A Pre-training Framework for Relational Data with Information-theoretic Principles

Quang Truong[1] **, Zhikai Chen**[1] **, Mingxuan Ju**[2] **, Tong Zhao**[2] **, Neil Shah**[2] **, Jiliang Tang**[1]
[1]Michigan State University, [2]Snap Inc.
{truongc4, chenzh85,tangjili}@msu.edu
{mju, tong, nshah}@snap.com

Abstract

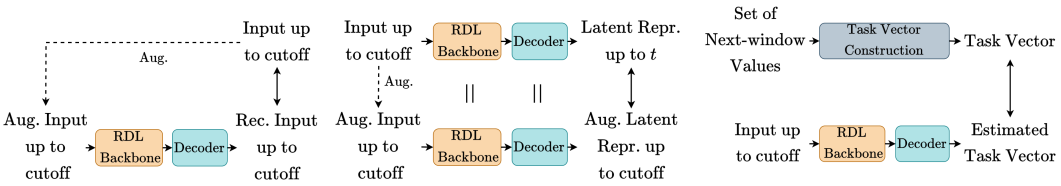
Relational databases underpin critical infrastructure across a wide range of domains, yet the design of generalizable pre-training strategies for learning from relational databases remains an open challenge due to task heterogeneity. Specifically, there exist many possible downstream tasks, as tasks are defined based on relational schema graphs, temporal dependencies, and SQL-defined label logics. An effective pre-training framework is desired to take these factors into account in order to obtain task-aware representations. By incorporating knowledge of the underlying distribution that drives label generation, downstream tasks can benefit from relevant side-channel information. To bridge this gap, we introduce Task Vector Estimation (TVE), a novel pre-training framework that constructs predictive supervisory signals via set-based aggregation over schema traversal graphs, explicitly modeling next-window relational dynamics. We formalize our approach through an information-theoretic lens, demonstrating that task-informed representations retain more relevant signals than those obtained without task priors. Extensive experiments on the RelBench benchmark show that TVE consistently outperforms traditional pre-training baselines. Our findings advocate for pre-training objectives that encode task heterogeneity and temporal structure as design principles for predictive modeling on relational databases. Our code is publicly available at `https://github.com/quang-truong/task-vector-estimation`.

1 Introduction

Relational databases (RDBs) have long served as the de facto standard for managing data storage across diverse industries [4, 5, 28], and learning tasks over RDBs depend on domain expertise for manually engineering meaningful features [24, 34]. While manual feature engineering can yield effective models, it demands significant human time and expertise, and it inherently introduces bias, susceptibility to human errors, and the risk of overlooking valuable predictive signals. Relational Deep Learning (RDL) [21, 24] has emerged as a powerful paradigm that exploits relational structures—defined by primary-foreign key (PK-FK) links across tables. By treating RDBs as graphs of tables, we can leverage existing state-of-the-art tabular feature encoders [17, 27] and Graph Neural Networks (GNNs) [26, 29, 35] to extract meaningful embeddings in an end-to-end manner. This integrated approach has already demonstrated significant improvements in efficiency over conventional manual feature engineering techniques due to the end-to-end learning pipeline [24, 46].

Designing pre-training objectives for RDL is particularly challenging. A single RDB typically supports multiple tasks, each definable by distinct SQL queries. Meanwhile, due to potential differences in label semantics, it is hard to guarantee that the latent representation obtained during the pre-training stage is relevant to downstream tasks, which leads to unreliable pre-trained models. Moreover, downstream tasks depend on temporal dynamics (e.g. next-window properties), which cannot be simply modeled by just pre-training on unlabeled input data alone. A pre-training framework is desired to model the dynamics in the latent representation intrinsically without having to rely

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Masked Autoencoder (b) Contrastive Learning (c) Task Vector Estimation

Figure 1: Overview of pre-training frameworks. Input graph is omitted for brevity.

on downstream fine-tuning for predictive modeling. In addition, what makes RDL distinct is the dependency of tasks on relational schema graphs, which leads to growing complexity on complicated RDBs. Recently, Liu et al. [40] showed that adapting traditional SSL methods [18, 30] to RDBs can inadvertently capture spurious yet seemingly informative patterns—so-called "interesting-looking noises"—that undermine downstream performance. This contradicts the core objective of traditional SSL, which aims to retain task-relevant information while discarding irrelevant noise [2, 6, 22, 33, 37], based on the assumption that different augmented views still share the same labels. Accordingly, our goal is to design a pre-training objective for RDL that avoids these pitfalls.

Main Contributions Our main contributions are summarized as follows:

1. **Task Vector Estimation (TVE)**: a pre-training framework for RDL that leverages schema graphs, next-window dynamics, and task heterogeneity to produce temporal, task-aware representations.
2. **Theoretical insights**: we prove that pre-training on full relational databases with our objectives preserves more downstream-relevant information than standard self-supervised methods.
3. **Empirical validation**: across diverse downstream tasks—especially in low-data regimes—TVE outperforms traditional SSL methods; ablation studies further uncover "flaky" task-dependent behaviors, underscoring the need to model both temporal dynamics and task heterogeneity.

2 Preliminaries

Relational database [24] RDB $(\mathcal{T}, \mathcal{L})$ is formally defined as a set of tables (or entity types) $\mathcal{T} = \{T_1, \ldots, T_n\}$ that are connected by a set of PK-FK relations (or edge types) $\mathcal{L} \subseteq \mathcal{T} \times \mathcal{T}$ [24]. Each table T_i can contain different data types for each column: numerical, categorical, text embeddings, etc. Additionally, each table contains a key column, which is either for primary keys or foreign keys, serving as links between tables. Tables are either *fact* or *dimension* tables [24]. Fact tables contain interaction between entities (e.g. T_{review}) with corresponding timestamps, while dimension tables contain static, immutable properties of entities (e.g. T_{customer} or T_{product}). Additionally, we refer root table to the table we would like to perform predictive modeling.

Graph perspectives of RDBs. RDB can be understood from two different perspectives, namely schema graph and relational entity graph. **Schema graph** $(\mathcal{T}, \mathcal{R})$ is a blueprint of the RDB, in which the relation is bi-directional $\mathcal{R} = \mathcal{L} \cup \mathcal{L}^{-1}$ [24]. **Relational entity graph** is the input graph for learning, and is a fine-grained view of the schema graph: we treat rows in a table as nodes, and they are connected to rows in other tables via the PK-FK links [24]. We use "nodes" and "rows" interchangeably in this work. We are interested in the relational entity graph snapshot up to a cutoff timestamp t:

$$\mathcal{G}^{(-\infty,t]} = (\mathcal{V}^{(-\infty,t]}, \mathcal{E}^{(-\infty,t]}, \varphi_{\mathcal{V}}, \varphi_{\mathcal{E}}, f_{\mathcal{V}}, f_{\mathcal{E}}), \tag{1}$$

where $\mathcal{V}^{(-\infty,t]}$ and $\mathcal{E}^{(-\infty,t]}$ are the sets of nodes and edges observed up to timestamp t, $\varphi_{\mathcal{V}} : \mathcal{V} \to \mathcal{T}$ maps each node to an entity type, $\varphi_{\mathcal{E}} : \mathcal{E} \to \mathcal{R}$ maps each edge to a PK-FK link type, and $f_{\mathcal{V}} : \mathcal{V} \to \mathbb{R}^d$ and $f_{\mathcal{E}} : \mathcal{E} \to \mathbb{R}^d$ map nodes and edges to their feature vectors, respectively.

Different tables have varying attribute sets, so node feature representations are inherently heterogeneous. Although edges could carry features, we use exclusively node features to simplify discourse.

Predictive tasks over RDBs. RDL spans a range of tasks, from table cell prediction [51] to next-window property forecasting [46]. In this study, we concentrate on predictive modeling over relational

data: given a training table T_{train}, the objective is to forecast future node-level properties as formalized by RelBench [46]. Each training node is associated with a labeled entity $v = (\mathcal{K}_v, t_v, y_v^{(t_v, t_v+\Delta t]})$, where $\mathcal{K}_v$ is a foreign key linking the entity to its originating table row, t_v is the cutoff timestamp, and $y_v^{(t_v, t_v+\Delta t]}$ is the next-window label. The learning objective is to estimate:

$$\hat{y}_v^{(t_v, t_v+\Delta t]} = f_\theta(f_{\mathcal{V}}(v), \mathcal{G}^{(-\infty, t_v]}, t_v), \tag{2}$$

where f_θ is a differentiable function parameterized by θ that uses the node's tabular features, the graph structure, and temporal context up to t_v to predict the future label.

Next, we would like to Fig. 2's schema graph to demonstrate the process of forecasting customers' review counts in the seven days following time t as an example. First, we extract $\mathcal{G}^{(-\infty, t]}$ for learning. Second, we label each node by issuing an SQL query that joins the root table (e.g. T_{customer}) with with a fact table (e.g. T_{review}) and aggregates event counts over the interval $(t, t + 7 \text{ days}]$. The node ID, the cutoff timestamp, and the corresponding label are recorded in the training table T_{train}. This join-and-aggregate step is necessary because the root table is a dimension table, thus relying on neighboring fact tables to provide temporal contexts. Third, follow [24, 46], we train a model to predict these labels. T_{train} is used only for lookup and loss computation, not for message passing.

RDL is inherently distinct from both static heterogeneous and temporal graph problems. Static heterogeneous graph methods [52, 60] omit temporal dynamics, while temporal graph approaches [16, 38] represent data as discrete-time snapshots or continuous-time event streams to capture graph evolution. By contrast, RDL treats timestamps as tabular features and uses them to impose causal constraints when extracting subgraphs. Lastly, RDL generates labels via SQL-driven schema-graph joins, leveraging fact tables, aggregates, and multi-hop joins absent from standard graph problems.

3 An Overview of the Proposed Pre-training Framework

We would like introduce an overview of our pre-training framework, namely Task Vector Estimation (TVE), as shown in Fig. 1c. Besides the RDL backbone to extract features and the decoder to map the extracted latent representation to the final task vector, our framework contains an external module called Task Vector Construction to construct an objective vector extracted from the set of next-window values. By depending on temporal dynamics, schema graph, and simple SQL logics [19], the vector represents a proxy representation for the set of next-window values—which plays an important roles in label generation process (see Section 4.1). Details are discussed in Section 4, where we outline the motivation, implementation, and theoretical justifications.

Most popular SSL frameworks, namely Masked Autoencoder (MAE) [30] and Contrastive Learning (CL) [18], are shown in Fig. 1a and Fig. 1b, respectively. MAE's objective is to reconstruct original feature values up to timestamp cutoff given augmentations, while CL attempts to minimize the distance between latent representations of two views of the same features (e.g. original features vs. corrupted features). Notably, the roles of decoders are different for MAE and CL. The decoder in MAE maps the extracted latent representation back to the original input space for reconstruction, whereas the decoder (often called the projection head) in CL maps the latent representation to a different, usually smaller space for contrastive learning. Both these methods depend on separation of data from inputs without modeling the next-window dynamics, which are critical for predictive modeling. Meanwhile, what sets TVE apart is the incorporation of an objective conditioned on next-window dynamics. TVE estimates task-relevant information during pre-training, and can be complementary to both MAE and CL (see Eq. 5 for the combined objective and Table 2 for experimental results). Our objective correlates input features and next-window dynamics, thus yielding a task-aware latent representation.

4 Predictive Task-aware Self-supervised Learning

In this section, we would like to shed light on the downstream label generation process, then propose our SSL method, namely **Task Vector Estimation (TVE)**, based on the label generation insights. Finally, we establish the benefits of this SSL objective over other methods from an information-theoretic perspective.

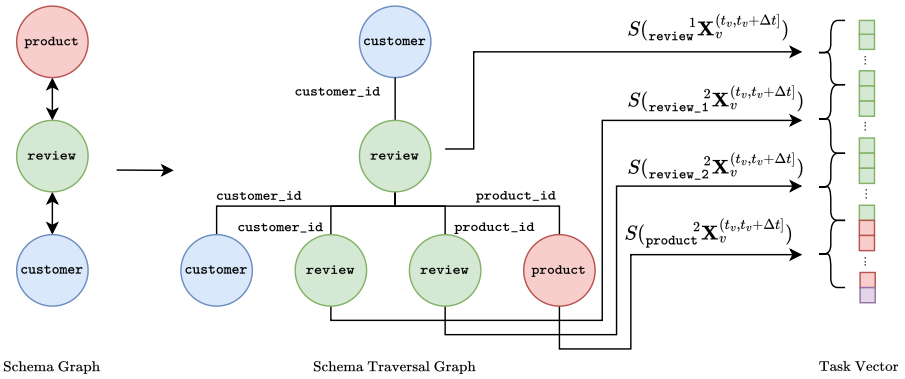


Figure 2: Construction of task vectors from relational schema. From schema graph, we derive a schema traversal graph, where edges represent valid join keys. For each reachable table R in k-hop from the source node v, we apply a set function S to the set of next-window values ${}_{R}^{k}\mathbf{X}_v^{(t_v,t_v+\Delta t]}$ reachable via the traversal path, where only paths contributing to the prediction task are retained. The task vector concatenates outputs from these set functions. A null indicator (purple cell) is activated when all other entries are absent (i.e., zero-valued).

4.1 Label Generation Process as a Set Function

Given the nature of labels defined in the training table introduced in Section 2, we make the following observations regarding label generation process. First, the training process is entity-centric—where we aim to perform predictive modeling for entities in the root table. Second, downstream tasks span across multiple tables in the RDB, and are conditioned on a set of nodes connected to the root table. Third, because the root table is often a dimension table, assigning a cutoff timestamp t_v requires joining it with fact tables. For example, tasks related to the `customer` table are timestamped via interactions with the `review` table, which is one hop away in the schema graph as shown in Fig. 2.

The set of next-window values depends not only on temporality, but also the schema graph structure and SQL logics. For example, predicting the number of reviews a customer will submit in the next window requires a one-hop join between the `customer` and `review` tables, with the `COUNT` aggregation. Likewise, computing the average price of products reviewed by a customer in that same window involves a two-hop join spanning `customer`, `review`, and `product`, coupled with the `AVERAGE` aggregation. For the sake of simplicity, assume we only consider the k-hop neighbors with the same entity-type, given an entity v and the timestamp t, its label $y_v^{(t)}$ is the output of a set function l defined over the features of its set of k-hop neighbors $\mathbf{X}_v^{(t+1)}$ in the next timestamp $t+1$:

$$y_v^{(t)} = l(\mathbf{X}_v^{(t+1)}), \quad \mathbf{X}_v^{(t+1)} = \left\{ f_{\mathcal{V}}(u) \mid u \in \mathcal{N}_k^{(t+1)}(v) \right\}. \tag{3}$$

The formal definition of Eq. 3, where we additionally differentiate sets of neighbors conditioned based on their entity type R, is included in Appendix B. Importantly, the traversal graph permits self-joins, e.g., joining entries in `review` by both `customer_id` and `product_id`. When the labeling function l is an identity function and sets contain a single element, Eq. 3 reduces to standard cell prediction, as often studied in Tabular Deep Learning [9, 27, 47, 49, 55].

4.2 Task Vector Estimation

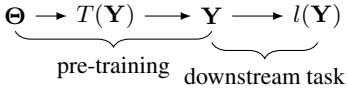


Figure 3: Markov Chain of Labels.

As demonstrated in Section 4.1, there are three key factors governing the label generation process: schema graphs, temporal dependencies, and SQL logics. While these three factors can complicate possible tasks, all tasks depend on ${}_{R}^{k}\mathbf{X}_v^{(t_v,t_v+\Delta t]}$, which are represented by all possible rows, possibly with different entity types R, that are k-hop from the root tables and associated with the next-window period $(t_v, t_v + \Delta t]$.

TVE's goal is to model task heterogeneity via estimating statistics representing ${}_{R}^{k}\mathbf{X}_v^{(t_v,t_v+\Delta t]}$. Generally, let $\mathbf{Y}$ represent a next-window set of values (e.g. ${}_{R}^{k}\mathbf{X}_v^{(t_v,t_v+\Delta t]}$). Our goal is to capture the

distributional properties of $\mathbf{Y}$, parameterized by $\mathbf{\Theta}$, as illustrated in Fig. 3. Here, $T(\mathbf{Y})$ denotes a sufficient statistic of $\mathbf{Y}$ for $\mathbf{\Theta}$. If the latent representation fully captures $T(\mathbf{Y})$ during pre-training, then the latent representation has all information regarding $\mathbf{Y}$ for $\mathbf{\Theta}$ [20]. Finally, the downstream labels $l(\mathbf{Y})$ depend on $\mathbf{Y}$ as illustrated in Fig. 3 and formally defined in Section 4.1. Thus, pre-training model to estimate $T(\mathbf{Y})$ yields a task-aware knowledge for downstream tasks.

TVE consists of two steps:

1. Generate a pre-training table by performing a `CROSS JOIN` between entity rows and candidate timestamps, then filtering out timestamps prior to the entity's first fact entry (e.g., first review).
2. Attach pretext labels by computing normalized set-level statistics S (mean, count, etc.) per column across all reachable ${}^{k}_{R}\mathbf{X}_{v}^{(t_v, t_v+\Delta t]}$, using SQL aggregation.

Regarding the first step, the intuition is that non-events entries are also important for downstream tasks such as churn prediction. The second step is illustrated in Fig. 2. Specifically, given a relational schema graph, we construct a schema traversal graph based on joinability between two tables via PK or FK, in which edges denote which key is used for joining. Paths in this graph possess semantic meanings. Given a customer, we can find the set of next-window reviews made by the customer $S({}_{\texttt{review}}{}^{1}\mathbf{X}_{v}^{(t_v, t_v+\Delta t]})$, the set of next-window products $S({}_{\texttt{product}}{}^{2}\mathbf{X}_{v}^{(t_v, t_v+\Delta t]})$, the set of reviews for all products made by the customer $S({}_{\texttt{review\_1}}{}^{2}\mathbf{X}_{v}^{(t_v, t_v+\Delta t]})$, and the set of reviews made by other customers for next-window products that the customer reviewed $S({}_{\texttt{review\_2}}{}^{2}\mathbf{X}_{v}^{(t_v, t_v+\Delta t]})$. Importantly, the extracted set must be causal, which means that the joined entries in the next hop must have the timestamp less than or equal to that of the entries in the previous hop. Then, we aggregate values in sets by simple SQL aggregators to get predictive pretext tasks. Note that we adopt the simple statistic $S(\mathbf{Y})$ for computational efficiency, while approximating $T(\mathbf{Y})$ is challenging. For further discussion regarding the sufficiency of simple statistics and future works on designing better approximations of sufficient statistics, please kindly refer to Appendix D. Our proposed task vector construction operates similarly to Breadth First Search, and tasks extracted from a schema path is concatenated to the final task vector, leading to a variable-length task vector depending on the traversal paths. The task vector summarizes statistics for every column of tables reachable within k-hops of the root table in the schema traversal graph, estimating this vector ensures that no column is prematurely excluded during pre-training. It is an important design choice because downstream tasks may depend on any columns of any table. Optionally, the vector can be compressed via dimensionality reduction techniques such as PCA [8] or MCA [1]. The pre-training loss is the Scaled Cosine Error [31]:

$$\mathcal{L}_{\text{SCE}} = \frac{1}{|T_{\text{train}}|} \sum_{v \in T_{\text{train}}} \left(1 - \frac{\hat{y}_v^T y_v}{||\hat{y}_v|| \cdot ||y_v||}\right)^{\alpha}, \quad \alpha \geq 1, \tag{4}$$

where y_v is the ground-truth task vector, $\hat{y}_v$ is the estimation of y_v, and α is a scaling factor. Additionally, to account for the potential imbalance due to null task vectors outnumbering others—we reweight each group's contribution to the loss inversely proportional to its relative frequency.

Optionally, to mitigate the potential loss of input-specific information during pre-training, we extend our objective beyond the TVE loss (Eq. 4) by combining a traditional self-supervised loss:

$$\mathcal{L} = \mathcal{L}_{\text{TVE}} + \beta\, \mathcal{L}_{\text{SSL}}, \tag{5}$$

where β is a balancing coefficient, and $\mathcal{L}_{\text{SSL}}$ denotes a standard self-supervised loss—either a reconstruction objective or a contrastive loss. This formulation encourages the model to retain both predictive information (captured via task vectors) and general input features.

4.3 Theoretical Evidence of Task Vector Estimation

In this section, we would like to characterize relationship between the latent representation of a pre-trained model and downstream tasks, and then we justify our design choice which can be proved to contain more task-relevant information than vanilla SSL methods.

Task heterogeneity introduces a unique challenge: how to design a predictive task-aware SSL method for pre-training so that the model can generalize across a combinatorially large set of potential downstream tasks? Each task is conditioned on different column subsets of ${}^{k}_{R}\mathbf{X}_{v}^{(t_v,t_v+\Delta t]}$, and these tasks may or may not be correlated. This raises a natural question: how much task-relevant information must a latent representation encode to fully capture the entire set of tasks?

Remark 4.1. *Suppose we have a set of tasks whose labels are denoted by* $\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_n}$*. The lower bound of latent coding length* L *is:*

$$L \geq H(\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_n}) = \left[\sum_{i=1}^{n} H(\mathbf{y_i})\right] - TC(\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_n}), \tag{6}$$

where $H(\mathbf{y}_i)$ *is the entropy of* $\mathbf{y_i}$ *and* $TC(\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_n})$ *is total correlation of all tasks. Additionally,* $TC(\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_n}) \geq 0$ *effectively becomes zero once all tasks are independent.*

The lower-bound of code length is covered in [20], and the relationship between joint entropy and total correlation is studied in [54]. Remark 4.1 underscores an important property of any generalizable latent representation: the more uncorrelated tasks to support, the longer the required latent code.

Some tasks may degrade model performance. For example, if spending behavior is independent of gender, then training on gender-conditioned spending prediction tasks introduces noise. By the Data Processing Inequality, information can only decrease or remain the same through processing:

Remark 4.2. *Given the Markov chain* $\mathbf{y} \to \mathbf{x} \to \mathbf{x}'$ *where* $\mathbf{y}$ *is label,* $\mathbf{x}$ *is original data, and* $\mathbf{x}'$ *is the filtered data, then* $I(\mathbf{y}; \mathbf{x}) \geq I(\mathbf{y}; \mathbf{x}')$ *by Data Processing Inequality.*

Next, we provide an information-theoretic justification for TVE. Specifically, we would like to characterize how guiding models toward a predictive task-aware objective can benefit downstream tasks compared with vanilla contrastive separation or mask reconstruction from input data only.

Let $\mathbf{x}$ denote input features, $\mathbf{t}$ be a random variable representing side-channel information (which includes task vector proposed in Section 4.2), and $\mathbf{y}$ denote downstream labels. Let $\mathbf{z}_1$ be a latent representation learned from both $\mathbf{x}$ and $\mathbf{t}$, and $\mathbf{z}_2$ be learned from $\mathbf{x}$ alone.

Prior literature [22] defines sufficiency of representation $\mathbf{z_2}$ of $\mathbf{x}$ for $\mathbf{y}$, in which $\mathbf{z_2}$ must be as predictive as $\mathbf{x}$ to predict $\mathbf{y}$. Similarly, we can define a condition for a sufficient representation of inputs and side channels for a downstream task.

Definition 4.1 (Sufficient representation given inputs and side-channel information)**.** *A representation* $\mathbf{z_1}$ *is sufficient for predicting* $\mathbf{y}$ *given input features* $\mathbf{x}$ *with side-channel information* $\mathbf{t}$ *if and only if* $I(\mathbf{xt}; \mathbf{y} \mid \mathbf{z_1}) = 0$.

We can prove that a representation of $\mathbf{x}$ with additional knowledge of $\mathbf{t}$ shares as least as much information with the downstream task than that of $\mathbf{x}$ without $\mathbf{t}$ does (see proof in Appendix A.1).

Theorem 4.1. *Let* $\mathbf{x}, \mathbf{t}, \mathbf{y}$ *be random variables with joint distribution* $p(\mathbf{x}, \mathbf{t}, \mathbf{y})$*. Assume that* $\mathbf{z_1}$ *is sufficient representation of* $\mathbf{x}$ *with side-channel information* $\mathbf{t}$ *for* $\mathbf{y}$*, and* $\mathbf{z_2}$ *is sufficient representation of* $\mathbf{x}$ *for* $\mathbf{y}$*. Then mutual information between* $\mathbf{z_1}$ *and* $\mathbf{y}$ *is at least as much as that between* $\mathbf{z_2}$ *and* $\mathbf{y}$*:*

$$I(\mathbf{z_1}; \mathbf{y}) \geq I(\mathbf{z_2}; \mathbf{y}).$$

The theorem justifies the construction of the task vector, introduced in Section 4.2. In other words, whenever side-channel signals actually carry extra task-relevant cues, using them in pre-training can equip the model with task-aware representations. The task vector is task-relevant, as it serves as the proxy representation of a set of values that are used during label generation process (as demonstrated in Eq. 3), thus improving fine-tuning. In contrast, standard SSL learns noise-invariant embeddings and only later fits a direct input–label mapping in the fine-tuning stage, without ever modeling the underlying set distribution in the pre-training stage.

5 Experiments

In this section, we conduct comprehensive experiments to answer the following research questions:

- RQ1: How does our proposed task vector improve model performance on downstream tasks?

Table 1: Performance across datasets and model variants in the low-data regime. Reported metrics in are ROC-AUC for classification tasks, and Mean Absolute Error for regression tasks. Split, Validation, and Test are abbreviated as S, V, and T for readability. Bold is the best performance.

Task	S		Model						
			Baseline	MAE-0.25	MAE-0.5	CTR-0.25	CTR-0.5	TVE-1	TVE-2
Classification									
rel-amz/ item-churn	least 50	V	76.04 ± 3.84	76.07 ± 4.38	76.06 ± 3.19	84.82 ± 2.12	84.32 ± 2.41	86.56 ± 1.31	**87.11 ± 2.06**
		T	73.39 ± 6.63	71.33 ± 5.96	67.02 ± 6.89	**83.97 ± 2.72**	79.89 ± 4.41	81.97 ± 0.95	83.25 ± 2.40
	least 100	V	76.64 ± 1.11	75.31 ± 2.05	74.03 ± 2.47	80.99 ± 1.27	80.09 ± 1.68	80.94 ± 1.35	**82.79 ± 1.20**
		T	68.65 ± 1.91	68.48 ± 2.58	70.12 ± 1.58	74.78 ± 1.56	73.86 ± 1.68	75.29 ± 2.27	**76.74 ± 1.46**
rel-amz/ user-churn	top 50	V	91.35 ± 1.27	91.53 ± 1.47	91.66 ± 1.13	91.01 ± 1.82	91.09 ± 1.10	**93.31 ± 1.19**	91.82 ± 1.25
		T	89.07 ± 2.14	87.86 ± 2.00	85.71 ± 3.37	89.66 ± 2.26	89.71 ± 1.90	**93.21 ± 1.58**	91.53 ± 1.84
	top 100	V	88.99 ± 0.61	90.00 ± 0.60	89.24 ± 0.68	91.32 ± 0.58	90.52 ± 0.86	**91.35 ± 0.48**	90.92 ± 0.81
		T	86.23 ± 0.64	86.69 ± 1.00	85.59 ± 1.66	84.86 ± 1.39	86.32 ± 1.41	87.74 ± 1.04	**88.13 ± 1.14**
rel-hm/ user-churn	top 50	V	74.41 ± 0.85	76.30 ± 1.48	76.47 ± 1.20	84.07 ± 2.07	**84.14 ± 1.74**	76.47 ± 2.43	74.97 ± 1.42
		T	60.23 ± 1.84	56.94 ± 3.59	58.11 ± 2.38	56.82 ± 3.05	57.48 ± 3.55	**65.01 ± 3.82**	64.12 ± 3.13
	top 100	V	72.63 ± 1.30	74.61 ± 1.48	76.22 ± 1.24	79.74 ± 0.67	**79.85 ± 1.41**	75.42 ± 1.53	75.60 ± 1.40
		T	63.43 ± 1.14	59.26 ± 1.77	60.25 ± 2.20	61.58 ± 1.15	60.69 ± 1.50	62.85 ± 1.87	**64.06 ± 2.29**
Regression									
rel-amz/ item-ltv	least 100	V	0.175 ± 0.000	0.174 ± 0.001	0.172 ± 0.003	0.175 ± 0.001	0.174 ± 0.002	0.166 ± 0.005	**0.165 ± 0.006**
		T	0.186 ± 0.000	0.184 ± 0.003	0.185 ± 0.002	0.187 ± 0.000	0.188 ± 0.003	**0.181 ± 0.005**	0.188 ± 0.006
rel-amz/ user-ltv	bad- rev.	V	8.172 ± 0.000	8.173 ± 0.000	8.172 ± 0.000	8.172 ± 0.000	8.172 ± 0.000	8.117 ± 0.008	**8.115 ± 0.010**
		T	9.520 ± 0.000	9.520 ± 0.000	9.520 ± 0.000	9.520 ± 0.000	9.520 ± 0.000	**9.372 ± 0.026**	9.373 ± 0.010
rel-hm/ item-sales	top 200	V	3.203 ± 0.043	3.139 ± 0.083	3.111 ± 0.062	3.170 ± 0.090	3.113 ± 0.084	2.974 ± 0.098	**2.945 ± 0.028**
		T	3.240 ± 0.049	3.271 ± 0.120	3.249 ± 0.039	3.277 ± 0.135	3.284 ± 0.117	**3.176 ± 0.078**	3.205 ± 0.081

Table 2: Performance across datasets and model variants in the sufficient-data regime [46]. Reported metrics are ROC-AUC for classification tasks, and Mean Absolute Error for regression tasks. Split, Validation, and Test are abbreviated as S, V, and T for readability. Bold is the best performance.

Task	S	Model						
		Baseline	MAE-0.25	MAE+TVE-1	MAE+TVE-2	CTR-0.25	CTR+TVE-1	CTR+TVE-2
Classification								
rel-amz/ item-churn	V	81.54 ± 0.06	**81.61 ± 0.14**	81.13 ± 0.04	81.19 ± 0.03	81.27 ± 0.08	81.29 ± 0.09	81.28 ± 0.10
	T	81.85 ± 0.10	**81.92 ± 0.16**	81.47 ± 0.04	81.49 ± 0.04	81.53 ± 0.09	81.57 ± 0.13	81.56 ± 0.12
rel-amz/ user-churn	V	69.95 ± 0.09	69.26 ± 0.22	69.92 ± 0.04	69.93 ± 0.05	69.85 ± 0.05	**70.04 ± 0.04**	70.03 ± 0.06
	T	69.82 ± 0.05	68.89 ± 0.29	69.78 ± 0.07	69.80 ± 0.03	69.64 ± 0.09	**69.95 ± 0.02**	69.85 ± 0.07
rel-hm/ user-churn	V	70.34 ± 0.09	70.39 ± 0.06	70.29 ± 0.09	70.35 ± 0.10	70.49 ± 0.13	70.53 ± 0.04	**70.54 ± 0.16**
	T	69.77 ± 0.33	69.81 ± 0.20	69.89 ± 0.09	69.80 ± 0.11	69.97 ± 0.09	**70.13 ± 0.22**	69.72 ± 0.51
Regression								
rel-amz/ item-ltv	V	45.818 ± 0.301	45.573 ± 0.091	45.590 ± 0.121	45.573 ± 0.045	45.630 ± 0.161	45.030 ± 0.100	**44.956 ± 0.078**
	T	50.603 ± 0.351	50.471 ± 0.308	50.437 ± 0.395	50.659 ± 0.417	50.472 ± 0.432	50.012 ± 0.287	**49.753 ± 0.192**
rel-amz/ user-ltv	V	12.416 ± 0.016	15.989 ± 1.154	12.429 ± 0.016	12.437 ± 0.031	12.434 ± 0.014	12.397 ± 0.010	**12.371 ± 0.018**
	T	14.665 ± 0.039	17.397 ± 0.907	14.658 ± 0.028	14.626 ± 0.019	14.687 ± 0.057	14.622 ± 0.046	**14.577 ± 0.020**
rel-hm/ item-sales	V	0.0643 ± 2e-4	0.0630 ± 1e-4	0.0627 ± 1e-4	**0.0624 ± 3e-4**	0.0635 ± 2e-4	0.0628 ± 4e-4	0.0627 ± 3e-4
	T	0.0556 ± 2e-4	0.0547 ± 1e-4	0.0544 ± 2e-4	**0.0543 ± 3e-4**	0.0554 ± 3e-4	0.0547 ± 3e-4	**0.0543 ± 3e-4**

- RQ2: Can TVE provide better latent representations compared to traditional SSL methods for downstream tasks?
- RQ3: How sensitive is our model to different hyperparameter settings and last linear head's initialization?

5.1 Experimental Settings

We compare TVE with other SSL methods that do not take into account next-window dynamics—Graph Masked Autoencoder and Graph Contrastive Learning. Specifically, these architectures resemble GraphMAE [31] and GraphCL [56], with minor modifications for RDL pre-training. All evaluated models used the same backbone proposed by [46]. We compare several models: a non-pretrained Baseline; Graph Masked Autoencoder with mask rates of 0.25 and 0.5 (MAE-0.25,

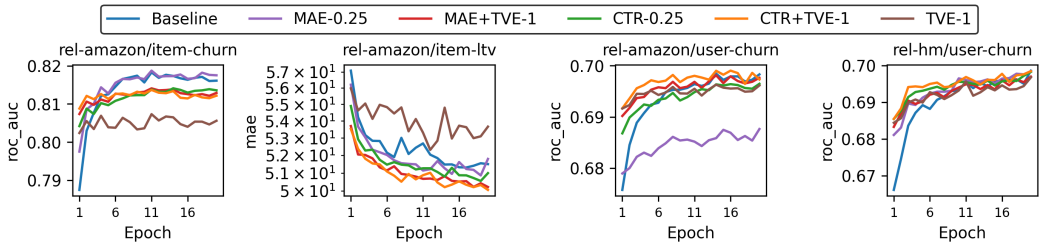


Figure 4: Test performance curves averaged over runs on data sufficient tasks.

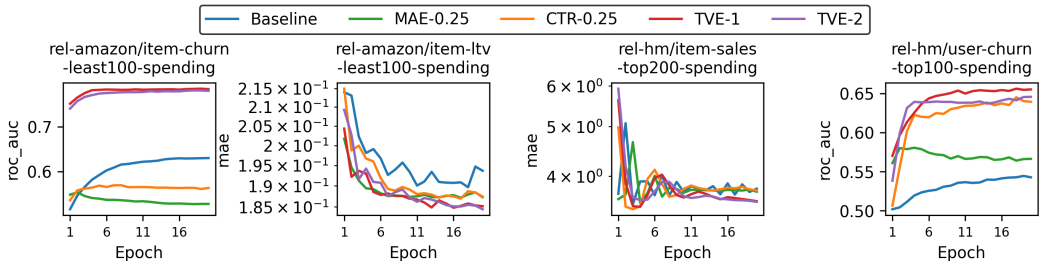


Figure 5: Linear probing results averaged over runs on test set.

MAE-0.5); Graph Contrastive Learning with the same mask rates (CTR-0.25, CTR-0.5); and our proposed method, TVE, using 1-hop and 2-hop schema traversals (TVE-1, TVE-2). All pre-training is conducted using the same training table constructed via `CROSS PRODUCT` as described in Section 4.2. While there are many pre-training techniques designed for vertical domains, their translations to RDL are non-trivial due to the fundamental differences outlined in Section 2; therefore, we only consider pre-training techniques based on tabular feature perturbations (see Appendix C for details).

We evaluate our proposed method on a variety of node-level tasks defined over two datasets: `rel-amazon` and `rel-hm`. Additionally, we introduce a suite of new tasks designed to reflect realistic scenarios. For instance, a business might wish to build predictive models for a specific cohort (e.g. low-spending customers) where label data is scarce. These scenarios often lack rich temporal context, so dynamics presented by labels alone may not support high-quality forecasting. Yet these tasks are crucial, since companies increasingly aim to personalize offerings for these groups to boost retention. To generate these tasks, we apply attribute-based filters on the prior period: for instance, an SQL query retrieves the top-k spenders over the last three months, and other customers are excluded from the training data for the current period.

To address this, we explicitly evaluate model performance under two regimes: (i) data limited tasks and (ii) data sufficient tasks (the original RelBench tasks). More details are outlined in Appendix C.

5.2 Data Limited Tasks

Table 1 presents results for tasks with limited labeled supervision. We find that TVE consistently outperforms other SSL baselines, achieving the best performance in 5 out of 6 classification tasks. The only exception is `rel-amazon/item-churn-least50-spending`, where CTR-0.25 achieves the highest test ROC-AUC. However, this task illustrates a form of *flaky behavior*: while CTR-0.25 and CTR-0.5 exhibit nearly identical validation scores (84.82 ± 2.12 vs. 84.32 ± 2.41), their performance diverges significantly on test set (83.97 ± 2.72 vs. 79.89 ± 4.41). This suggests sensitivity to hyperparameter tuning (e.g., augmentation strength), which are later studied in Section 5.5.

More broadly, MAE and CTR often excel on validation but falter on the test set—sometimes below the Baseline—likely due to distribution shift and recency bias from temporal splits. Our method explicitly captures next-window dynamics during pre-training, allowing it to generalize across time. Similar patterns hold in regression tasks: for instance, in `rel-amazon/user-ltv-bad-reviews`, MAE and CTR match Baseline performance, while TVE allows the model to achieve better performance. These results answer RQ1, which demonstrates the effectiveness of TVE.

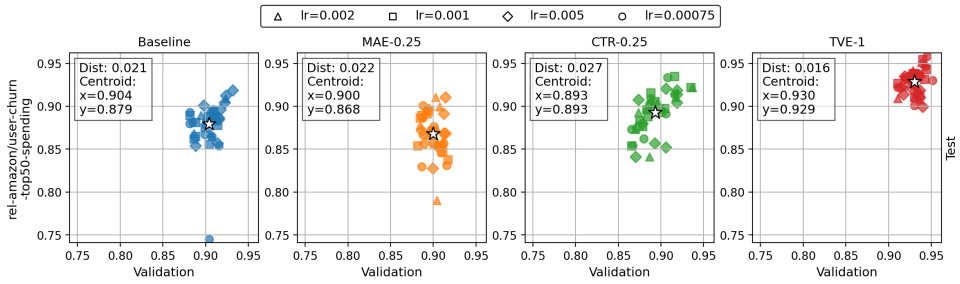


Figure 6: Validation vs. Test scatter plots for data limited node classification tasks across varying learning rates. Average distance to centroid (denoted ☆) are used to measure performance variance.

5.3 Data Sufficient Tasks

In Table 2, we benchmark each model on the original RelBench tasks, which involve substantially larger downstream datasets. In data-rich settings, all SSL methods—including vanilla TVE—produce only marginal gains or even underperform, since plentiful fine-tuning data alone can recover temporal patterns. This matches Theorem 4.1: TVE's extra pre-training benefit arises only if the representation remains jointly sufficient for both inputs and task vectors, so any drop signals lost input information. As mentioned in Section 3, TVE is complementary to existing SSL methods. Therefore, we employ the combined-loss model from Eq. 5, which penalizes the omission of original information. As Fig. 4 demonstrates, this hybrid objective produces a stronger initialization, and Table 2 confirms that it outperforms single-objective models on five of the six tasks.

5.4 Linear Probing Performance

To answer RQ2, we evaluate the quality of the learned representations independently of downstream fine-tuning by linear probing experiments [62]. Specifically, we freeze the pre-trained backbone and train only a linear head that maps latent representations to task outputs. All model shared the same backbone with the channel size of 128.

As shown in Fig. 5, representations learned via TVE outperform all other SSL methods, as well as the Baseline. Furthermore, CTR-0.25 achieves strong performance on three of four tasks, yet falls below the Baseline on `rel-amazon/item-churn-least100-spending`, illustrating the presence of "interesting-looking noise" [40]. In contrast, TVE does not rely on augmentations, thus yielding stable and transferable embeddings.

5.5 Performance Sensitivity Across Splits and Tasks

In order to answer RQ3, we further investigate the robustness of each method to hyperparameter variation by analyzing the stability of performance across learning rates and last linear head's random initialization. We vary the learning rate across 10 runs per task, yielding 40 data points per scatterplot in Fig. 6. All models share the same backbone with the channel size of 128.

The results show that MAE-0.25 and CTR-0.25 are highly sensitive to hyperparameters and last linear head's random initialization: the spread of points is substantially larger compared to TVE. These results suggest that the task-aware, temporal pre-training of TVE leads to more stable and transferable representations in the low-data setting. Experiments for other tasks are provided in Appendix C.

6 Related Works, Discussion, and Conclusion

Relational Deep Learning. RDL has emerged as a paradigm for learning from RDBs [10, 21, 24, 25, 46, 51]. Although SSL techniques are advanced for individual domains—temporal graphs [7, 15, 16, 38, 39, 48], static graphs [31, 50, 56], and tabular data [9, 49, 55]—SSL investigations on RDBs are still limited [40]. Drawing on [43]'s k-means column-subset pretext tasks, TVE instead generates objectives via schema-graph traversals and temporal SQL aggregations. Unlike SQL-join feature synthesis [34] or recent RDL advances on architectures [57, 61] and efficiency [36], our approach centers on task-space representations for SSL pre-training.

Task Modeling. Generalizing to new tasks remains a core challenge. Prior approaches include task taxonomies to reveal inter-task structure [59], task embeddings for selecting feature extractors [3, 14], and modeling data–auxiliary task node interactions [13]. In contrast, we analyze label-generation in relational deep learning and propose a pretext objective that explicitly preserves the next-window set representations from which any downstream labels are derived.

Representation Learning. Recent SSL–information-theoretic work frames representations as noise-invariant [2] and studies how multi-view SSL discards redundant features for downstream tasks [22]. Augmentations can erase view-specific cues, underscoring the need to preserve task-relevant signals [37]. Another important SSL technique is to train models to predict future observations (e.g., frames or tokens) [41, 44]. TVE follows these principles by forecasting next-window dynamics to capture the core information for downstream labels.

Limitations. Although our theory leverages sufficient statistics, task vectors—built via simple SQL aggregations—may not fully capture the original next-window sets, thus leading to loss of information. Their dimensionality grows with the number of columns and joinable paths, which necessitates dimensionality reduction techniques [1, 8]. Finally, a learnable representation for next-window sets may be more expressive; we leave this exploration to future work.

Conclusion. We introduce Task Vector Estimation, a pre-training framework that integrates schema graphs, temporal dynamics, and task heterogeneity. Our theoretical results show that task-aware objectives retain more downstream-relevant information than conventional SSL methods, producing representations that are predictive for downstream tasks. Our findings underscore the importance of treating task heterogeneity and temporal context as first-class citizens in RDL pre-training strategies.

7 Acknowledgements

Quang Truong, Zhikai Chen, and Jiliang Tang are supported by the National Science Foundation (NSF) under grant numbers CNS2321416, IIS2212032, IIS2212144, IIS 2504089, DUE2234015, CNS2246050, DRL2405483 and IOS2035472, the Michigan Department of Agriculture and Rural Development, US Dept of Commerce, Gates Foundation, Amazon Faculty Award, Meta, NVIDIA, Microsoft and SNAP.

References

[1] Hervé Abdi and Dominique Valentin. Multiple correspondence analysis. *Encyclopedia of measurement and statistics*, 2(4):651–657, 2007.

[2] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(50):1–34, 2018.

[3] Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C. Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[4] B Aditya, Gaurav Bhalotia, Soumen Chakrabarti, Arvind Hulgeri, Charuta Nakhe, S Sudarshanxe, et al. Banks: Browsing and keyword searching in relational databases. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 1083–1086. Elsevier, 2002.

[5] Rakesh Agrawal, Amit Somani, and Yirong Xu. Storage and querying of e-commerce data. In *VLDB*, volume 1, pages 149–158, 2001.

[6] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=HyxQzBceg`.

[7] Mohammad Alomrani, Mahdi Biparva, Yingxue Zhang, and Mark Coates. Dyg2vec: Efficient representation learning for dynamic graphs. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=YRKS2J0x36`.

[8] Karl Pearson and. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901. doi: 10.1080/14786440109462720. URL https://doi.org/10.1080/14786440109462720.

[9] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CuV_qYkmKb3.

[10] Gleb Bazhenov, Oleg Platonov, and Liudmila Prokhorenkova. Tabgraphs: A benchmark and strong baselines for learning on graphs with tabular node features, 2024. URL https://arxiv.org/abs/2409.14500.

[11] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 531–540. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/belghazi18a.html.

[12] Lukas Biewald. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

[13] Kaidi Cao, Jiaxuan You, and Jure Leskovec. Relational multi-task learning: Modeling relations between data and tasks. *arXiv preprint arXiv:2303.07666*, 2023.

[14] Kaidi Cao, Jiaxuan You, Jiaju Liu, and Jure Leskovec. Autotransfer: Automl with knowledge transfer–an application to graph neural networks. *arXiv preprint arXiv:2303.07669*, 2023.

[15] Ke-Jia Chen, Jiajun Zhang, Linpu Jiang, Yunyun Wang, and Yuxuan Dai. Pre-training on dynamic graph neural networks. *Neurocomputing*, 500:679–687, 2022.

[16] Ke-Jia Chen, Linsong Liu, Linpu Jiang, and Jingqiang Chen. Self-supervised dynamic graph representation learning via temporal subgraph contrast. *ACM Trans. Knowl. Discov. Data*, 18 (1), September 2023. ISSN 1556-4681. doi: 10.1145/3612931. URL https://doi.org/10.1145/3612931.

[17] Kuan-Yu Chen, Ping-Han Chiang, Hsin-Rung Chou, Ting-Wei Chen, and Darby Tien-Hao Chang. Trompt: towards a better deep neural network for tabular data. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[18] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[19] Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[20] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. ISBN 0471241954.

[21] Milan Cvitkovic. Supervised learning on relational databases with graph neural networks, 2020. URL https://arxiv.org/abs/2002.02046.

[22] Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *International Conference on Learning Representations*, 2020.

[23] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[24] Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Relational deep learning: Graph representation learning on relational databases, 2023. URL https://arxiv.org/abs/2312.04615.

[25] Quan Gan, Minjie Wang, David Wipf, and Christos Faloutsos. Graph machine learning meets multi-table relational data. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6502–6512, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671471. URL https://doi.org/10.1145/3637528.3671471.

[26] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272. JMLR.org, 2017.

[27] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

[28] Terry Halpin and Tony Morgan. *Information modeling and relational databases*. Morgan Kaufmann, 2010.

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

[30] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[31] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 594–604, 2022.

[32] Weihua Hu, Yiwen Yuan, Zecheng Zhang, Akihiro Nitta, Kaidi Cao, Vid Kocijan, Jure Leskovec, and Matthias Fey. Pytorch frame: A modular framework for multi-modal tabular learning. *arXiv preprint arXiv:2404.00776*, 2024.

[33] Tao Huang, Yanxiang Ma, Shan You, and Chang Xu. Learning mask invariant mutual information for masked image modeling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=NoiaAT0eec.

[34] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*, pages 1–10. IEEE, 2015.

[35] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[36] Veronica Lachi, Antonio Longa, Beatrice Bevilacqua, Bruno Lepri, Andrea Passerini, and Bruno Ribeiro. Boosting relational deep learning with pretrained tabular models, 2025. URL https://arxiv.org/abs/2504.04934.

[37] Paul Pu Liang, Zihao Deng, Martin Q. Ma, James Zou, Louis-Philippe Morency, and Russ Salakhutdinov. Factorized contrastive learning: Going beyond multi-view redundancy. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=alLs7EtRJP.

[38] Lingwen Liu, Guangqi Wen, Peng Cao, Jinzhu Yang, Weiping Li, and Osmar R. Zaiane. Capturing temporal node evolution via self-supervised learning: A new perspective on dynamic graph learning. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 443–451, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703713. doi: 10.1145/3616855.3635765. URL https://doi.org/10.1145/3616855.3635765.

[39] Meng Liu, Ke Liang, Yawei Zhao, Wenxuan Tu, Sihang Zhou, Xinbiao Gan, Xinwang Liu, and Kunlun He. Self-supervised temporal graph learning with temporal and structural intensity alignment. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[40] Shengchao Liu, David Vazquez, Jian Tang, and Pierre-André Noël. Flaky performances when pretraining on relational databases (student abstract). In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i13.26993. URL https://doi.org/10.1609/aaai.v37i13.26993.

[41] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[42] Jan Motl and Oliver Schulte. The ctu prague relational learning repository. *arXiv preprint arXiv:1511.03086*, 2015.

[43] Jaehyun Nam, Jihoon Tack, Kyungmin Lee, Hankook Lee, and Jinwoo Shin. STUNT: Few-shot tabular learning with self-generated tasks from unlabeled tables. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=_xlsjehDvlY.

[44] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[46] Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan Eric Lenssen, Yiwen Yuan, Zecheng Zhang, Xinwei He, and Jure Leskovec. Relbench: A benchmark for deep learning on relational databases. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=WEFxOm3Aez.

[47] Ivan Rubachev, Artem Alekberov, Yury Gorishniy, and Artem Babenko. Revisiting pretraining objectives for tabular deep learning, 2022. URL https://arxiv.org/abs/2207.03208.

[48] Sheng Tian, Ruofan Wu, Leilei Shi, Liang Zhu, and Tao Xiong. Self-supervised representation learning on dynamic graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 1814–1823, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384469. doi: 10.1145/3459637.3482389. URL https://doi.org/10.1145/3459637.3482389.

[49] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. Subtab: Subsetting features of tabular data for self-supervised representation learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vrhNQ7aYSdr.

[50] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rklz9iAcKQ.

[51] Minjie Wang, Quan Gan, David Wipf, Zheng Zhang, Christos Faloutsos, Weinan Zhang, Muhan Zhang, Zhenkun Cai, Jiahang Li, Zunyao Mao, et al. 4dbinfer: A 4d benchmarking toolbox for graph-centric predictive modeling on rdbs. *Advances in Neural Information Processing Systems*, 37:27236–27273, 2024.

[52] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The World Wide Web Conference*, WWW '19, page 2022–2032, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313562. URL https://doi.org/10.1145/3308558.3313562.

[53] Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. Griffin: Towards a graph-centric relational database foundation model. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=TxeCxVb3cL.

[54] Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4(1):66–82, 1960. doi: 10.1147/rd.41.0066.

[55] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela Van der Schaar. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in neural information processing systems*, 33:11033–11043, 2020.

[56] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

[57] Yiwen Yuan, Zecheng Zhang, Xinwei He, Akihiro Nitta, Weihua Hu, Manan Shah, Blaž Stojanovič, Shenyang Huang, Jan Eric Lenssen, Jure Leskovec, and Matthias Fey. ContextGNN: Beyond two-tower recommendation systems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=nzOD1we8Z4.

[58] Manzil Zaheer, Satwik Kottur, Siamak Ravanbhakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J Smola. Deep sets. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3394–3404, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[59] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[60] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 793–803, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330961. URL https://doi.org/10.1145/3292500.3330961.

[61] Han Zhang, Quan Gan, David Wipf, and Weinan Zhang. Gfs: Graph-based feature synthesis for prediction over relational databases, 2023. URL https://arxiv.org/abs/2312.02037.

[62] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduce accurately describe the proposed framework, namely Task Vector Estimation and theoretical contributions in Section 4. Experimental results with performance gains are included in Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The theoretical assumptions are clearly stated in the statements in main texts and Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper discusses the proposed method in depth in Section 3 and Section 4, with experiment details in Section 5 and Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Codes, with sufficient instructions, are included in a public URL.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed experimental settings are stated in both main texts and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Result tables include mean performance and standard deviation for every experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational details are included in Appendix, where we state the CPU and GPU of machines we use.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We do not observe any negative societal impacts for our research.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not observe any risks associated with our research.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have given proper attributions to the original authors cited in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide details of our proposed tasks in the appendix and codes in the public URL.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We only use LLMs for editing writing.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Proofs

In this section, we assume that all random variables $\mathbf{x}, \mathbf{t}, \mathbf{y}, \mathbf{z}_1, \mathbf{z}_2$ are discrete. We use the following identities to prove the theorem reported in Section 4.3. For any random variable $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$, we have the following properties:

Non-negativity:

$$I(\mathbf{x}; \mathbf{y}) \geq 0, \quad I(\mathbf{x}; \mathbf{y} \mid \mathbf{z}) \geq 0. \tag{7}$$

Chain rule:

$$I(\mathbf{xy}; \mathbf{z}) = I(\mathbf{y}; \mathbf{z}) + I(\mathbf{x}; \mathbf{z} \mid \mathbf{y}). \tag{8}$$

Chain rule (Multivariate Mutual Information):

$$I(\mathbf{x}; \mathbf{y}; \mathbf{z}) = I(\mathbf{y}; \mathbf{z}) - I(\mathbf{y}; \mathbf{z} \mid \mathbf{x}). \tag{9}$$

Entropy and Mutual Information:

$$H(\mathbf{x}) = H(\mathbf{x} \mid \mathbf{y}) + I(\mathbf{x}; \mathbf{y}). \tag{10}$$

Conditional Mutual Information and Conditional Entropy:

$$I(\mathbf{x}; \mathbf{y} \mid \mathbf{z}) = H(\mathbf{x} \mid \mathbf{z}) - H(\mathbf{x} \mid \mathbf{yz}). \tag{11}$$

Federici et al. [22] assumes the following property when $\mathbf{z_2}$ is a representation of $\mathbf{x}$, which implies that the only source of stochastic in $\mathbf{z_2}$ is from $\mathbf{x}$:

$$I(\mathbf{z_2}; \mathbf{y} \mid \mathbf{x}) = 0. \tag{12}$$

Federici et al. [22] also proves the following proposition, which implies that $\mathbf{z_2}$ must be as predictive as $\mathbf{x}$ to predict $\mathbf{y}$ when $\mathbf{z_2}$ is a sufficient representation of $\mathbf{x}$ for $\mathbf{y}$.

Proposition A.1 ([22])**.** *Let $\mathbf{x}$ and $\mathbf{y}$ be random variables with joint distribution $p(\mathbf{x}, \mathbf{y})$. Let $\mathbf{z_2}$ is a representation of $\mathbf{x}$. Then $\mathbf{z_2}$ is sufficient for $\mathbf{y}$ if and only if:*

$$I(\mathbf{x}; \mathbf{y}) = I(\mathbf{y}; \mathbf{z_2}). \tag{13}$$

Similarly, we also assume stochasticity for $\mathbf{z_1}$ is conditionally independent from any other variables in the system once $\mathbf{x}$ and $\mathbf{t}$ are observed:

$$I(\mathbf{z_1}; \mathbf{y} \mid \mathbf{xt}) = 0. \tag{14}$$

A.1 Proof for Theorem 4.1

First, we need to prove sufficient representation of inputs and side-channel information for a task implies that the representation does not discard any task-relevant information contained in the inputs and side channels.

Proposition A.2. *Let $\mathbf{x}, \mathbf{t}, \mathbf{y}$ be random variables with joint distribution $p(\mathbf{x}, \mathbf{t}, \mathbf{y})$. Let $\mathbf{z_1}$ be a latent representation of $\mathbf{x}$ with additional side-channel information $\mathbf{t}$. Then $\mathbf{z_1}$ is sufficient of $\mathbf{x}$ and $\mathbf{t}$ for $\mathbf{y}$ if and only if $I(\mathbf{xt}; \mathbf{y}) = I(\mathbf{y}; \mathbf{z_1})$.*

Proof.

$$\begin{aligned} I(\mathbf{xt}; \mathbf{y} \mid \mathbf{z_1}) &\overset{(9)}{=} I(\mathbf{xt}; \mathbf{y}) - I(\mathbf{xt}; \mathbf{y}; \mathbf{z_1}) \\ &\overset{(9)}{=} I(\mathbf{xt}; \mathbf{y}) - I(\mathbf{y}; \mathbf{z_1}) - I(\mathbf{y}; \mathbf{z_1} \mid \mathbf{xt}) \\ &\overset{(14)}{=} I(\mathbf{xt}; \mathbf{y}) - I(\mathbf{y}; \mathbf{z_1}) \end{aligned}$$

Thus, $\mathbf{z_1}$ is sufficient of $\mathbf{x}$ and $\mathbf{t}$ for $\mathbf{y}$ ($I(\mathbf{xt}; \mathbf{y} \mid \mathbf{z_1}) = 0$) if and only if $I(\mathbf{xt}; \mathbf{y}) = I(\mathbf{y}; \mathbf{z_1})$ □

Consequently, the mutual information between the sufficient representation $\mathbf{z_1}$ and the label $\mathbf{y}$ is at least as large as that of the input $\mathbf{x}$ and the label $\mathbf{y}$.

Corollary A.1. *Let* $\mathbf{x}, \mathbf{t}, \mathbf{y}$ *be random variables with joint distribution* $p(\mathbf{x}, \mathbf{t}, \mathbf{y})$*. Let* $\mathbf{z_1}$ *be a sufficient latent representation of* $\mathbf{x}$ *with additional side-channel information* $\mathbf{t}$ *for* $\mathbf{y}$*. Then,* $I(\mathbf{z_1}; \mathbf{y}) \geq I(\mathbf{x}; \mathbf{y})$*.*

Proof.

$$I(\mathbf{y}; \mathbf{z_1}) = I(\mathbf{xt}; \mathbf{y}) \overset{(8)}{=} I(\mathbf{x}; \mathbf{y}) + I(\mathbf{t}; \mathbf{y} \mid \mathbf{x}) \overset{(7)}{\geq} I(\mathbf{x}; \mathbf{y}).$$

□

The equality occurs when the side-channel information are irrelevant to the downstream task given the observation of the input. Lastly, we prove our main theorem.

Theorem 4.1. *Let* $\mathbf{x}, \mathbf{t}, \mathbf{y}$ *be random variables with joint distribution* $p(\mathbf{x}, \mathbf{t}, \mathbf{y})$*. Assume that* $\mathbf{z_1}$ *is sufficient representation of* $\mathbf{x}$ *with side-channel information* $\mathbf{t}$ *for* $\mathbf{y}$*, and* $\mathbf{z_2}$ *is sufficient representation of* $\mathbf{x}$ *for* $\mathbf{y}$*. Then mutual information between* $\mathbf{z_1}$ *and* $\mathbf{y}$ *is at least as much as that between* $\mathbf{z_2}$ *and* $\mathbf{y}$*:*

$$I(\mathbf{z_1}; \mathbf{y}) \geq I(\mathbf{z_2}; \mathbf{y}).$$

Proof. From Corollary A.1, we have:

$$I(\mathbf{z_1}; \mathbf{y}) \geq I(\mathbf{x}; \mathbf{y}) \tag{15}$$

$$\overset{(10)}{\iff} -H(\mathbf{y} \mid \mathbf{z_1}) + H(\mathbf{y}) \geq -H(\mathbf{y} \mid \mathbf{x}) + H(\mathbf{y}) \tag{16}$$

$$\iff H(\mathbf{y} \mid \mathbf{z_1}) \leq H(\mathbf{y} \mid \mathbf{x}) \tag{17}$$

Then we expand $I(\mathbf{z_1}; \mathbf{y})$:

$$\begin{aligned} I(\mathbf{z_1}; \mathbf{y}) &\overset{(8)}{=} I(\mathbf{xz_1}; \mathbf{y}) - I(\mathbf{x}; \mathbf{y} \mid \mathbf{z_1}) \\ &\overset{(8)}{=} I(\mathbf{x}; \mathbf{y}) + I(\mathbf{z_1}; \mathbf{y} \mid \mathbf{x}) - I(\mathbf{x}; \mathbf{y} \mid \mathbf{z_1}) \\ &\overset{(11)}{=} I(\mathbf{x}; \mathbf{y}) + (H(\mathbf{y} \mid \mathbf{x}) - H(\mathbf{y} \mid \mathbf{xz_1})) - (H(\mathbf{y} \mid \mathbf{z_1}) - H(\mathbf{y} \mid \mathbf{xz_1})) \\ &= I(\mathbf{x}; \mathbf{y}) + H(\mathbf{y} \mid \mathbf{x}) - H(\mathbf{y} \mid \mathbf{z_1}) \end{aligned}$$

Given Eq. 17 and Proposition A.1, we have:

$$I(\mathbf{z_1}; \mathbf{y}) \geq I(\mathbf{z_2}; \mathbf{y})$$

□

B Formal Definition of Label Generation

In Section 4.1, we provided a compact notation for RDL labels, assuming that k-hop neighbors share the same entity type and that timestamps are discretized with natural integer indices. In this section, we provide the full notations for the label generation process. Formally, downstream labels can be represented as:

$$y_v^{(t_v, t_v + \Delta t]} = l\big({}^{k}_{R}\mathbf{X}_v^{(t_v, t_v + \Delta t]}\big), \tag{18}$$

$${}^{k}_{R}\mathbf{X}_v^{(t_v, t_v + \Delta t]} = \big\{ f_{\mathcal{V}}(u) \mid u \in \mathcal{N}_k^{(t_v, t_v + \Delta t]}(v) \wedge \varphi_{\mathcal{V}}(u) = R \big\}, \tag{19}$$

where l is a labeling set function and ${}^{k}_{R}\mathbf{X}_v^{(t_v, t_v + \Delta t]}$ denotes the set of k-hop neighbors' features of v, of entity type R, which appear within the prediction window, and $\mathcal{N}_k^{(t_v, t_v + \Delta t]}$ denotes the set of temporal k-hop neighbors based on schema traversal. $y_v^{(t_v, t_v + \Delta t]}$ corresponds to $y_v^{(t)}$ in Eq. 3, which represents the label associated with the entity v at the cutoff timestamp t.

C Experiments

In this section, we would like to provide further details regarding our experiments. All of the experiments are implemented in Pytorch [45], Pytorch Geometric [23], and Pytorch Frame [32]. All of the experiments are executed on one of the following 4 machines: 1) AMD EPYC 7763 64-Core Processor and NVIDIA RTX A6000, 2) AMD EPYC 7543 32-Core Processor and NVIDIA RTX A5000, 3) AMD EPYC 7543 32-Core Processor and NVIDIA RTX A5000, and 4) AMD EPYC 7513 32-Core Processor and NVIDIA RTX A6000. Experiments are logged and kept tracked by Weights & Biases [12].

C.1 Dataset and Task Descriptions

Here, we provide details regarding datasets and tasks for our experiments. Datasets are proposed by RelBench [46]. In addition to the original tasks proposed by RelBench [46], we also provide additional data limited tasks.

`rel-amazon` The `rel-amazon` dataset is sourced from the Amazon e-commerce platform. It comprises four original downstream tasks [46]:

- `user-churn`: predict whether a user remains inactive (i.e., has no interactions) over the next three months.
- `item-churn`: predict whether an item remains without interactions over the next three months.
- `user-ltv`: estimate the total monetary value of purchases made by a user over the next three months.
- `item-ltv`: estimate the total monetary value of purchases received by an item over the next three months.

Additionally, we propose data limited tasks, which are conditioned on a certain attribute of entities up to the cutoff timestamp t and period Δt, where $\Delta t = 90$ days:

- `item-churn-least-k-spending`: among k least bought products in the previous period $(t - \Delta t, t]$, predict products that remains without interactions over the next three months.
- `user-churn-top-k-spending`: among k top spending customers in the previous period $(t - \Delta t, t]$, predict those who will be inactive over the next three months.
- `item-ltv-least-k-spending`: among k least spending products in the previous period $(t - \Delta t, t]$, estimate the total monetary value of purchases received by an item over the next three months.
- `user-ltv-bad-reviews` (abbreviated as rel-amz/user-ltv-bad-rev. in Table 1): among those customers who only left bad reviews (review score of 1, 2, or 3) in the previous period $(t - \Delta t, t]$, estimate the total monetary value of purchases made by those users over the next three months.

`rel-hm` The `rel-hm` dataset is sourced from the H&M relational database. Similar to `rel-amazon`, `rel-hm` comprises similar tasks as originally proposed by [46]:

- `user-churn`: predict if a customer does not make any transactions in the next 7 days.
- `item-sales`: predict total sales for an article in the next 7 days.

Additionally, we also propose the following tasks, which are conditioned on a certain attribute of entities up to the cutoff timestamp t and period Δt, where $\Delta t = 7$ days:

- `user-churn-top-k-spending`: among k top spending customers in the previous period $(t - \Delta t, t]$, predict those who do not make any transactions in the next 7 days.
- `item-sales-top-k-spending`: among k top bought articles in the previous period $(t - \Delta t, t]$, estimate the total sales for those article in the next 7 days.

Table 3: Performance across datasets and TVE variants in the sufficient-data regime [46]. Reported metrics are ROC-AUC for classification tasks, and Mean Absolute Error for regression tasks. Split, Validation, and Test are abbreviated as S, V, and T for readability. Bold is the best performance.

Task	S	Model					
		TVE-1	TVE-2	MAE+TVE-1	MAE+TVE-2	CTR+TVE-1	CTR+TVE-2
Classification							
rel-amz/ item-churn	V	80.67 ± 0.06	80.76 ± 0.08	81.13 ± 0.04	81.19 ± 0.03	81.29 ± 0.09	81.28 ± 0.10
	T	80.91 ± 0.13	81.03 ± 0.10	81.47 ± 0.04	81.49 ± 0.04	81.57 ± 0.13	81.56 ± 0.12
rel-amz/ user-churn	V	69.85 ± 0.03	69.92 ± 0.04	69.92 ± 0.04	69.93 ± 0.05	70.04 ± 0.04	70.03 ± 0.06
	T	69.67 ± 0.08	69.78 ± 0.05	69.78 ± 0.07	69.80 ± 0.03	69.95 ± 0.02	69.85 ± 0.07
rel-hm/ user-churn	V	70.35 ± 0.11	70.38 ± 0.07	70.29 ± 0.09	70.35 ± 0.10	70.53 ± 0.04	70.54 ± 0.16
	T	69.86 ± 0.17	69.81 ± 0.17	69.89 ± 0.09	69.80 ± 0.11	70.13 ± 0.22	69.72 ± 0.51
Regression							
rel-amz/ item-ltv	V	48.183 ± 0.804	47.667 ± 0.401	45.590 ± 0.121	45.573 ± 0.045	45.030 ± 0.100	44.956 ± 0.078
	T	51.244 ± 0.229	51.167 ± 0.341	50.437 ± 0.395	50.659 ± 0.417	50.012 ± 0.287	49.753 ± 0.192
rel-amz/ user-ltv	V	12.442 ± 0.021	12.455 ± 0.018	12.429 ± 0.016	12.437 ± 0.031	12.397 ± 0.010	12.371 ± 0.018
	T	14.690 ± 0.058	14.676 ± 0.034	14.658 ± 0.028	14.626 ± 0.019	14.622 ± 0.046	14.577 ± 0.020
rel-hm/ item-sales	V	0.0637 ± 3e-4	0.0636 ± 2e-4	0.0627 ± 1e-4	0.0624 ± 3e-4	0.0628 ± 4e-4	0.0627 ± 3e-4
	T	0.0554 ± 4e-4	0.0553 ± 2e-4	0.0544 ± 2e-4	0.0543 ± 3e-4	0.0547 ± 3e-4	0.0543 ± 3e-4

C.2 Models

In this section, we would like to provide further details regarding models included in this study.

Baseline This is the base model adopted from RelBench [46] for node-level tasks.

Masked Autoencoder (MAE) Building on GraphMAE [31], we introduce the following modifications to adapt it to relational data. First, instead of masking entire node feature vectors, we apply cell-level masking—a technique well established in tabular learning [9, 27, 55]—and replace masked entries with values sampled from each column's marginal distribution. This obviates the need for a dedicated mask token and a re-masking stage. Our decoder still retains a projection layer after the GNN encoder, a single-layer GNN decoder, and last linear heads. Second, because cell-level masking can alter a large fraction of each row, we reconstruct the full feature vector, as in [40, 55]. We limit reconstruction to numerical, categorical, and textual features: normalized numerical values and text embeddings are recovered using mean-squared error loss, while categorical features are recovered via cross-entropy loss. Any entries missing in the original database are excluded from the loss computation. The final training objective is the sum of these individual reconstruction losses for all tables.

Contrastive Learning (CTR) Building on GraphCL [56], we make several adjustments similar to MAE. Specifically, we adopt the same masking augmentation as outlined in the previous paragraph. Also, follow SCARF [9], we consider two views of the inputs to be original features and augmented features, and the final objective is to maximize agreements between latent representations of two views via a symmetric InfoNCE loss [44] for all tables.

Task Vector Estimation (TVE) This is our proposed method, where we provide details in Section 4. We also apply mask augmentation with probability ($p = 0.15$) similar to MAE and CTR for regularization. A noteworthy observation is that our method is root-table-centric, which means that the objective (Eq. 4) is computed for a single table only. The hybrid models, namely MAE+TVE and CTR+TVE, also follow the same pipeline, with only changes in the decoder to cater for different loss objectives. For both hybrid models, we set the balancing coefficient β to 0.1 (see Eq. 5).

C.3 RelBench Experiments on Data Limited and Data Sufficient Tasks

In this section, we would like to provide further details regarding experiments reported in Section 5.2 and Section 5.3.

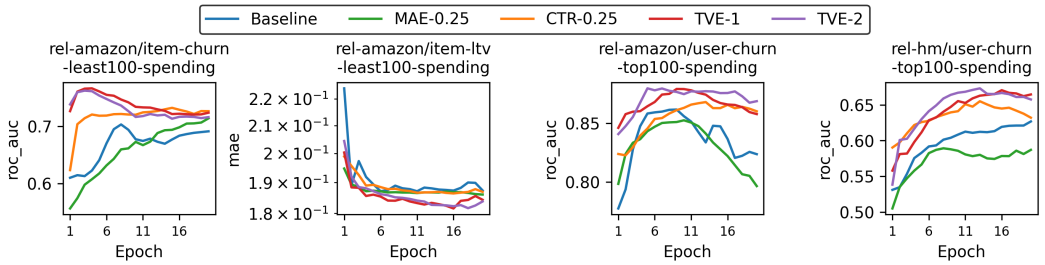


Figure 7: Test performance curves averaged over runs on data limited tasks.

Table 4: Hyperparameter settings for each model in Table 1. Each configuration is in the format [learning rate, channel size].

Task		Model						
		Baseline	MAE-0.25	MAE-0.5	CTR-0.25	CTR-0.5	TVE-1	TVE-2
rel-amz/ item-churn	least50	[0.001, 128]	[0.005, 256]	[0.001, 256]	[0.002, 256]	[0.005, 256]	[0.001, 256]	[0.001, 256]
	least100	[0.00075, 128]	[0.005, 128]	[0.005, 128]	[0.001, 256]	[0.002, 256]	[0.00075, 256]	[0.00075, 256]
rel-amz/ user-churn	top50	[0.005, 128]	[0.005, 256]	[0.005, 256]	[0.005, 256]	[0.005, 128]	[0.00075, 128]	[0.002, 256]
	top100	[0.002, 256]	[0.002, 256]	[0.005, 256]	[0.002, 128]	[0.001, 128]	[0.005, 256]	[0.005, 128]
rel-hm/ user-churn	top50	[0.001, 128]	[0.002, 128]	[0.002, 256]	[0.001, 256]	[0.00075, 256]	[0.005, 128]	[0.005, 128]
	top100	[0.005, 128]	[0.002, 128]	[0.001, 128]	[0.00075, 256]	[0.001, 256]	[0.005, 256]	[0.002, 256]
rel-amz/item-ltv	least100	[0.00075, 128]	[0.002, 128]	[0.002, 256]	[0.005, 256]	[0.005, 256]	[0.001, 256]	[0.002, 256]
rel-amz/user-ltv	bad-rev.	[0.00075, 128]	[0.00075, 128]	[0.00075, 128]	[0.00075, 128]	[0.00075, 128]	[0.00075, 256]	[0.00075, 128]
rel-hm/item-sales	top200	[0.001, 256]	[0.005, 256]	[0.005, 128]	[0.005, 256]	[0.005, 256]	[0.005, 256]	[0.005, 256]

For pre-training, we train these models for 81 epochs with the max steps per epoch to be 1000 per GPU, which effectively means 4000 steps per epoch due to distributed training. Learning rate is 0.001 for the first 50 epochs and reduces to 0.0001 for the remaining epochs. During fine-tuning, we fine-tune the pre-trained models for 20 epochs (except to `rel-hm/item-sales-top200-spending` where we fine-tune the models for 50 epochs). We also perform light hyperparameter tuning where we vary the following hyperparameters: learning rate $\in \{0.0075, 0.001, 0.002, 0.005\}$ and number of channels $\in \{128, 256\}$. For the original tasks proposed by RelBench, we adopt the learning rate of 0.005 and 0.002 for classification and regression tasks respectively. Similarly to the pre-training stage, we also perform distributed training on tasks with large amount of data. Data limited tasks are repeated 10 times (except to `rel-amazon/user-ltv-bad-reviews` and `rel-hm/item-sales-top200-spending` where we only repeat 5 times), while data sufficient tasks are repeated 5 times. All experiments use a batch size of 512 (reduced to 256 if batches exceed 24 GB VRAM), mean aggregation, two GNN layers, neighbor sampling of 128 then 64 hops, and uniform temporal sampling. Detailed hyperparameter differences are reported in Table 4 and Table 5 for data limited and data sufficient tasks, respectively.

Fig. 7 shows learning curves on low-data tasks, supplementing Table 1. We can observe that TVE pre-training both accelerates convergence and outperforms MAE and CTR. Table 3, which complements Table 2, compares all TVE variants in the data-rich regime and shows that combining vanilla TVE with any standard SSL objective yields additional gains, underscoring their complementary strengths.

C.4 Linear Probing Experiments

We would like to give more results supplementing the one reported in Section 5.4. Table 6 reports numerical results that are visualized in Fig. 5. Specifically, in the linear probing setting, TVE consistently outperforms both MAE and CTR on every evaluated task. Notably, for `rel-amazon/item-churn-least100-spending` and `rel-amazon/item-ltv-least100-spending`, neither MAE-0.25 nor CTR-0.25 yields any improvement, highlighting their unreliability as pre-training objectives for downstream tasks.

Table 5: Hyperparameter settings for each model in Table 2. Each configuration is in the format [learning rate, channel size].

Task	Model						
	Baseline	MAE-0.25	MAE + TVE-1	MAE + TVE-2	CTR-0.25	CTR + TVE-1	CTR + TVE-2
rel-amz/item-churn	[0.005, 128]	[0.005, 256]	[0.005, 128]	[0.005, 128]	[0.005, 256]	[0.005, 256]	[0.005, 256]
rel-amz/user-churn	[0.005, 128]	[0.005, 128]	[0.005, 128]	[0.005, 128]	[0.005, 128]	[0.005, 128]	[0.005, 256]
rel-hm/user-churn	[0.005, 256]	[0.005, 128]	[0.005, 256]	[0.005, 128]	[0.005, 256]	[0.005, 256]	[0.005, 256]
rel-amz/item-ltv	[0.002, 256]	[0.002, 256]	[0.002, 128]	[0.002, 128]	[0.002, 256]	[0.002, 256]	[0.002, 256]
rel-amz/user-ltv	[0.002, 128]	[0.002, 128]	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]
rel-hm/item-sales	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]	[0.002, 256]

Table 6: Linear probing results. Reported metric is ROC-AUC (higher is better) for classification tasks, and Mean Absolute Error (lower is better) for regression tasks. Split, Validation, and Test are abbreviated as S, V, and T for readability. Bold is the best performance.

Task	S	Model				
		Baseline	MAE-0.25	CTR-0.25	TVE-1	TVE-2
Classification						
rel-amz/item-churn-least100-spending	V	69.72 ± 4.98	61.18 ± 1.07	68.13 ± 1.78	81.57 ± 0.37	**82.76 ± 0.43**
	T	63.01 ± 4.98	52.92 ± 0.79	56.96 ± 2.07	**78.06 ± 0.65**	77.83 ± 0.95
rel-amz/user-churn-top100-spending	V	84.53 ± 1.97	88.44 ± 0.29	88.62 ± 0.38	**91.42 ± 0.07**	90.94 ± 0.14
	T	78.36 ± 1.72	83.86 ± 1.04	81.63 ± 0.46	**86.56 ± 0.36**	85.72 ± 0.38
rel-hm/user-churn-top100-spending	V	69.43 ± 1.56	64.27 ± 0.60	**74.57 ± 0.73**	71.34 ± 0.24	70.09 ± 0.61
	T	53.93 ± 1.46	56.65 ± 0.53	64.22 ± 1.47	**65.33 ± 0.35**	64.15 ± 0.61
Regression						
rel-amz/item-ltv-least100-spending	V	0.1753 ± 0.0005	0.1751 ± 0.0001	0.1755 ± 0.0005	0.1749 ± 0.0011	**0.1746 ± 0.0013**
	T	0.1867 ± 0.0004	0.1865 ± 0.0001	0.1866 ± 0.0007	0.1841 ± 0.0014	**0.1836 ± 0.0013**
rel-hm/item-sales-top200-spending	V	3.6312 ± 0.0118	3.6948 ± 0.0126	3.6524 ± 0.0123	3.7160 ± 0.0140	**3.5519 ± 0.0056**
	T	3.5325 ± 0.0386	3.4925 ± 0.0059	3.5087 ± 0.1426	3.4815 ± 0.0210	**3.4583 ± 0.0320**

C.5 Experiments on Weakly Schema-dependent Tasks

So far, we have only focused on scenarios where tasks are strictly defined by SQL queries. However, some tasks may weakly depend on the schema graph or relational entity graph—such as those based on manual labels or involving database errors. To quantify TVE's sensitivity to the underlying relational entity graph's quality, we simulated these weak dependencies by running controlled "noise" experiments on the RelBench's `rel-amazon` database, using the following steps:

1. We randomly dropped 20% and 40% of the rows in the `review` table to corrupt the relational entity graph. The corrupted graph here is considered clean signals, and tasks depend on this newly created graph.

2. We treat the unaltered database as noisy database; thus, the downstream tasks defined in Step 1 are no longer fully dependent on this noisy database or its original graph structure. Therefore, a robust pre-training paradigm overfitting to the noisy relational entity graph cannot generalize well to the tasks in Step 1.

3. We compared three pre-training paradigms—MAE, CTR, and our TVE—on two tasks `user-churn-top100-spending` and `user-churn`. We then (a) pre-trained our models on the noisy graph (including task vector constructed from this noisy graph), and (b) fine-tuned (and evaluated) on the same noisy graph using labels generated from the clean database (constructed in Step 1). In other words, the only factor that deviates from prior experiments is the downstream labels, which are generated based on a simulated clean database.

Table 7 compares robustness to noises of different SSL paradigms. We can observe that even when the downstream tasks loosely depend on schema graph understanding, and when TVE is pre-trained on a noisy relational database, TVE's task-aware pre-training consistently outperforms standard SSL objectives. This demonstrates that our framework is robust to errors in the schema graph/relational entity graph and retains its effectiveness across a range of noise levels.

Table 7: Performance across different model variants on the scenarios where tasks weakly depend on the relational entity graphs of `rel-amazon`. Reported metric is ROC-AUC.

Task			Baseline	MAE-0.25	CTR-0.25	TVE-1
user-churn-top100-spending	Linear Probing	Drop 20%	75.42 ± 3.10	82.26 ± 0.27	78.98 ± 1.17	86.79 ± 0.26
		Drop 40%	73.95 ± 1.76	78.16 ± 0.46	76.38 ± 1.80	81.44 ± 0.17
	Fine-tuning	Drop 20%	80.72 ± 1.14	82.83 ± 0.41	81.46 ± 2.31	85.04 ± 0.92
		Drop 40%	76.39 ± 1.16	76.97 ± 1.43	75.72 ± 1.26	78.29 ± 1.79
user-churn	Linear Probing	Drop 20%	60.92 ± 0.61	65.62 ± 0.04	64.59 ± 0.01	70.17 ± 0.03
		Drop 40%	61.95 ± 0.67	66.96 ± 0.03	66.14 ± 0.04	71.31 ± 0.01
	Fine-tuning	Drop 20%	70.09 ± 0.15	69.74 ± 0.21	69.98 ± 0.13	70.26 ± 0.10
		Drop 40%	71.35 ± 0.06	70.61 ± 0.24	71.20 ± 0.17	71.37 ± 0.20

Table 8: Comparison of linear probing performance of TVE-1 pretrained on different entity types and other models. Reported metric is ROC-AUC.

Model	item-churn-least100-spending	user-churn-top100-spending
Baseline	63.01 ± 4.98	78.36 ± 1.72
MAE-0.25	52.92 ± 0.79	83.86 ± 1.04
CTR-0.25	56.96 ± 2.07	81.63 ± 0.46
TVE-1 (item)	78.06 ± 0.65	83.28 ± 0.63
TVE-1 (user)	65.88 ± 2.25	86.56 ± 0.36

C.6 In-database Cross-entity Transfers

Since our proposed pre-training approach is entity-type-centric, where the constructed task vector is limited to a certain entity type in the database, it is natural to question the transferability between pre-trained models on two different entity types in the same database. To better understand the separability of the pre-trained latent representation, we further conduct a linear probing experiment, where we investigate if an item-centric pre-trained model can be fine-tuned for a user-task, and vice versa. Table 8 reports the results. TVE-1 (item) refers to the pre-trained model on the item's task vector, while TVE-1 (user) refers to the pre-trained model on the user's task vector. Surprisingly, even when the task vector is not matched with downstream tasks' entity type, it can still yield better performance than other generic SSL methods. For example, TVE-1 (user) is only worse than TVE-1 (item) on `item-churn-least100-spending`, and TVE-1 (item) is as strong as MAE-0.25 on `user-churn-top100-spending`. Consequently, this experiment highlights the necessity to create an entity type-specific pre-training objective to maximize downstream performance gains.

C.7 Experiments on Cell-level Prediction

While previous experiments focus on predictive modeling over the RelBench benchmark [46], there exists other benchmarks such as 4DBInfer [51] where tasks are defined differently. Specifically, RelBench focuses on inference an SQL-generated label attached to rows in the dimension table (the table that is static and does not contain timestamps), while 4DBInfer focuses on cell-level prediction.

We would like to show that our proposed framework is performant regardless of evaluation settings. We pre-trained TVE-1 on `seznam` dataset [42], and evaluated downstream performance on an entity attribute classification task called `charge`, where we would like to predict cells under column `sluzba` of entity table `dobito`. We reported 4 metrics—Accuracy, Macro F1, Micro F1, and Mean Reciprocal Rank (MRR)—on both validation and test splits. We leverage the `AutoCompleteTask` implementation provided by RelBench [46] to create the corresponding training table for cell-level prediction, and we split the dataset such that the cut-off timestamps for the validation set and test set are at 2015-03-31 and 2015-07-31, respectively. Table 9 reports results for both linear probing and fine-tuning settings.

We can observe that the performance of MAE and CTR is not consistent across datasets, while TVE consistantly performs well and exhibits competitive performance (matching MAE for linear probing and matching CTR for full-finetuning). This is because both MAE and CTR have limitations. For example, CTR does not offer a separable representation after pre-training (low performance

Table 9: Results on `seznam/charge`. Split, Validation, and Test are abbreviated as S, V, and T for readability.

Setting	Model	S	Accuracy	Macro F1	Micro F1	MRR
Linear Probing	Baseline	V	60.27 ± 1.22	28.89 ± 2.09	60.27 ± 1.22	76.96 ± 0.70
		T	59.36 ± 1.00	27.99 ± 1.86	59.36 ± 1.00	76.48 ± 0.50
	MAE-0.25	V	71.03 ± 0.04	44.50 ± 1.24	71.03 ± 0.04	84.00 ± 0.03
		T	70.52 ± 0.05	43.32 ± 1.74	70.52 ± 0.05	83.76 ± 0.04
	CTR-0.25	V	42.88 ± 0.09	12.74 ± 0.81	42.88 ± 0.09	64.56 ± 0.10
		T	41.11 ± 0.32	10.01 ± 0.48	41.11 ± 0.32	62.79 ± 0.21
	TVE-1	V	69.98 ± 0.21	44.20 ± 1.00	69.98 ± 0.21	83.44 ± 0.09
		T	69.90 ± 0.16	43.39 ± 0.64	69.90 ± 0.16	83.55 ± 0.08
Fine-tuning	Baseline	V	81.13 ± 0.21	62.02 ± 0.76	81.13 ± 0.21	89.96 ± 0.12
		T	80.97 ± 0.50	63.43 ± 0.83	80.97 ± 0.50	89.91 ± 0.28
	MAE-0.25	V	81.18 ± 0.15	62.00 ± 2.37	81.18 ± 0.15	89.99 ± 0.09
		T	81.08 ± 0.32	63.31 ± 1.00	81.08 ± 0.32	89.97 ± 0.18
	CTR-0.25	V	81.21 ± 0.17	63.28 ± 1.05	81.21 ± 0.17	90.01 ± 0.10
		T	81.31 ± 0.41	63.69 ± 0.92	81.31 ± 0.41	90.10 ± 0.22
	TVE-1	V	81.39 ± 0.22	63.92 ± 1.17	81.39 ± 0.22	90.11 ± 0.12
		T	81.26 ± 0.29	63.82 ± 0.75	81.26 ± 0.29	90.07 ± 0.16

during linear probing), while MAE is not as performant as CTR during full fine-tuning. Moreover, these gains come despite a conceptual mismatch—TVE was optimized for next-state prediction on dimension tables, whereas the downstream tasks operate on fact-table rows. Regardless of evaluation setting, TVE consistently performs well across different evaluation settings.

C.8 Performance Sensitivity Across Splits and Tasks

To further assess model robustness, Fig. 8 presents additional scatterplots for three more tasks, namely `rel-amazon/item-churn-least50-spending`, `rel-amazon/item-churn-least100-spending`, and `rel-hm/user-churn-top50-spending`, with the purpose of complementing the results reported in Fig. 6. Across all three tasks, TVE achieves higher validation and test scores with consistently smaller average distances to the centroid, whereas traditional SSL methods suffer from recency bias, where they exhibit inflated validation performance but degraded test results (e.g. `rel-hm/user-churn-top50-spending`). These findings underscore TVE's stability to hyperparameter choices and the last linear head's weight initialization.

C.9 Complexity Analysis

In this section, we present complexity analysis on task vector construction and provide comparison of pre-training runtimes between our proposed approach and the baseline SSL methods.

C.9.1 Task Vector Construction

First, we provide formal worst-case time complexity of task vector construction, and then provide empirical analysis of task vector construction on `rel-amazon` and `rel-hm` via elapsed time and peak memory usages.

Remark C.1. *Let the number of rows per table be $\mathcal{O}(N)$, the number of timestamps be T, the branching factor of relational entity graph (the input graph for learning) be b, the number of aggregation functions be r, the branching factor of schema graph (the blueprint of the Relational Database) be p, and the number of hops for generating the task vector be k. Then, the time complexity*

of task vector construction is

$$T_{TVE} = \mathcal{O}(rNTp^k b^k).$$

Proof. We would like to compute the computational cost at each step for constructing the task vector. First, we compute the cost taken for a single path from the schema graph perspective (Step 1 - 3), and then sum up to get the final cost (Step 4). Remember that for every schema graph path, there are many paths fall under the same schema graph path since schema graph is just a blueprint of the input graph.

Step 1. Joining with the time table, which is a necessary step regardless of pre-training paradigm, costs $\mathcal{O}(NT)$.

Step 2. Next, every join causes each row to fan out by a factor of $\mathcal{O}(b)$, so the cost of joining up to u successive times is:

$$\mathcal{O}(NTb + NTb^2 + ... + NTb^u) = \mathcal{O}(NT \sum_{i=1}^{u} b^i).$$

For $b > 1$, the geometric sum is scaled on the order of b^u:

$$\sum_{i=1}^{u} b^i = \frac{b^{u+1} - b}{b-1} \approx \frac{b^{u+1}}{b-1} = \frac{b}{b-1} b^u = \Theta(b^u).$$

Therefore, the total cost of for joining up to length u is $\mathcal{O}(NTb^u)$.

Step 3. One aggregation of r functions over the joined rows costs $\mathcal{O}(rNTb^u)$, so every path (from the schema graph perspective) cost $\mathcal{O}(NTb^u + rNTb^u) = \mathcal{O}(rNTb^u)$.

Step 4. There are $\mathcal{O}(p^u)$ paths of length u from the schema graph perspective, summing the cost:

$$T_{\text{TVE}} = \sum_{u=1}^{k} [p^u \times \mathcal{O}(rNTb^u)] = \mathcal{O}\left(rNT \sum_{u=1}^{k} (pb)^u\right) = \mathcal{O}(rNTp^k b^k)$$

□

Therefore, the cost blows up exponentially in k through both p^k and b^k, with linear dependence on r, N, and T. In practice, branching factor of the schema graph p is typically small, and the schema graph is small, leading to small k. Furthermore, a path is only meaningful if it contains at least one fact table (the table containing timestamps), so that we can construct the predictive pre-training objective. Finally, this operation is only done once, and is not involved during pre-training optimization.

To demonstrate our point, we would like to report the time and peak memory usage for constructing task vector across datasets in Table 10. As we can see from the table, this operation is feasible even with large-scale databases, where `rel-amazon` and `rel-hm` contain approximately 24M and 33M rows, respectively. For our experiments, the maximum hop is 2, while the possible aggregation functions include `MEAN`, `MIN`, `MAX`, `SUM`, `COUNT`, and `STDDEV` for numerical values, and `MODE`, `COUNT`, and `COUNT DISTINCT` for categorical values. In industry scenario, such cost is expendable.

C.9.2 Pre-training Runtimes

We present an empirical comparison of per-batch pre-training runtimes (batch size = 512) for all SSL methods. Using a shared RDL backbone, we measure the wall-clock time for a single training step, averaged over 100 runs per model. As shown in Table 11, TVE achieves the lowest overhead, since it computes losses to approximate task vectors only for root-table entities. In contrast, MAE and CTR must compute losses for all nodes in the sampled subgraphs, and CTR incurs additional cost from its double forward passes and pairwise distance calculations.

Table 10: Elapsed time and peak memory usage for different datasets given root tables.

Model	rel-amazon/item	rel-amazon/user	rel-hm/item	rel-hm/user
TVE-1	64.59 seconds / 39.98 GB	90.17 seconds / 40.07 GB	19.38 seconds / 4.62 GB	11.74 seconds / 19.85 GB
TVE-2	382.75 seconds / 40.03 GB	652.85 seconds / 52.67 GB	98.03 seconds / 16.15 GB	871.26 seconds / 148.19 GB

Table 11: Time complexity of different SSL methods for every step in seconds.

Dataset	Root Table	Model			
		MAE	CTR	TVE-1	TVE-2
rel-amazon	user	0.0554 ± 0.0071	0.0587 ± 0.0088	0.0445 ± 0.0102	0.0432 ± 0.0165
	item	0.0583 ± 0.0065	0.0663 ± 0.0059	0.0440 ± 0.0089	0.0478 ± 0.0101
rel-hm	user	0.0610 ± 0.0115	0.0615 ± 0.0062	0.0504 ± 0.0288	0.0466 ± 0.0053
	item	0.0814 ± 0.0075	0.3110 ± 0.0552	0.0609 ± 0.0166	0.0571 ± 0.0038

D Further Discussions

D.1 On Designing Better Approximation of Sufficient Statistics

Designing richer approximations of sufficient statistics is indeed a vital and challenging direction. For example, beyond fixed aggregators like task vectors, we can employ a small neural "set encoder" (e.g., DeepSets [58] or transformer-based pooling) that learns to compress each next-window set into a vector. This approach can, in principle, capture all information in the set without hand-crafting statistics. However, training two networks—one to encode the set and one to predict the label—introduces instability. For instance, a GAN-style formulation (where the encoder acts as a generator and the predictor as a discriminator) risks mode collapse, mapping diverse inputs to the same latent code. Therefore, designing a sophisticated loss function to avoid mode collapse is a challenge for this strategy.

Our current task-vector objective optimizes only one encoder for input graph, which makes training stable and efficient. Despite being simpler, these vectors capture key moments of the distribution and are not susceptible to GAN-style collapse. They provide a reliable surrogate that practitioners can use immediately, while leaving more complex, learnable summaries to future work.

D.2 Sufficiency of Simple Statistics

Computing minimal sufficient representations has long been a grand challenge. Because direct evaluation of mutual information is often intractable, Alemi et al. [6] instead optimize a tractable lower bound, while Belghazi et al. [11] turn to neural estimators to approximate mutual information directly.

Given the above challenges of sufficient statistics approximation, we would like to provide complementary arguments showing why—and under what conditions—simple statistics can suffice, and how we mitigate the information loss when compared with sufficient statistics in practice.

Statistical sufficiency in common distributions. In many distributions, a small collection of aggregators is provably sufficient. Notable examples include:

- The sum of samples is sufficient for exponential, Poisson, and Bernoulli distributions.
- The max is sufficient for a uniform distribution over a bounded interval.
- The mean is sufficient for a normal distribution with known variance.

Whenever the underlying dynamics match one of these canonical families, no information is lost by reducing the full set of observations to the corresponding statistic.

Enriching the task vector to minimize loss. Real-world label-derivation queries often depend on different columns, thus leading to potential different distributions. To guard against this, we augment our task vectors with as many simple statistics—include `MEAN`, `MIN`, `MAX`, `SUM`, `COUNT`, and `STDDEV` for numerical values, and `MODE`, `COUNT`, and `COUNT DISTINCT` for categorical values. This "overcomplete" summary ensures that, even if some statistics are redundant, we capture a broader range of moment and order information, thereby shrinking the worst-case information gap.

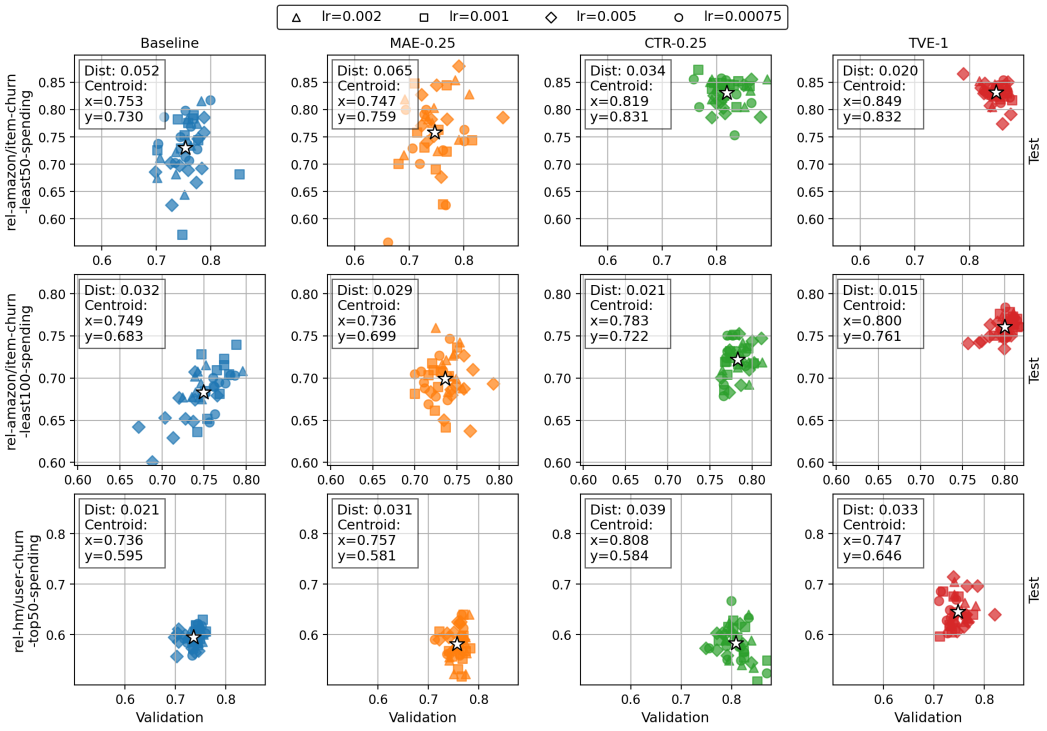


Figure 8: Additional Results for Validation vs. Test scatter plots for low-data node classification tasks across varying learning rates. Distances to centroid (denoted ☆) are used to measure performance variance.

Set-theoretic recovery and downstream expressivity. From a set-theoretic standpoint, certain combinations of simple statistics can perfectly reconstruct small multisets. For instance, if exactly three purchases occur in the prediction window, then the triplet $\{\texttt{MIN}, \texttt{MAX}, \texttt{SUM}\}$ uniquely determines the multiset. More generally, accurate pre-training on these statistics helps the model learn an implicit approximation of the original value-set, which in turn improves its ability to approximate any downstream labeling function—since most such functions operate on those same sets (cf. Eq. 3).

Empirical validation in tabular domains. Prior work on Deep Feature Synthesis (DFS) [34] demonstrates that stacking joins with simple aggregations often matches more expressive graph-based architectures on many benchmarks [51, 53]. While DFS features incur theoretical information loss relative to a full relational graph encoding, their competitive accuracy shows that richly engineered statistics can suffice in practice. Our investigation is orthogonal: we systematically characterize all node-level SQL-generated labels as set functions and show how a task-vector approach can act as surrogate representation for next-window sets. Additionally, our task vector includes similar aggregators to that of DFS.

Together, these points explain why simple statistics are not only theoretically grounded but also practically powerful—and how we effectively minimize any information loss.