Tackling prediction tasks in relational databases with LLMs

Anonymous ACL submission

Abstract

Despite Large Language Models (LLMs) demonstrating strong results on a range of NLP tasks, their application to predictive tasks in relational databases remains largely unexplored. They pose unique challenges due to their interconnected tables, complex relationships, and heterogeneous data types. RelBench benchmark offers realistic datasets for these tasks but has not yet been approached using LLMs. In this work, we show that the straightforward application of pre-trained LLMs can achieve competitive results in relational deep learning challenges. Our result establishes LLMs as a new, simple baseline for machine learning on relational databases and encourages further research in this direction.

1 Introduction

The application of Large Language Models (LLMs) and foundational models to relational databases remains largely uncharted territory. While the number of resources for tabular data and structured data is recently rapidly growing, with resources available for both unsupervised and supervised learning tasks (e.g., GitTables (Hulsebos et al., 2023), TabLib (Eggert et al., 2023), and evaluation datasets (see the list by Gardner et al. (2024)) – relational databases have not received comparable attention, particularly in the context of LLMs.

Relational databases are inherently more complex than single tables due to their collections of interconnected tables linked by primary and foreign keys, encompassing one-to-many and many-to-many relationships and heterogeneous data types. The number of machine learning benchmarks that involve such databases is limited. Notable examples include CTU Prague Relational Learning Repository (Motl and Schulte, 2024), SJTUTable (Li et al., 2024b), Text-to-SQL benchmarks such as Spider (Yu et al., 2018) and Bird (Li et al., 2024a), and RelBench (Relational Deep Learning Benchmark) (Robinson et al., 2024).

In this work, we focus on RelBench, a recently published, realistic and readily available benchmark for predictive tasks such as classification and regression on relational data. Previous work on RelBench has reported results using standard machine learning approaches (e.g. gradient boosting) and the relational deep learning method (Fey et al., 2023; Robinson et al., 2024). To date, no results using LLMs–trained in the standard manner or with tabular data–have been reported on RelBench. In this paper, we present a series of results obtained using LLMs on RelBench. We demonstrate that by traversing links between tables, one can construct information-rich documents that allow pre-trained LLMs to make predictions competitive to the relational deep learning approach. Our results establish a new simple baseline and motivate further research in this direction.

2 Relational data and RelBench

A table T is defined as a collection of n rows $T = \{R, \ldots, R_n\}$. All the rows in the same table have the same set of attributes, also called columns. Following the nomenclature used by Robinson et al. (2024), we also refer to a single row from a single table as an entity. With each column, we associate a data type, such as numerical, time, text, image, etc. A relational database can be represented as a collection of tables $\mathcal{T} = \{T_1, \ldots T_t\}$ and links between those tables $\mathcal{L} = \{L_1, \ldots, L_l\}$. A single link $L = (T_{\text{fkey}}, T_{\text{pkey}})$ consists of a foreign key column in T_{fkey} that points to a primary key column of T_{pkey}. The primary key is a unique identifier of a row in a table, while the foreign key is a reference to a primary key in another table. Furthermore, each row in a table can be associated with a timestamp t, which indicates the time the row was created. Tables that have a timestamp column are

called fact tables, while tables that do not have a timestamp and contain static information are called dimension tables (Garcia-Molina et al., 2008).

The Relational Deep Learning Benchmark (RelBench; Robinson et al. (2024)) is a collection of 7 relational databases from different domains, each with a set of predictive tasks, giving a total of 30 tasks. Each predictive task is defined by a task table, which contains foreign keys to the main tales in the database, the prediction time (also called the seed time), and the target variable y to be predicted. The task table is divided into train, validation, and test tables based on the prediction time t_p (the same entities may appear multiple times in train, validation, and test tables with different timestamps). The prediction time indicates at which time the target is to be predicted. It filters future data in the database to ensure the temporal consistency of the data and prevent information leaks. Formally, when predicting for an entity with prediction time t_p, the model can get as input only information from another entity v with timestamp $t_v < t_p$.

Currently, RelBench provides 3 types of predictive tasks: (1) entity-wise binary classification tasks where $x, y \in \{0, 1\}$ (e.g., churn prediction), evaluated using AUROC (McNeil and Hanley, 1984), (2) entity-wise regression tasks where $y \in \mathrm{R}$ (e.g., prediction of the total amount of sold items), evaluated using mean absolute error (MAE), and (3) tasks predicting a link between two entities (e.g., user item recommendation), evaluated using mean average precision (MAP). In summary, RelBench aims to represent typical prediction and forecasting tasks in real-world relational databases.

3 Straightforward application of LLMs

The challenge of the RelBench benchmark comes from the nature of relational data. Before applying classical machine learning models such as gradient boosted trees (GBT) (Friedman, 2001), one needs to "flatten" the complex relational structure into a single table that will serve as a representation of instances.

Applying naive denormalization to the task table that follows only links in the direction from the foreign key to the primary key and appending columns from the linked table only partially solves the problem. Much important information for the task is usually stored in the links in the opposite direction, from the primary key to the foreign key. However, following these one-to-many links requires the usage of some aggregation function, usually selected by human experts in the process of feature engineering.

To address the problem of mundane feature engineering in relational databases, Robinson et al. (2024) proposes using graph neural networks. They first use deep tabular models that encode each row's attributes into initial entity embedding, mainly ResNet tabular model (Gorishniy et al., 2021)) or textual embeddings such as GloVe or BERT. These initial node embeddings are then fed into a type of graph neural network (GNN) (Gilmer et al., 2017; Hamilton et al., 2017) iteratively updating the node embeddings based on their neighbors. In their work, they used two different architectures, the GraphSAGE model (Hamilton et al., 2017) and ID-GNN (You et al., 2021). Output node embeddings are fed into task-specific prediction heads and are learned end-to-end. Fey et al. (2023); Robinson et al. (2024) refer to this approach as relational deep learning (RDL).

In this work, we investigated the feasibility of applying a much simpler approach to solving the task. We propose representing the prediction problem as a text document and allowing a pre-trained large language model to predict the output. Our framework consists of two parts: 1) construction of text document and 2) task-aware inference.

3.1 Constructing documents for LLMs

Our approach starts with the construction of a document for each sample $\boldsymbol{x}$ in the test set. The document consists of three parts. First, we include the task context, that is, a short description of a relational database followed by a short description of the prediction task. We use the same descriptions as provided by Robinson et al. (2024, Section 4 and Appendix A). The next part consists of n_{inc}, a specified number of in-context examples (including the predicted values) that are entities from the train table. The in-context examples can be followed by the maximum n_{rel} latest related examples, which are entities from the train table linked to the same set of primary keys. For every in-context example v, we ensure that $t_v < t_p$. Finally, the entity for which the prediction needs to be made is added.

We apply a denormalization process to every entity added to the document, following links from a foreign key to a primary key. We gather links related to all entities joined in the process of denormalization. Following the links in the direction from a primary key to a foreign key, we select up

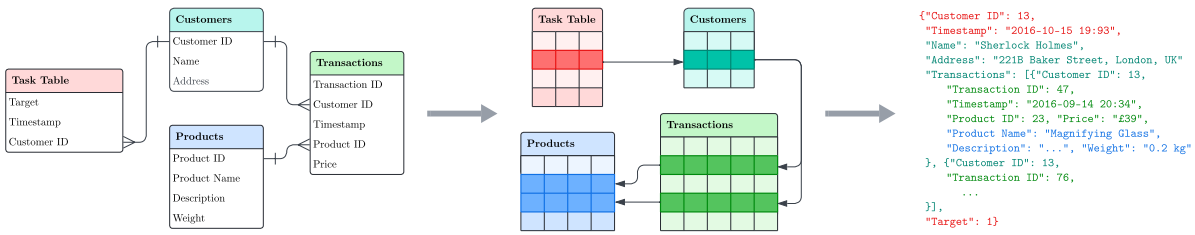

Figure 1: Process of constructing a single example for LLM-based inference.

n_{nest} entities from those tables. We recursively apply the same procedure to them in breath-first order, up to the recursion depth d, but skip the tables already visited. Note that this process can be performed very efficiently on the fly on the database if hash indexes on all primary and foreign keys are constructed. For binary classification, we use stratified sampling to obtain in-context examples to ensure both negative and positive samples. The set of in-context examples is always the same for every document.

As a result, we get an entity representation that includes nested entities from other tables. These entities are then serialized with JSON, as was demonstrated, that JSON performs well as the text representation of tabular data for LLMs Singha et al. (2023). Additionally, the JSON format allows for placing nested rows from other tables within the example. This reduces the requirement for LLM to perform multi-hop inference, which often negatively impacts performance (Wu et al., 2024a,b). The target variable is always listed as the last field in the serialized representation. The process of generating serialized examples is demonstrated in Figure 1. In Appendix B, we present the pseudocode of the whole procedure.

3.2 Metric aware inference

We find out that using a base version of LLM to simply fill out the tokens representing the target or asking for an instruct version for prediction is not working well for such generated documents.Therefore, following the work of Lukasik et al. (2024), we use base models and examine the probability distribution of the next tokens. Under the assumption that a base model is trained only with a next-token prediction task using cross-entropy loss, the minimizer under an unrestricted hypothesis class is the true conditional distribution $\mathbb{P}(y|\boldsymbol{x})$ (Gneiting and Raftery, 2007).

In RelBench, entity classification tasks are evaluated using the AUROC metric. It can be shown that the optimal decision rule for AUROC takes the form of any strictly monotone transformation of $\mathbb{P}(y = 1|\boldsymbol{x})$ (Clémençon et al., 2008; Uematsu and Lee, 2014). Because of that, we simply use the probability of the token representing a positive class ("1").

For the regression tasks, RelBench uses the mean absolute error for evaluation, for which the optimal prediction is the median of $\mathbb{P}(y|\boldsymbol{x})$ (Bishop and Nasrabadi, 2006). In the case of LLMs, the median can be calculated by sampling probabilities of values between minimum and maximum values seen in the train set.

3.3 Applying simple prediction head

Although using metric-aware inference leads to good AUROC results in binary classification tasks, a median prediction approach does not help in the case of regression tasks. To further test the strength of the representation produced by LLMs, instead of looking at the predicted distribution of tokens, we train a small multilayered perception (MLP) head with a single hidden layer of size 10 using a small subset of training documents of size n_{train} generated in the same way as described in Section 3.1.

4 Experimental results

In this section, we compare the results of our method with the results reported by Robinson et al. (2024), that is, with a simple baseline obtained using a LightGBM model (Ke et al., 2017) on naively normalized task tables and RDL approach. We use two recently released Llama 3.2 models in size of 1B and 3B parameters (Dubey et al., 2024). For each task, we generate documents for test-set examples using different combinations of in-context examples $n_{\text{inc}} \in \{0, 8, 16\}$, related examples $n_{\text{rel}} \in \{0, 8, 16\}$, nested rows from linked tables $n_{\text{nest}} \in \{0, 4, 8\}$, and depth of relation graph traversal $d \in \{0, 1\}$. For the variant with trained

Dataset	Task	LightGBM	RDL	Llama 3.2 1B	+ MLP	Llama 3.2 3B	+ MLP
rel-amazon	user-churn	52.22	**70.42**	60.56	*66.56*	62.55	65.00
	item-churn	62.54	**82.81**	71.96	*80.16*	73.41	63.49
rel-stack	user-engagement	63.39	**90.59**	81.01	*87.09*	81.23	79.80
	user-badge	63.43	**88.86**	71.13	*88.19*	79.99	87.60
rel-trial	study-outcome	*70.09*	68.60	55.72	68.38	59.17	**70.82**
rel-f1	driver-dnf	68.85	72.62	65.81	78.41	*80.03*	**82.33**
	driver-top3	73.93	75.54	*88.47*	87.36	87.11	**89.70**
rel-hm	user-churn	55.21	**69.88**	64.34	*68.72*	63.81	65.55
rel-event	user-repeat	68.04	**76.89**	76.38	*76.72*	70.11	61.01
	user-ignore	79.93	*81.62*	78.55	**84.02**	68.65	69.61
rel-avito	user-visits	53.05	**66.20**	60.28	64.98	53.10	*65.92*
	user-clicks	53.60	65.90	61.32	*71.31*	52.75	**74.71**

Table 1: Comparison of LLMs' best-achieved results (AUROC, higher is better) with RDL and GBT models on entity classification tasks in RelBench. The best values are in **bold**, second best in *italic*.

Dataset	Task	LightGBM	RDL	Llama 3.2 1B + MLP	Llama 3.2 3B + MLP
rel-amazon	user-ltv	16.783	**14.313**	*14.864*	14.887
	item-ltv	60.569	**50.053**	*52.682*	56.185
rel-stack	post-votes	*0.068*	**0.065**	0.090	0.095
rel-trial	study-adverse	**44.011**	*44.473*	51.845	48.383
	site-success	*0.425*	**0.400**	0.441	0.448
rel-f1	driver-position	4.170	4.022	*3.539*	**3.092**
rel-hm	item-sales	0.076	**0.056**	*0.057*	0.065
rel-event	user-attendance	*0.264*	**0.258**	0.293	0.308
rel-avito	ad-ctr	*0.041*	0.041	**0.036**	0.041

Table 2: Comparison of LLMs' best-achieved results (MAE, lower is better) with RDL and GBT models on entity regression tasks in RelBench. The best values are in **bold**, second best in *italic*.

MLP heads, we use either $n_{\text{train}} \in \{1\text{e}4, 1\text{e}5\}$.

We compare our approach with both baselines on all classification and regression tasks from RelBench, for which we present the results in Tables 1 and 2. In these tables, we report the result achieved by the best set of parameters, selected using the validation set. The detailed results for each parameter set are reported in Appendix D. In almost all cases, metric-aware LLM inference beats the LightGBM baseline on classification tasks. However, thanks to specialized MLP heads, the LLM beats or gets very close to the results of RLD using only a fraction of its resources. Additionally, we observe that, depending on the task, different information is important. Some tasks benefit mainly from a large number of n_{rel}, while others require information from linked tables (high values of d and n_{nest}). Rel-f1 is the only dataset where adding related examples from a train set decreases model performance, as the model seems to be mostly relying on its internal knowledge of Formula-1.

5 Conclusion

In this preliminary investigation, we showed that LLMs can be used successfully for prediction tasks in the environment of relational databases without the need for additional fine-tuning. We hypothesize that training models on tabular and relational data (similar to works of Tran et al. (2024); Li et al. (2024b)) may further enhance the performance of the proposed approach. Additionally, we demonstrated that different tasks may require including different information in the context. So far, we have selected the best combination of parameters via a naive search using a validation dataset; however, due to the limited context sizes of LLMs, more intelligent selection of information (e.g. which columns to include and links to follow) may improve predictive performance as well as reduce computational cost.

Limitations

We test our approach on a single, though arguably diverse, relational benchmark. For specific databases and tasks, the input window required by the in-context learning we propose here might be too long for many popular LLMs.

References

Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Stéphan Clémençon, Gábor Lugosi, and Nicolas Vayatis. 2008. Ranking and empirical minimization of u-statistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gus Eggert, Kevin Huo, Mike Biven, and Justin Waugh. 2023. TabLib: A dataset of 627m tables with context. *Preprint*, arXiv:2310.07875.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. 2023. Relational deep learning: Graph representation learning on relational databases. *arXiv preprint arXiv:2312.04615*.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. 2008. *Database Systems: The Complete Book*, 2 edition. Prentice Hall Press, USA.

Josh Gardner, Juan C Perdomo, and Ludwig Schmidt. 2024. Large scale transfer learning for tabular data via language modeling. *arXiv preprint arXiv:2406.12031*.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Tilmann Gneiting and Adrian E Raftery. 2007. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Madelon Hulsebos, Çagatay Demiralp, and Paul Groth. 2023. GitTables: A large-scale corpus of relational tables. *Proceedings of the ACM on Management of Data*, 1(1):1–17.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024a. 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.

Weichen Li, Xiaotong Huang, Jianwu Zheng, Zheng Wang, Chaokun Wang, Li Pan, and Jianhua Li. 2024b. rLLM: Relational table learning with LLMs. *arXiv preprint arXiv:2407.20157*.

Michal Lukasik, Harikrishna Narasimhan, Aditya Krishna Menon, Felix Yu, and Sanjiv Kumar. 2024. Metric-aware LLM inference. *arXiv preprint arXiv:2403.04182*.

Barbara J McNeil and James A Hanley. 1984. Statistical approaches to the analysis of receiver operating characteristic (ROC) curves. *Medical decision making*, 4(2):137–150.

Jan Motl and Oliver Schulte. 2024. The CTU Prague Relational Learning Repository. *Preprint*, arXiv:1511.03086.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan E Lenssen, Yiwen Yuan, Zecheng Zhang, et al. 2024. RelBench: A benchmark for deep learning on relational databases. *arXiv preprint arXiv:2407.20060*.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. 2023. Tabular representation, noisy operators, and impacts on table structure understanding tasks in LLMs. *arXiv preprint arXiv:2310.10358*.

Quan M Tran, Suong N Hoang, Lam M Nguyen, Dzung Phan, and Hoang Thanh Lam. 2024. TabularFM: An open framework for tabular foundational models. *arXiv preprint arXiv:2406.09837*.

Kazuki Uematsu and Yoonkyung Lee. 2014. Statistical optimality in multipartite ranking and ordinal regression. *IEEE transactions on pattern analysis and machine intelligence*, 37(5):1080–1094.

Jian Wu, Linyi Yang, Manabu Okumura, and Yue Zhang. 2024a. MRKE: The multi-hop reasoning evaluation of LLMs by knowledge edition. *arXiv preprint arXiv:2402.11924*.

Jian Wu, Linyi Yang, Zhen Wang, Manabu Okumura, and Yue Zhang. 2024b. Evaluating LLMs' inherent multi-hop reasoning ability.

Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, and Jure Leskovec. 2021. Identity-aware graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10737–10745.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. 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*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

A RelBench datasets statistics

In Tables 3 and 4, we report statistics of benchmarks used in this work.

B Formal algorithm

In Algorithm 1 we present the pseudocode of the procedure that constructs document representation D_X of an example X, as described in Section 3.1. Here, we assume that examples X behave as a dictionary, containing a set of objects identifiable by keys. We denote a set of visited tables as $\mathcal{V}$.

C Experimental details

All the experiments were conducted on a computational node with eight Nvidia H100 GPUs with 80GB of memory each. However, all the experiments can be reproduced using a single such GPU. All the computation was performed using bfloat16 precision.

We use the same MLP architecture for all prediction tasks, that is, with the input of size of LLM's single token embedding (2048 for Llama 3.2 1B, and 3072 for Llama 3.2 3B), one hidden layer of size 10 and an output. All MLP heads were trained using the Adam optimizer using an initial learning rate equal to $1\mathrm{e}{-4}$, which is linearly decreased for over 100 epochs, with the weight decay set to constant $1\mathrm{e}{-3}$. The MLP is never trained on samples larger than 1e5 examples from the train set. For validation purposes, we never use samples larger than 1e4 from the validation set. We will publish the code for replicating all the results.

D Extended results

In Tables 5 and 6, we present the detailed results for different combination of document parameters: $n_{\text{inc}} \in \{0, 8, 16\}$, related examples $n_{\text{rel}} \in \{0, 8, 16\}$, nested rows from linked tables $n_{\text{nest}} \in \{0, 4, 8\}$, and depth of relation graph traversal $d \in \{0, 1\}$ and Llama 3.2 1B model. For each set of document generation parameters, we conducted a single run. For the results of the Llama 3.2 3B model, we only tested selected combinations that turned out to be the most effective for the 1B variant.

Name	Domain	#Tables	#Rows	#Cols	Start	Val	Test
rel-amazon	E-commerce	3	15,000,713	15	2008-01-01	2015-10-01	2016-01-01
rel-avito	E-commerce	8	20,679,117	42	2015-04-25	2015-05-08	2015-05-14
rel-event	Social	5	41,328,337	128	1912-01-01	2012-11-21	2012-11-29
rel-f1	Sports	9	74,063	67	1950-05-13	2005-01-01	2010-01-01
rel-hm	E-commerce	3	16,664,809	37	2019-09-07	2020-09-07	2020-09-14
rel-stack	Social	7	4,247,264	52	2009-02-02	2020-10-01	2021-01-01
rel-trial	Medical	15	5,434,924	140	2000-01-01	2020-01-01	2021-01-01

Table 3: Statistics of RelBench datasets. Datasets vary significantly in the number of tables, total number of rows, and number of columns. In this table, we only count rows available for test inference, i.e., rows up to the test time cutoff.

Dataset	Task name	#Rows of task table			#Unique	%Train/Test
		Train	Validation	Test	Entities	Entity Overlap
rel-amazon	user-churn	4,732,555	409,792	351,885	1,585,983	88.0
	item-churn	2,559,264	177,689	166,482	416,352	93.9
	user-ltv	4,732,555	409,792	351,885	1,585,983	88.0
	item-ltv	2,707,679	166,978	178,384	427,537	93.7
rel-avito	ad-ctr	5,100	1,766	1,816	4,997	59.8
	user-clicks	59,454	21,183	47,996	66,449	44.3
	user-visits	86,619	29,979	36,129	13,405	64.6
rel-event	user-attendance	19,261	2,014	2,006	9,694	14.6
	user-repeat	3,842	268	246	1,154	11.5
	user-ignore	19,239	4,185	4,010	9,979	21.1
rel-f1	driver-dnf	11,411	566	702	821	50.0
	driver-top3	1,353	588	726	134	50.0
	driver-position	7,533	499	864	4,430	44.6
rel-hm	user-churn	3,871,410	76,556	74,575	1,002,984	89.7
	item-sales	5,488,184	105,542	105,542	1,005,542	100.0
rel-stack	user-engagement	1,360,850	85,838	88,137	88,137	97.4
	user-badge	3,386,276	247,398	255,360	255,360	98.9
	post-votes	2,453,921	156,216	160,903	160,903	97.1
rel-trial	study-outcome	11,994	960	825	13,729	0.0
	study-adverse	43,335	3,596	3,098	50,029	0.0
	site-success	276,474	12,687	12,927	129,642	9.9

Table 4: List of RelBench predictive tasks used in this work with a number of train, validation, and test examples.

Algorithm 1 GENERATE DOCUMENT$(X, T_{\text{train}}, \mathcal{T}, \mathcal{L}, n_{\text{inc}}, n_{\text{rel}}, d, n_{\text{nest}})$

Requires: data and task descriptions, test entity X, train task table dataset $T_{\text{train}} = [(X_i, y_i)]_{i=1}^{n_{\text{train}}}$, relation database $(\mathcal{T}, \mathcal{L})$, number of in-context examples n_{inc}, number of related examples n_{rel}, maximum depth of database traversal d, maximum number of nested rows n_{nest}

procedure ADD RELATED ENTITIES$(X, \mathcal{T}, \mathcal{L}, n_{\text{nest}}, d_{\text{cur}}, d_{\max}, \mathcal{V})$
  for $(T_{\text{fkey}}, T_{\text{pkey}}) \in \mathcal{L}$: fkey $\in X'$ **do**
    $X \leftarrow X \cup T_{\text{pkey}}$
    $\mathcal{V} \leftarrow \mathcal{V} \cup T_{\text{pkey}}$
  if $d = d_{\max}$ **then return** X
  for $(T_{\text{fkey}}, T_{\text{pkey}}) \in \mathcal{L}$: pkey $\in X'$ **do**
    for $i \in [n_{\text{nest}}]$ **do**
      $X' \leftarrow \arg\max_{(X'' \in T_{\text{fkey}} : t_{X''} < t_X \wedge X'' \notin X)} t_{X''}$
      $X' \leftarrow$ ADD RELATED ENTITIES$(X', \mathcal{T}, \mathcal{L}, n_{\text{nest}}, d_{\text{cur}}, d_{\max}, \mathcal{V})$
      $X \leftarrow X \cup X'$
    $\mathcal{V} \leftarrow \mathcal{V} \cup T_{\text{fkey}}$
  return X'

$D_X \leftarrow$ data description + task description
for $i \in [n_{\text{inc}}]$ **do**
  $(X', y') \leftarrow$ sample from T_{train} such that timestamp $t_{X'} < t_X$
  $X' \leftarrow$ ADD RELATED ENTITIES$(X', \mathcal{T}, \mathcal{L}, n_{\text{nest}}, 0, d, \emptyset)$
  $D_X \leftarrow D_X +$ JSON(X', y')
for $i \in [n_{\text{rel}}]$ **do**
  $(X', y') \leftarrow \arg\max_{((X'', y'') \in T_{\text{train}} : t_{X''} < t_X \wedge X'' \notin D_X)} t_{X''}$
  $X' \leftarrow$ ADD RELATED ENTITIES$(X', \mathcal{T}, \mathcal{L}, n_{\text{nest}}, 0, d, \emptyset)$
  $D_X \leftarrow D_X +$ JSON(X', y')
$D_X \leftarrow D_X +$ JSON(X)
return D_X

Dataset	Task	Llama 3.2 1B											
		$n_{\text{inc}}=8$ $n_{\text{rel}}=0$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=8$ $n_{\text{rel}}=0$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=0$ $n_{\text{rel}}=8$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=0$ $n_{\text{rel}}=8$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=8$ $n_{\text{rel}}=8$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=8$ $n_{\text{rel}}=8$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=16$ $n_{\text{rel}}=0$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=16$ $n_{\text{rel}}=0$ $d=1$ $n_{\text{rel}}=8$	$n_{\text{inc}}=0$ $n_{\text{rel}}=16$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=0$ $n_{\text{rel}}=16$ $d=1$ $n_{\text{rel}}=8$	$n_{\text{inc}}=16$ $n_{\text{rel}}=16$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=16$ $n_{\text{rel}}=16$ $d=1$ $n_{\text{rel}}=8$
rel-amazon	user-churn	52.32	50.24	50.20	52.16	57.98	59.40	53.37	56.44	51.06	55.04	**60.56**	*60.37*
	item-churn	50.90	54.06	63.94	58.45	**71.96**	70.18	51.12	51.09	64.50	56.27	*70.33*	70.09
rel-stack	user-engagement	63.54	*67.05*	**81.01**	-	-	-	-	-	-	-	-	-
	user-badge	52.45	*67.58*	**71.13**	-	-	-	-	-	-	-	-	-
rel-trial	study-outcome	53.17	52.85	*53.75*	50.16	53.17	52.85	51.82	**55.72**	53.75	51.21	51.82	52.32
rel-f1	driver-dnf	51.12	52.98	*64.95*	57.99	59.24	53.29	55.84	52.24	**65.81**	60.22	62.04	63.55
	driver-top3	**88.47**	72.78	62.65	72.37	77.40	69.82	*78.65*	71.80	66.01	68.93	57.79	50.56
rel-hm	user-churn	54.53	54.86	58.51	57.09	*62.12*	52.85	54.07	55.36	59.99	57.61	**64.34**	58.97
rel-event	user-repeat	67.43	60.02	67.43	65.31	**76.38**	69.98	56.96	53.80	66.97	69.93	74.19	*74.58*
	user-ignore	66.46	*71.09*	57.19	54.73	66.16	**78.55**	61.44	69.37	57.43	51.93	64.74	63.06
rel-avito	user-visits	50.42	**60.28**	50.45	60.09	50.43	*60.22*	50.38	59.64	50.55	59.79	50.49	59.59
	user-clicks	50.89	*60.34*	52.19	**61.32**	51.75	59.44	51.90	58.33	51.35	58.84	52.30	59.12
Dataset	Task	Llama 3.2 1B + MLP											
		$n_{\text{inc}}=8$ $n_{\text{rel}}=0$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=8$ $n_{\text{rel}}=0$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=0$ $n_{\text{rel}}=8$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=0$ $n_{\text{rel}}=8$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=8$ $n_{\text{rel}}=8$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=8$ $n_{\text{rel}}=8$ $d=1$ $n_{\text{rel}}=4$	$n_{\text{inc}}=16$ $n_{\text{rel}}=0$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=16$ $n_{\text{rel}}=0$ $d=1$ $n_{\text{rel}}=8$	$n_{\text{inc}}=0$ $n_{\text{rel}}=16$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=0$ $n_{\text{rel}}=16$ $d=1$ $n_{\text{rel}}=8$	$n_{\text{inc}}=16$ $n_{\text{rel}}=16$ $d=0$ $n_{\text{rel}}=0$	$n_{\text{inc}}=16$ $n_{\text{rel}}=16$ $d=1$ $n_{\text{rel}}=8$
rel-amazon	user-churn	56.88	45.77	*66.46*	65.28	65.85	59.67	56.75	51.21	**66.56**	65.71	66.43	66.40
	item-churn	61.49	58.75	78.78	77.97	78.58	76.42	62.26	56.51	78.92	**80.16**	*78.93*	75.62
rel-stack	user-engagement	65.44	70.17	85.90	83.95	85.95	82.97	66.91	82.96	**87.09**	*86.98*	86.86	79.22
	user-badge	61.86	85.45	80.87	87.07	80.79	87.19	62.27	*87.47*	84.17	**88.19**	83.53	82.13
rel-trial	study-outcome	*68.16*	66.47	65.55	64.49	67.58	66.47	67.91	66.86	64.76	63.10	**68.38**	67.18
rel-f1	driver-dnf	*77.14*	51.78	70.66	65.89	74.13	48.88	**78.41**	54.60	70.52	67.13	72.69	49.55
	driver-top3	**87.36**	71.27	65.36	74.32	68.78	42.90	*83.46*	40.41	65.43	63.35	69.65	52.43
rel-hm	user-churn	55.11	51.64	68.45	64.08	68.16	64.47	56.03	48.32	**68.72**	66.74	*68.62*	63.33
rel-event	user-repeat	62.22	54.20	**76.72**	71.91	76.49	73.28	65.11	53.40	75.13	*76.56*	74.62	50.58
	user-ignore	68.77	64.16	*83.51*	81.43	**84.02**	80.22	68.53	62.52	83.50	80.11	83.02	77.15
rel-avito	user-visits	54.51	62.84	57.23	63.39	57.28	63.27	54.69	62.79	56.18	*64.18*	56.62	**64.98**
	user-clicks	60.34	65.57	66.91	65.73	66.16	70.03	58.68	68.11	67.92	*70.07*	67.34	**71.31**

Table 5: Comparison of Llama 3.2 1B results (AUROC, higher is better) on documents generated using different parameters. The best values are in **bold**, second best in *italic*.

Dataset	Task	Llama 3.2 1B + MLP											
		$n_{\text{inc}} = 8$ $n_{\text{rel}} = 0$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 8$ $n_{\text{rel}} = 0$ $d = 1$ $n_{\text{rel}} = 4$	$n_{\text{inc}} = 0$ $n_{\text{rel}} = 8$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 0$ $n_{\text{rel}} = 8$ $d = 1$ $n_{\text{rel}} = 4$	$n_{\text{inc}} = 8$ $n_{\text{rel}} = 8$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 8$ $n_{\text{rel}} = 8$ $d = 1$ $n_{\text{rel}} = 4$	$n_{\text{inc}} = 16$ $n_{\text{rel}} = 0$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 16$ $n_{\text{rel}} = 0$ $d = 1$ $n_{\text{rel}} = 8$	$n_{\text{inc}} = 0$ $n_{\text{rel}} = 16$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 0$ $n_{\text{rel}} = 16$ $d = 1$ $n_{\text{rel}} = 8$	$n_{\text{inc}} = 16$ $n_{\text{rel}} = 16$ $d = 0$ $n_{\text{rel}} = 0$	$n_{\text{inc}} = 16$ $n_{\text{rel}} = 16$ $d = 1$ $n_{\text{rel}} = 8$
rel-amazon	user-ltv	16.737	16.742	**14.864**	16.635	*15.019*	16.613	16.713	16.743	15.025	16.254	15.022	16.138
	item-ltv	61.705	62.095	*53.450*	54.790	53.597	58.282	61.528	61.847	53.939	56.666	**52.682**	55.370
rel-stack	post-votes	0.095	0.093	0.099	0.092	0.093	**0.090**	*0.091*	0.104	0.099	0.110	0.108	0.114
rel-trial	study-adverse	53.749	53.116	**52.281**	*53.066*	54.406	53.848	53.168	53.425	53.838	53.569	54.884	53.324
	site-success	0.450	0.450	0.444	0.446	0.449	0.446	0.445	0.447	0.446	*0.442*	**0.441**	0.449
rel-f1	driver-position	**3.539**	4.271	4.235	4.369	4.460	4.002	*3.902*	4.400	4.400	4.379	3.948	4.360
rel-hm	item-sales	0.095	0.097	0.070	0.070	0.073	**0.057**	0.098	0.099	0.062	*0.062*	0.064	0.082
rel-event	user-attendance	0.410	0.311	0.309	0.305	**0.293**	0.315	0.369	0.313	0.332	*0.304*	0.334	0.317
rel-avito	ad-ctr	0.044	0.044	0.043	0.043	0.043	**0.036**	0.045	0.044	0.044	*0.043*	0.043	0.043

Table 6: Comparison of Llama 3.2 1B results (MAE, lower is better) on documents generated using different parameters. The best values are in **bold**, second best in *italic*.