data, and can easily adjust to new datasets. This is especially beneficial for converting text into SQL queries due to the various dialects of SQL. However, a downside is that their performance may not be as high as desired. As an illustration, while CodeX[11] and ChatGPT [12] have demonstrated encouraging outcomes in converting text into SQL queries using in-context learning methods, they still fall short compared to fine-tuned models with moderately sized LLMs. SQL-PALM [13], the prior best-in-class, demonstrated considerable enhancements by employing both few-shot learning and fine-tuning on the PALM-2 [9], [13] model using the Spider dataset. Meanwhile, DIN-SQL adopts a least-to-most prompting strategy[14], dividing the Text-to-SQL task into smaller elements such as connecting schemas, categorizing queries, and breaking them down. Subsequently, it employs few-shot prompting specifically for each sub-task with customized prompts. Notably, DIN-SQL[15] is the first to surpass the effectiveness of fine-tuned state-of-the-art models in evaluations using a few-shot prompting approach.

In this paper we propose, Supervised fine-tuning as another option to regular fine-tuning for training LLM for better text-to-SQL generational task. We have used open Llama-V2 due to its several architectural advantages including pre-normalization, SwiGLU activation, and Rotary embeddings. The model, when trained, attained top-tier results on the Spider

[1]*B. Tech in Bioinformatics Co-Founder, Attentions Data Labs Pvt. Ltd*
Email ID: puneetkumar.2705@gmail.com
[2]*B. Tech in Computer Science, Data Scientist, Attentions Data Labs Pvt. Ltd, Indian Institute of Information Technology (IIIT) Una*
Email ID: devxabhishek@gmail.com
[3]*Masters in Machine Learning and Artificial Intelligence Co-Founder, Attentions Data Labs Pvt. Ltd, , Liverpool John Moores University*
Email ID: ankitagr2312@gmail.com
[4]*B. Tech in Computer Science Solution Architect, Attentions Data Labs Pvt. Ltd, United college of engineering Allahabad*
Email ID: parikshitcs0072@gmail.com

development set boasting a notable execution accuracy of 89.6% alongside a precise match accuracy of 86.8%.

2. SFT for Text-to-SQL

2.1 LLM's training techniques

2.1.1 Few shot prompting

LLM's prompting is a method of constraining a model to give desired outputs. First identified in [5], in-context learning leverages the capability of few-shot learning and zero-shot through prompting. This method integrates a limited set of examples and instructions inside the prompt, creating a 'context' that enables LLMs to adapt to new tasks and examples without any alterations to the model. As highlighted in [10], the efficacy of few-shot prompting is particularly more evident in LLMs above a specific size margin. The achievement of in-context learning has led to the innovation of advanced prompting techniques like two chain-of-thought prompting (CoT) [16], least-to-most prompting [14], and self-consistency prompting [17], which are efficient strategies for large-shot adaptation. For the Llama-7b model we were only able to get an accuracy score of 11.8% out-of-the box from few-shot prompting only. Although the model was able to generate the output but was very poor at understanding how to put joins and multiple clauses for filtering through the data.

2.1.2 Fine-tuning

Fine-tuning is a training method where the model parameters are changed slightly for a downstream task to improve the models performance on that task. LLMs have demonstrated exceptional capabilities across a range of difficult tasks, like those in BIG-bench [18]. This is largely attributed to the extensive knowledge gained from large-scale pre-training, which is then enhanced by instruction-based fine-tuning on various tasks, known as FLAN-fine-tuning. Fine-tuning has proven to be very much effective in neural networks, however in LLM's it often induces a lot of hallucination after output is generated in smaller models, resulting in poor model's generation quality and overall poorly generated queries, we measured an accuracy of 45.5% only when trained with fine-tuning (see Table 1).

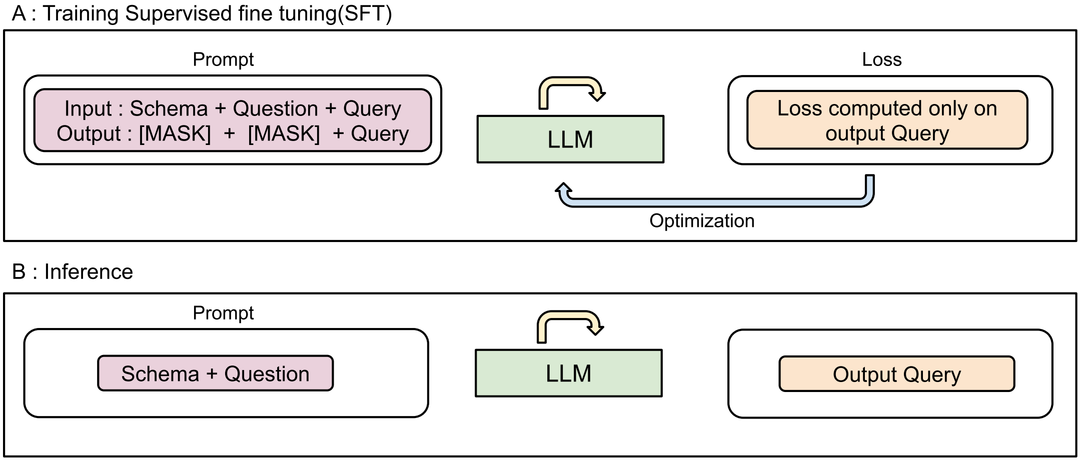
A : Training Supervised fine tuning(SFT)
Prompt
Input : Schema + Question + Query
Output : [MASK] + [MASK] + Query
LLM
Loss
Loss computed only on output Query
Optimization
B : Inference
Prompt
Schema + Question
LLM
Output Query

Fig 1: A Supervised Fine Tuning (SFT) of Llama model on spider **B** Inference prompting on Llama model

2.1.3 Supervised fine tuning

SFT, or Supervised Fine Tuning, entails modifying a model for a new downstream task by fine-tuning the LLM with labeled data. In general, the entire context is passed at once but the final loss is computed only over the label (see figure 1) that the model is required to generate this allows for the model to learn only the syntactic generation of label rather than entire statement, in our case the schema and question were masked and loss was computed only on the generated query. This allowed for much better learning and text-to-sql generations. Our efforts led to an impressive achievement of 89.4% accuracy (Table 1) on the Spider dev-set.

3. Experiments

3.1 Dataset

Analyzed the extensive, cross-domain Text-to-SQL benchmark known as Spider [19], consisting of 7000 training examples across 166 databases and 1034 evaluation samples ('Dev split') spanning 20 databases. Spider-SYN [20], an intricate iteration of the Spider Dev dataset, is generated by manually substituting synonyms within the natural language queries. Spider-realistic [21] selects 508 text-SQL pairings from the Spider Dev Split, omitting direct references to column names in the natural language questions. Additionally, Spider-DK [22] draws 535 question-SQL pairs from 10 databases in the Spider Dev split, adding domain knowledge to these pairings.

3.2 Model

Open Llama-V2 [23]is an open-source replication of the Llama model. Llama has shown very promising results across several benchmarks despite its smaller size compared to GPT-4, GPT-3 and chat-GPT models. For our task we chose Llama V2 with 7 billion parameters as roughly being the sweet spot for decent size to performance tradeoffs.

Code Llama [24] is finetuned on coding data representing a constellation of large language models , for large contexts ,code infilling, and zero-shot instruction following for programming tasks.

3.3 Baselines

For fine-tuning approaches, SQL-PALM [13] leverages the transformer-based PALM-2 [9] model, applying both fine-tuning and few-shot techniques for the text-to-SQL task. PICARD [3] employs incremental parsing to limit auto-regressive decoding, and RASAT [4] is a transformer model that fuses relation-aware self-attention with controlled auto-regressive decoders. Additionally, RESDSQL [2] innovatively separates schema linking from skeleton parsing, employing a decoding framework that is aware of the query structure and an encoding framework enhanced with ranking.

In the domain of in-context learning, a detailed evaluation of CodeX and GPT-3's text-to-SQL capabilities is presented in [25], while an in-depth analysis of ChatGPT's [7] performance is offered in [12]. DIN-SQL [15] methodically decomposes Text-to-SQL into subtasks such as employing few-shot prompting with GPT-4 [8] in tasks such as query classification, schema linking, self-correction , SQL generation, and decomposition. The Self-debugging methodology [26] incorporates error messages into prompts and executes successive iterations of few-shot prompting for error rectification. According to the data in Table 2, ChatGPT [12] utilizes the prompting techniques suggested by OpenAI. It's noteworthy that Self-debugging [26] focuses exclusively on Execution accuracy (EX).

3.4 Evaluation

We have utilized two primary evaluation metrics on the Spider test-suite: execution accuracy (EX) and exact match (EM). Execution accuracy (EX) evaluates if the predicted SQL query aligns precisely with the gold SQL query through their conversion into a specialized data structure. In contrast, exact match (EM) juxtaposes the outcomes of executing the predicted SQL query against the gold SQL query. It's worth highlighting that, in contrast, the EX metric is influenced by the values generated within the query, whereas the EM metric remains unaffected by this factor.

4. Results

We demonstrate execution accuracy of various learning methods on the Llama-7B model in Table 1. We can clearly see from the results in the table that Supervised fine tuning far outperforms regular fine-tuned model. In our testing, fine-tuning smaller models resulted in much more hallucinations and as such resulted in poor performance as compared to the SFT counterpart.

Methods	**Easy**	**Medium**	**Hard**	**Extra hard**	**All**
Few shot (out of box**)**	**29.4**	**9.0**	**4.0**	**1.8**	**11.8**
Fine Tuning	**66.1**	**42.6**	**38.7**	**29.5**	**45.5**
Supervised fine tuning	**94.8**	**91.0**	**86.2**	**80.1**	**89.4**

Table 1: Comparison of Llama-V2 7B performance on few-shot learning, fine-tuning and supervised finetuning on test suite accuracy spider dev set.

We delve into how our proposed method fares across different levels of difficulty in SQL query generation. These levels are determined by various factors, including: SQL keywords used, the incorporation of attributes aggregations or selections and the utilization of nested sub-queries. Table 2 illustrates comparative performance of proposed method against a standard few-shot prompting approach using CodeX-davinci and GPT-4, as well as against DIN-SQL[15] and the prior SOTA, SQL-PALM, on the Spider development set. Our method consistently outshines the alternatives at all levels of difficulty, showing significant improvements. This indicates that our method does not exhibit a bias towards any specific category of difficulty. Our model specifically improved in generation of hard and extra hard SQL's resulting in significant performance improvements over the alternatives, and previous SOTA by almost 11% and being almost 50 times smaller.

Table 3 reports the EM and EX results on spider dev-set for various LLM's for various non-seq2seq models and seq2seq models with our results.

Methods/model	Easy	Medium	Hard	Extra hard	All
Few-shot CodeX-davinci	84.7	67.3	47.1	26.5	61.5
Few-shot GPT-4	86.7	73.1	59.2	31.9	67.4
DIN-SQL[2] CodeX-davinci	89.1	75.6	58.0	38.6	69.9
DIN-SQL[2] GPT-4	91.1	79.8	64.9	43.4	74.2
Few-shot SQL-PaLM2	93.5	84.8	62.6	48.2	77.3
Fine-tuned SQL-PaLM2	93.5	85.2	68.4	47.0	78.2
SFT Llama 7b V2(Ours)	**93.5**	**89.9**	**85.6**	**80.1**	**88.5**
SFT Code Llama7b(Ours)	**96.0**	**90.8**	**90.2**	**75.9**	**89.6**

Table 2 : Accuracy on the Spider dev split test-suite: SQL results are classified into different levels. The first two rows represent the conventional few-shot prompting approach. Beginning six rows are from [13]

Approach	EX(dev set)	EM(dev set)
Non-seq2seq methods		
GRAPPA + RAT-SQL [27]	73.4	-
NatSQL + RAT-SQL + GAP [28]	73.7	75.0
GRAPPA + SMBOP [29]	74.7	75.0
RoBERTa + DT-Fixup SQL-SP [30]	75.0	-
ELECTRA + LGESQL [31]	75.1	-
S2SQL + ELECTRA [32]	76.4	-
Seq2seq methods		
T5-3B [33]	71.5	74.4
PICARD + T5-3B[33]	75.5	79.3
PICARD + RASAT [34]	75.3	80.5
RESDSQL-3B	78.0	81.8
RESDSQL-3B + NatSQL	80.5	84.1

Our proposed method		
Llama-7B v2 (SFT)	**86.7**	**88.5**
Code Llama	**86.8**	**89.6**

Table 3: Comparison of various models performance on spider dev-set for text-to-SQL, non-sequence evaluation metrics include Exact Match (EM) and Execution Accuracy (EX) and seq2seq methods performance from [2]

5. Conclusion

We present a LLM based model SFT Code Llama-7B and SFT Open Llama 7B v2 for text-to-SQL task which leverages Llama transformer supervised fine tuning. We demonstrate significant performance improvements by simply changing the learning method to adopt the model to new data. Our model being even 50 times smaller compared to PALM-2 outperforms the competition setting a newer SOTA score on the spider test suite of 89.6% in execution accuracy and 86.8% in exact match. More importantly SFT Code-Llama-7B was able to produce very decent results, when prompted in the exact same way demonstrating the efficacy and understanding of the model towards text-to-SQL generation task.

References

[1] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, “Generalizing from a Few Examples: A Survey on Few-Shot Learning,” ACM Comput Surv, vol. 53, no. 3, Apr. 2019, doi: 10.1145/3386252.

[2] H. Li, J. Zhang, C. Li, and H. Chen, “RESDSQL: Decoupling Schema Linking and Skeleton Parsing for Text-to-SQL,” Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023, vol. 37, pp. 13067–13075, Feb. 2023, doi: 10.1609/aaai.v37i11.26535.

[3] T. Scholak, N. Schucher, and D. Bahdanau, “PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models,” EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings, pp. 9895–9901, Sep. 2021, doi: 10.18653/v1/2021.emnlp-main.779.

[4] J. Qi et al., “RASAT: Integrating Relational Structures into Pretrained Seq2Seq Model for Text-to-SQL,” Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, pp. 3215–3229, May 2022, doi: 10.18653/v1/2022.emnlp-main.211.

[5] T. B. Brown et al., “Language Models are Few-Shot Learners,” Adv Neural Inf Process Syst, vol. 2020-December, May 2020, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2005.14165v4

[6] A. Chowdhery et al., “PaLM: Scaling Language Modeling with Pathways,” Apr. 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2204.02311v5

[7] “ChatGPT.” Accessed: Jan. 25, 2024. [Online]. Available: https://chat.openai.com/chat

[8] OpenAI, “GPT-4 Technical Report”.

[9] Google, “PaLM 2 Technical Report”.

[10] J. Wei et al., “Emergent Abilities of Large Language Models,” Jun. 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2206.07682v2

[11] M. Chen et al., “Evaluating Large Language Models Trained on Code,” Jul. 2021, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2107.03374v2

[12] A. Liu, X. Hu, L. Wen, and P. S. Yu, “A comprehensive evaluation of ChatGPT’s zero-shot Text-to-SQL capability,” Mar. 2023, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2303.13547v1

[13] R. Sun et al., “SQL-PaLM: Improved Large Language Model Adaptation for Text-to-SQL,” May 2023, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2306.00739v3

[14] D. Zhou et al., “Least-to-Most Prompting Enables Complex Reasoning in Large Language Models,” May 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2205.10625v3

[15] M. Pourreza and D. Rafiei, “DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction,” Apr. 2023, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2304.11015v3

[16] J. Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” Adv Neural Inf Process Syst, vol. 35, Jan. 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2201.11903v6

[17] X. Wang et al., “Self-Consistency Improves Chain of Thought Reasoning in Language Models,” Mar.

2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2203.11171v4

[18] A. Srivastava et al., “Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models,” Jun. 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2206.04615v3

[19] T. Yu et al., “Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task,” Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018, pp. 3911–3921, Sep. 2018, doi: 10.18653/v1/d18-1425.

[20] Y. Gan et al., “Towards Robustness of Text-to-SQL Models against Synonym Substitution,” ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference, pp. 2505–2515, Jun. 2021, doi: 10.18653/v1/2021.acl-long.195.

[21] X. Deng, A. H. Awadallah, C. Meek, O. Polozov, H. Sun, and M. Richardson, “Structure-Grounded Pretraining for Text-to-SQL,” NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, pp. 1337–1350, Oct. 2020, doi: 10.18653/v1/2021.naacl-main.105.

[22] Y. Gan, X. Chen, and M. Purver, “Exploring Underexplored Limitations of Cross-Domain Text-to-SQL Generalization,” EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings, pp. 8926–8931, Sep. 2021, doi: 10.18653/v1/2021.emnlp-main.702.

[23] H. Touvron et al., “Llama 2: Open Foundation and Fine-Tuned Chat Models,” Jul. 2023, Accessed: Jan. 28, 2024. [Online]. Available: https://arxiv.org/abs/2307.09288v2

[24] B. Rozière et al., “Code Llama: Open Foundation Models for Code,” Aug. 2023, Accessed: Jan. 28, 2024. [Online]. Available: https://arxiv.org/abs/2308.12950v2

[25] N. Rajkumar, R. Li, and D. Bahdanau, “Evaluating the Text-to-SQL Capabilities of Large Language Models,” Mar. 2022, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2204.00498v1

[26] X. Chen, M. Lin, N. Schärli, and D. Zhou, “Teaching Large Language Models to Self-Debug,” Apr. 2023, Accessed: Jan. 25, 2024. [Online]. Available: https://arxiv.org/abs/2304.05128v2

[27] T. Yu et al., “GraPPa: Grammar-Augmented Pre-Training for Table Semantic Parsing,” ICLR 2021 - 9th International Conference on Learning Representations, Sep. 2020, Accessed: Jan. 27, 2024. [Online]. Available: https://arxiv.org/abs/2009.13845v2

[28] Y. Gan et al., “Natural SQL: Making SQL Easier to Infer from Natural Language Specifications,” Findings of the Association for Computational Linguistics, Findings of ACL: EMNLP 2021, pp. 2030–2042, Sep. 2021, doi: 10.18653/v1/2021.findings-emnlp.174.

[29] X. Deng, A. H. Awadallah, C. Meek, O. Polozov, H. Sun, and M. Richardson, “Structure-Grounded Pretraining for Text-to-SQL,” NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, pp. 1337–1350, 2021, doi: 10.18653/v1/2021.naacl-main.105.

[30] P. Xu et al., “Optimizing Deeper Transformers on Small Datasets”, Accessed: Jan. 27, 2024. [Online]. Available: https://github.com/BorealisAI/DT-Fixup

[31] 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,” ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference, pp. 2541–2555, Jun. 2021, doi: 10.18653/v1/2021.acl-long.198.

[32] B. Hui et al., “S^{2}SQL: Injecting Syntax to Question-Schema Interaction Graph Encoder for Text-to-SQL Parsers,” Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 1254–1262, Mar. 2022, doi: 10.18653/v1/2022.findings-acl.99.

[33] T. Scholak, N. Schucher, and D. Bahdanau, “PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models,” EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings, pp. 9895–9901, Sep. 2021, doi: 10.18653/v1/2021.emnlp-main.779.

[34] J. Qi et al., “RASAT: Integrating Relational Structures into Pretrained Seq2Seq Model for Text-to-SQL,” Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, pp. 3215–3229, May 2022, doi: 10.18653/v1/2022.emnlp-main.211.