A Appendix

A.1 Benchmark Details

In this section, we include the details on the datasets and the tasks in RelBench [42] which we use for our evaluation. RelBench consists of 7 datasets from diverse relational database domains, including e-commerce, clinical records, social networks, and sports, among others. These datasets are curated from their respective source domains and consist a wide range of sizes, from 1.3K to 5.4M records in the training set for the prediction tasks, with a total of 47M training records. For each dataset, multiple predictive tasks are defined, such as predicting a user's engagement with an advertisement within the next four days or determining whether a clinical trial will achieve its primary outcome within the next year. In total, RelBench has 30 tasks across the 7 datasets, covering entity classification, entity regression, and recommendation. For our evaluation, we focus on 21 tasks on entity classification and regression as RelGT primarily serves as a node representation learning model in RDL. We exclude recommendation tasks in this work since they involve specific considerations, such as identifying target nodes [54] or using pair-wise learning architectures [55] and using RelGT trivially in RDL is sub-optimal. We detail the dataset and task statistics in Table 3.

A.1.1 Datasets.

`rel-amazon`. The Amazon E-commerce dataset consists of product details, user information, and review interactions from Amazon's platform, including metadata like pricing and categories, along with review ratings and content.

`rel-avito`. Avito's marketplace dataset contains search queries, advertisement characteristics, and contextual information from this major online trading platform that facilitates transactions across various categories including real estate and vehicles.

`rel-event`. The Event Recommendation dataset from Hangtime mobile app tracks users' social planning, capturing interactions, event details, demographic data, and social connections to reveal how relationships impact user behavior.

`rel-f1`. The F1 dataset provides comprehensive Formula 1 racing information since 1950, documenting drivers, constructors, manufacturers, and circuits with detailed records of race results, standings, and specific data on various racing sessions and pit stops.

`rel-hm`. H&M's dataset contains customer-product interactions from their e-commerce platform, featuring customer demographics, product descriptions, and purchase histories.

`rel-stack`. The Stack Exchange dataset documents activity from this network of Q&A websites, including user biographies, posts, comments, edits, votes, and question relationships where users earn reputation through contributions.

`rel-trial`. The clinical trial dataset from the AACT initiative has study protocols and outcomes, containing trial designs, participant information, intervention details, and results metrics, serving as a key resource for medical research.

A.1.2 Tasks.

The following entity classification and regression tasks are defined in RelBench for the above datasets.

(1) **`rel-amazon`**
   (a) `user-churn`: Predict whether a user will discontinue reviewing products within the next three months.
   (b) `item-churn`: Predict if a product will have no reviews in the next three months.
   (c) `user-ltv`: Estimate the total monetary value of merchandise in dolloar that a user will purchase and review within the next three months.
   (d) `item-ltv`: Estimate the total monetary value of purchases and reviews a product will receive during the next three months.

(2) **`rel-avito`**
   (a) `user-visits`: Predict if a user will engage with several (advertisements) ads within the upcoming four days.
   (b) `user-clicks`: Predict whether a user will interact with multiple ads through clicking within the upcoming four days.
   (c) `ad-ctr`: Estimate the interaction probability for an ad, assuming it receives an interaction within four days.

(3) **`rel-event`**
   (a) `user-attendance`: Estimate the number of of events a user will confirm attendance to (RSVP yes or maybe) within the upcoming seven days.
   (b) `user-repeat`: Predict whether a user will join an event (RSVP yes or maybe) within the upcoming seven days, provided they attended in an event during the previous fourteen days.
   (c) `user-ignore`: Predict whether a user will disregard or ignore more than two events invitations within the upcoming seven days.

(4) **`rel-f1`**
   (a) `driver-dnf`: Predict if a driver will not finish a race within the upcoming month.
   (b) `driver-top3`: Determine if a driver will achieve a top-three qualifying position in a race within the upcoming month.
   (c) `driver-position`: Estimate a driver's average finishing placement across all races in the upcoming two months.

(5) **`rel-hm`**
   (a) `user-churn`: Predict whether a customer will not perform any transactions in the upcoming week.
   (b) `item-sales`: Estimate total revenue generated by a product in the upcoming week.

(6) **`rel-stack`**
   (a) `user-engagement`: Predict whether a user will contribute through voting, posting, or commenting within the upcoming three months.
   (b) `user-badge`: Predict whether a user will secure a new badge within the upcoming three months.
   (c) `post-votes`: Estimate the number of votes a user's post will accumulate over the upcoming three months.

(7) **`rel-trial`**
   (a) `study-outcome`: Predict whether a clinical trial will achieve its principal outcome within the upcoming year.

Table 3: Dataset and task statistics from RelBench used for our evaluation.

Dataset	Task	Task type	#Rows of training table			#Unique	%train/test
			Train	Validation	Test	Entities	Entity Overlap
rel-amazon	user-churn	classification	4,732,555	409,792	351,885	1,585,983	88.0
	item-churn	classification	2,559,264	177,689	166,842	416,352	93.1
	user-ltv	regression	4,732,555	409,792	351,885	1,585,983	88.0
	item-ltv	regression	2,707,679	166,978	178,334	427,537	93.5
rel-avito	user-clicks	classification	59,454	21,183	47,996	66,449	45.3
	user-visits	classification	86,619	29,979	36,129	63,405	64.6
	ad-ctr	regression	5,100	1,766	1,816	4,997	59.8
rel-event	user-repeat	classification	3,842	268	246	1,514	11.5
	user-ignore	classification	19,239	4,185	4,010	9,799	21.1
	user-attendance	regression	19,261	2,014	2,006	9,694	14.6
rel-f1	driver-dnf	classification	11,411	566	702	821	50.0
	driver-top3	classification	1,353	588	726	134	50.0
	driver-position	regression	7,453	499	760	826	44.6
rel-hm	user-churn	classification	3,871,410	76,556	74,575	1,002,984	89.7
	item-sales	regression	5,488,184	105,542	105,542	105,542	100.0
rel-stack	user-engagement	classification	1,360,850	85,838	88,137	88,137	97.4
	user-badge	classification	3,386,276	247,398	255,360	255,360	96.9
	post-votes	regression	2,453,921	156,216	160,903	160,903	97.1
rel-trial	study-outcome	classification	11,994	960	825	13,779	0.0
	study-adverse	regression	43,335	3,596	3,098	50,029	0.0
	site-success	regression	151,407	19,740	22,617	129,542	42.0

(b) `study-adverse`: Estimate the number of patients who will experience significant adverse effects or mortality in a clinical trial over the upcoming year.
(c) `site-success`: Estimate the success rate of a clinical trial site in the upcoming year.

A.2 Node initialization for Subgraph GNN PE in RelGT

As described in Section 3.1, we employ a lightweight GNN PE to capture local graph structures that cannot be represented by other elements of the token, particularly the parent-child relationships among nodes in the local subgraph. The GNN is implemented as:

$$h_{\text{pe}}(v_j) = \text{GNN}(A_{\text{local}}, Z_{\text{random}})_j \in \mathbb{R}^d \tag{9}$$

where $\text{GNN}(\cdot,\cdot)_j$ is a lightweight GNN applied to the local subgraph, yielding the encoding for node v_j. Here, $A_{\text{local}} \in \mathbb{R}^{K\times K}$ represents the adjacency matrix of the sampled subgraph containing K nodes, and $Z_{\text{random}} \in \mathbb{R}^{K\times d_{\text{init}}}$ denotes randomly initialized node features for the GNN (with d_{init} as the initial feature dimension). In RelGT, we set $d_{\text{init}} = 1$.

The randomly initialized node features (Z_{random}) provide enhanced properties as discussed in Section 3.1. We investigate the alternative approach of using Laplacian PE (Z_{LapPE}) computed over the subgraph instead of random initialization and report these results in Table 4. For these results, we utilized a positional encoding dimension size of 4. Our findings indicate that Z_{LapPE} consistently underperforms compared to Z_{random}, while also introducing additional computational overhead ranging from 1.02× to 3.38× across the 8 selected tasks in our study. This shows the challenges of using existing PEs such as Laplacian PE in relational entity graphs and signify the use of GNN PE as part of RelGT's tokenization strategy.

A.3 HGT Baseline

In the main experiments (Section 4), we use the Heterogeneous Graph Transformer (HGT) [20] as a graph transformer (GT) baseline, and report results for two variants to demonstrate the advantages of RelGT over existing GT models. Specifically, we consider the standard HGT model and an enhanced version, HGT+PE, which incorporates Laplacian positional encodings (LapPE). These positional encodings are computed on sampled subgraphs rather than the full graph.

For implementation, we use the `HGTConv` layer from PyTorch Geometric [14] and integrate it into the RDL pipeline [42] by replacing the default GNN module. Both variants use 4 attention heads and 2 layers, similar to the configuration of the GNN module in RDL, with residual connections and layer normalization applied between layers. For the HGT+PE variant, we use LapPE of dimension 4 for all tasks, except for `rel-amazon` `item-ltv` and `rel-hm` `item-sales`, where we use dimension 2. Notably, because the relational entity graphs are heterogeneous, the Laplacian positional encodings is computed multiple times for each node type, unlike the original homogeneous setting for which LapPE was designed [10].

Table 4: Study of node initialization in Subgraph GNN PE. Relative drop is expressed as percentage drop of using Z_{LapPE} vs. Z_{random} and runtime ratio compares the time for Z_{LapPE} vs. Z_{random}.

Dataset	Task (# train)	MAE ↓	Performance Z_{random}	Performance Z_{LapPE}	% Rel Drop	Epoch time (m) Z_{random}	Epoch time (m) Z_{LapPE}	Runtime Ratio
rel-avito	ad-ctr	Test	**0.035**	0.0369	-5.43	0.76	2.57	3.38
		Val	0.0314	0.0314				
rel-trial	site-success	Test	**0.326**	0.3452	-5.89	32.88	36.09	1.1
		Val	0.359	0.3683				
rel-hm	item-sales	Test	**0.0536**	0.0573	-6.9	49.26	53.8	1.09
		Val	0.0627	0.0667				

Dataset	Task (# train)	AUC ↑	Performance Z_{random}	Performance Z_{LapPE}	% Rel Drop	Epoch time (m) Z_{random}	Epoch time (m) Z_{LapPE}	Runtime Ratio
rel-avito	user-clicks	Test	**0.607**	0.583	-3.95	6.42	7.43	1.16
		Val	0.656	0.6564				
	user-visits	Test	**0.664**	0.6626	-0.21	9.26	10.50	1.13
		Val	0.699	0.7002				
rel-event	user-ignore	Test	**0.8**	0.7988	-0.15	1.85	2.77	1.5
		Val	0.881	0.8916				
rel-trial	study-outcome	Test	**0.674**	0.6532	-3.09	1.41	1.52	1.08
		Val	0.689	0.6719				
rel-amazon	user-churn	Test	0.7039	**0.7044**	0.07	168.00	170.55	1.02
		Val	0.7036	0.7036				

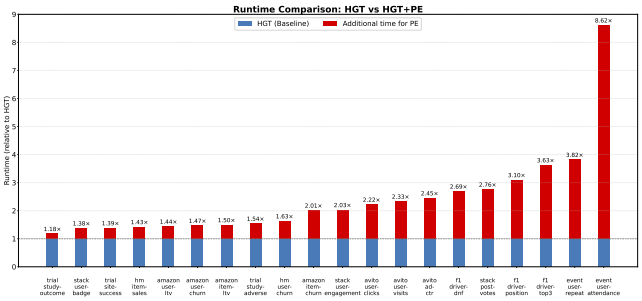

Figure 5: Runtime Comparison of HGT and HGT+PE baseline. Adding the Laplacian Positional Encoding increases computational overhead, with penalties on average training time per epoch. The overhead for PE reaches up to 761% relative to the training time of HGT on the same dataset.

In addition to the main results in Table 1, we report per-epoch runtimes in Figure 5 and Table 5. We observe a significant computational overhead from precomputing Laplacian positional encodings, with slowdowns ranging from 1.8× to 8.62×, highlighting the challenge of directly applying existing graph PE techniques *as is* to relational entity graphs, and signifying the contributions of RelGT.

A.4 Detailed Results

In Table 6, we report the full results of different configurations we tuned for RelGT, particularly on the smaller datasets with lesser than a million training nodes. Table 7 provides the full scores for the RelGT component study in Table 2, while Table 8 provides the supporting results for Figure 4. Finally, we provide the elaborated version of the Tables 1a and 1b in Tables 9 and 9, respectively.

Table 5: Relative performance drop (%) when position encoding (PE) is removed from HGT+PE models and average training time per epoch of HGT and HGT+PE. Negative scores suggest the PE is critical, and vice-versa. HGT+PE consistently requires more training time per epoch compared to HGT without PE across all datasets.

Dataset	Task	No PE	HGT(s)	HGT+PE(s)
rel-f1	driver-position	1.79	1.47	4.56
rel-avito	ad-ctr	10.73	1.63	4.00
rel-event	user-attendance	−2.85	4.36	37.57
rel-trial	study-adverse	−2.03	9.72	15.02
rel-trial	site-success	1.29	45.73	63.41
rel-amazon	user-ltv	3.45	73.59	106.21
rel-amazon	item-ltv	−0.93	73.68	110.33
rel-stack	post-votes	0.15	191.23	528.25
rel-hm	item-sales	−2.18	94.66	135.05
rel-f1	driver-dnf	0.46	2.54	6.84
rel-f1	driver-top3	−23.39	0.38	1.38
rel-avito	user-clicks	3.08	11.09	24.66
rel-avito	user-visits	−1.24	17.16	40.07
rel-event	user-repeat	1.93	1.35	5.16
rel-event	user-ignore	2.29	4.49	651.10
rel-trial	study-outcome	−0.21	4.09	4.83
rel-amazon	user-churn	0.29	78.56	115.53
rel-amazon	item-churn	−0.20	75.51	152.06
rel-stack	user-engagement	0.52	175.16	356.07
rel-stack	user-badge	1.57	153.68	212.21
rel-hm	user-churn	4.34	77.73	127.04
	Average	−0.05	52.28	128.64

A.5 Resource Information.

We implement RelGT using PyTorch framework [39], PyTorch Geometric framework [14] and adapt the codebase of relational deep learning [42] https://github.com/snap-stanford/relbench. All our experiments are conducted on an NVIDIA A100 GPU server with 8 GPU nodes.

Table 6: RelGT results using $L \in 1, 4, 8$ and dropout $\in 0.3, 0.4, 0.5$ for the smaller datasets with less than a million training nodes.

Dataset	Task (# train)	MAE ↓	L1 0.3	L1 0.4	L1 0.5	L4 0.3	L4 0.4	L4 0.5	L8 0.3	L8 0.4	L8 0.5
rel-f1	driver-position (7k)	Test	4.942	5.6431	**3.917**	4.6316	4.0851	4.0042	5.5273	5.5569	4.6085
		Val	3.1897	3.1817	3.3257	3.1046	3.3352	3.1276	3.1589	3.2907	3.1843
rel-avito	ad-ctr (5k)	Test	0.0358	0.0352	**0.0345**	0.035	0.0366	0.038	0.0354	0.0358	0.0356
		Val	0.0322	0.0313	0.0314	0.0314	0.0322	0.0335	0.0317	0.0322	0.0324
rel-event	user-attendance (19k)	Test	0.2635	0.2595	0.2635	**0.2502**	0.2543	0.2584	0.2635	0.2637	0.2635
		Val	0.2618	0.2558	0.2618	0.2548	0.2534	0.253	0.2618	0.2599	0.2618
rel-trial	study-adverse (43k)	Test	44.8553	44.2260	44.848	44.8893	44.4310	**43.9923**	44.2245	44.5878	44.5013
		Val	46.3538	46.3193	46.2056	46.1031	45.9498	46.2148	46.1804	46.1381	46.4332
	site-success (151k)	Test	0.3490	0.3652	0.3830	0.4019	0.386	**0.3262**	0.3783	0.3431	0.3644
		Val	0.3493	0.3455	0.3550	0.3771	0.392	0.3593	0.3848	0.3643	0.3669
Dataset	**Task (# train)**	**AUC ↑**	**L1 0.3**	**L1 0.4**	**L1 0.5**	**L4 0.3**	**L4 0.4**	**L4 0.5**	**L8 0.3**	**L8 0.4**	**L8 0.5**
rel-f1	driver-dnf (11k)	Test	0.7434	0.7587	0.7521	**0.7587**	0.745	0.6957	0.7349	0.7393	0.741
		Val	0.6877	0.6761	0.6896	0.6804	0.6762	0.6768	0.6702	0.6803	0.6865
	driver-top3 (1k)	Test	0.7845	0.8203	0.8	0.8171	0.8157	**0.8352**	0.7871	0.8217	0.8222
		Val	0.7775	0.783	0.7764	0.7841	0.79	0.7958	0.7893	0.7847	0.7829
rel-avito	user-clicks (59k)	Test	0.6524	0.6233	0.6212	0.6067	0.5893	0.596	0.6245	**0.683**	0.6507
		Val	0.6649	0.6616	0.6501	0.6564	0.6608	0.6579	0.6587	0.6649	0.6648
	user-visits (86k)	Test	0.6627	0.6663	0.665	0.6615	0.6584	0.6642	0.6647	**0.6678**	0.664
		Val	0.7005	0.6993	0.7001	0.6954	0.6958	0.699	0.6995	0.7024	0.7011
rel-event	user-repeat (3k)	Test	0.6981	0.7403	0.7452	0.7563	0.7236	0.7432	**0.7609**	0.7316	0.7418
		Val	0.7172	0.7386	0.7319	0.7245	0.7207	0.736	0.7285	0.7209	0.7064
	user-ignore (19k)	Test	0.8006	0.802	0.7986	0.799	0.787	0.8002	0.7956	0.8076	**0.8157**
		Val	0.8739	0.8721	0.8729	0.878	0.8731	0.881	0.8757	0.8801	0.8868
rel-trial	study-outcome (11k)	Test	0.6808	0.6753	0.6837	0.6488	0.6818	0.6744	**0.6861**	0.6562	0.6649
		Val	0.6815	0.6792	0.6751	0.6737	0.676	0.689	0.6678	0.6746	0.6768

Table 7: Relative drop (%) in performance in RelGT after removing a model component. Negative scores suggest the component is critical in RelGT, and vice-versa.

Dataset	Task (# train)	MAE ↓	RelGT (Full)	RelGT (No Global)	% Rel. Drop	RelGT (No GNN)	% Rel. Drop	RelGT (No Type)	% Rel. Drop	RelGT (No Hop)	% Rel. Drop	RelGT (No Time)	% Rel. Drop
rel-avito	ad-ctr	Test	**0.0350**	0.0371	-6.0	0.0354	-1.14	0.0375	-7.14	0.0362	-3.43	0.0382	-9.14
		Val	0.0314	0.0323		0.0315		0.0328		0.0322		0.0337	
rel-trial	site-success	Test	**0.3262**	0.3882	-19.01	0.3561	-9.17	0.3356	-2.88	0.3963	-21.49	0.3285	-0.71
		Val	0.3593	0.3342		0.3637		0.3655		0.3614		0.3615	
rel-hm	item-sales	Test	0.0536	0.0586	-9.33	0.0629	-17.35	0.0604	-12.69	**0.0531**	0.93	0.095	-77.24
		Val	0.0627	0.0676		0.073		0.0696		0.0623		0.1025	
Dataset	**Task (# train)**	**AUC ↑**	**RelGT (Full)**	**RelGT (No Global)**	**% Rel. Drop**	**RelGT (No GNN)**	**% Rel. Drop**	**RelGT (No Type)**	**% Rel. Drop**	**RelGT (No Hop)**	**% Rel. Drop**	**RelGT (No Time)**	**% Rel. Drop**
rel-avito	user-clicks	Test	0.6067	0.6543	7.85	0.5148	-15.15	0.6371	5.01	0.6417	5.77	**0.6575**	8.37
		Val	0.6564	0.6496		0.6551		0.6559		0.6482		0.6579	
	user-visits	Test	0.6642	0.6619	-0.35	0.6484	-2.38	0.6635	-0.11	**0.6668**	0.39	0.6592	-0.75
		Val	0.699	0.6892		0.6879		0.6991		0.7016		0.7005	
rel-event	user-ignore	Test	0.8002	0.7898	-1.3	0.8012	0.12	0.7993	-0.11	**0.8055**	0.66	0.7995	-0.09
		Val	0.881	0.8575		0.8637		0.8873		0.8852		0.8789	
rel-trial	study-outcome	Test	0.6744	0.66	-2.14	0.6628	-1.72	**0.6996**	3.74	0.6715	-0.43	0.6911	2.48
		Val	0.689	0.664		0.6775		0.6728		0.6705		0.6578	
rel-amazon	user-churn	Test	0.7039	0.6994	-0.64	0.6984	-0.78	**0.705**	0.16	0.7043	0.06	0.6884	-2.2
		Val	0.7036	0.6994		0.6994		0.7042		0.704		0.6882	

Table 8: Ablation of context size K in RelGT.

Dataset	Task (# train)	MAE ↓	RelGT K=100	RelGT K=300	RelGT K=500
rel-avito	ad-ctr	Test	0.0375	0.0374	**0.0351**
		Val	0.0329	0.0319	0.031
rel-trial	site-success	Test	0.3739	**0.3674**	0.3842
		Val	0.3708	0.372	0.376
rel-hm	item-sales	Test	0.055	0.0532	**0.052**
		Val	0.0643	0.0619	0.061

Dataset	Task (# train)	AUC ↑	RelGT K=100	RelGT K=300	RelGT K=500
rel-avito	user-clicks	Test	**0.6628**	0.6491	0.6334
		Val	0.6437	0.6622	0.6632
	user-visits	Test	**0.6664**	0.6653	0.6627
		Val	0.7013	0.701	0.7005
rel-event	user-ignore	Test	0.7674	**0.8105**	0.8068
		Val	0.8682	0.8853	0.8843
rel-trial	study-outcome	Test	**0.7078**	0.6526	0.666
		Val	0.6575	0.663	0.6877
rel-amazon	user-churn	Test	0.7038	**0.7054**	0.7043
		Val	0.7033	0.7044	0.7042

Table 9: Results on the entity regression tasks in RelBench. Lower is better. Best values are in bold. Relative gains are expressed as percentage improvement over RDL baseline.

Dataset	Task	MAE ↓	RDL Baseline	HGT	HGT +PE	RelGT (ours)	% Rel. Gain
rel-f1	driver-position	Test	4.022	4.1598	4.2358	**3.9170**	2.61
		Val	3.193	3.3517	2.9894	3.3257	
rel-avito	ad-ctr	Test	0.041	0.0441	0.0494	**0.0345**	15.85
		Val	0.037	0.0409	0.0456	0.0314	
rel-event	user-attendance	Test	0.258	0.2635	0.2562	**0.2502**	2.79
		Val	0.255	0.2617	0.2574	0.2548	
rel-trial	study-adverse	Test	44.473	43.3253	**42.4622**	43.9923	1.08
		Val	46.290	45.9957	45.7966	46.2148	
	site-success	Test	0.400	0.4374	0.4431	**0.3263**	18.43
		Val	0.401	0.4198	0.4245	0.3593	
rel-amazon	user-ltv	Test	14.313	15.3804	15.9296	**14.2665**	0.32
		Val	12.132	13.1017	13.5599	12.1151	
	item-ltv	Test	50.053	56.1384	55.6211	**48.9222**	2.26
		Val	45.1401	51.2139	50.3468	43.8161	
rel-stack	post-votes	Test	**0.065**	0.0679	0.0680	**0.0654**	-0.62
		Val	0.059	0.0617	0.0618	0.0592	
rel-hm	item-sales	Test	0.056	0.0655	0.0641	**0.0536**	4.29
		Val	0.065	0.0749	0.0735	0.0627	

Table 10: Results on the entity classification tasks in RelBench. Higher is better. Best values are in bold. Relative gains are expressed as percentage improvement over RDL baseline.

Dataset	Task	AUC ↑	RDL Baseline	HGT	HGT +PE	RelGT (ours)	% Rel. Gain
rel-f1	driver-dnf	Test	0.7262	0.7142	0.7109	**0.7587**	4.48
		Val	0.7136	0.7678	0.7318	0.6804	
	driver-top3	Test	0.7554	0.6389	0.8340	**0.8352**	10.56
		Val	0.7764	0.6659	0.6079	0.7958	
rel-avito	user-clicks	Test	0.6590	0.6584	0.6387	**0.6830**	3.64
		Val	0.6473	0.5977	0.5656	0.6649	
	user-visits	Test	0.6620	0.6426	0.6507	**0.6678**	0.88
		Val	0.6965	0.6696	0.6732	0.7024	
rel-event	user-repeat	Test	**0.7689**	0.6717	0.6590	0.7609	-1.04
		Val	0.7125	0.6247	0.5974	0.7285	
	user-ignore	Test	0.8162	**0.8348**	0.8161	0.8157	-0.06
		Val	0.9170	0.8896	0.8940	0.8868	
rel-trial	study-outcome	Test	0.6860	0.5679	0.5691	**0.6861**	0.01
		Val	0.6818	0.5985	0.5925	0.6678	
rel-amazon	user-churn	Test	**0.7042**	0.6608	0.6589	0.7039	-0.04
		Val	0.7045	0.6639	0.6622	0.7036	
	item-churn	Test	**0.8281**	0.7824	0.7840	0.8255	-0.31
		Val	0.8239	0.7845	0.7846	0.8220	
rel-stack	user-engagement	Test	0.9021	0.8898	0.8852	**0.9053**	0.35
		Val	0.9059	0.8914	0.8847	0.9033	
	user-badge	Test	**0.8986**	0.8652	0.8518	0.8632	-3.94
		Val	0.8886	0.8760	0.8691	0.8741	
rel-hm	user-churn	Test	**0.6988**	0.6773	0.6491	0.6927	-0.87
		Val	0.7042	0.6814	0.6502	0.6988	