# APPENDIX

## A CREATION DETAILS AND STATISTICS OF TAXOKG-BENCH

Figures 2a-2f show the concept, entity and relation frequency histograms on six aligned TaxoKGs, where x-axis tick "*#m-n*" denotes the frequency bins ranges from $m$ to $n$, and y-axis denotes the proportion of cases that falls into each bin. "* × ReVerb" in Figure 2 captions indicates that histograms are produced on the three AutoTAXOs aligned with the particular OpenKG constructed from *ReVerb*. Similarly, "* × ReVerb" indicates that histograms are produced on the three AutoTAXOS aligned with *OPIEC*. *MSCG* × *ReVerb* and *MSCG* × *OPIEC* are two large-scale TaxoKGs containing billions knowledge triplets of before filtering. Therefore, we set high thresholds for them. In particular, concepts with at least 20 grounded entities are kept in both MSCG × ReVerb and *MSCG* × *OPIEC* datasets, while entities with frequency greater than or equal to 40, 25 are kept in *MSCG* × *ReVerb* and *MSCG* × *OPIEC*, respectively. For relation, frequencies greater than or equal to 35, 3 are kept. Nevertheless, the remaining knowledge triplets are still in million scales, which makes the evaluation on these two *Taxo-KG*s very slow. We then conduct further down-samplings to build lightweight yet diverse testbeds. Similarly, we set the concept threshold, entity threshold, relation threshold for *SEMedical* aligned and *SEMusic* aligned *Taxo-KG*s as $\{3, 2, 2\}$ and $\{3, 4, 3\}$, respectively.

In Table 5, the columns *#entity*, *#concept* and *#pair* denote number of unique entities, concepts and entity-concept pairs reside in AutoTAXO part, while *#mention*, *#relation* and *#triplet* denote number of subject/object mentions, relation and subject-relation-object triplets reside in OpenKG part.

## B SUMMARY OF EXISTING RELATION-GCN MODELS

The message passing functions of existing relational-GCN models can be viewed in Table 7. $\boldsymbol{h_u}, \boldsymbol{h_r}, \boldsymbol{h_v}$ denotes embeddings of source node $u$, relation $r$ and target node $v$ (message receiver). $\boldsymbol{W}$, $\boldsymbol{W_r}$, $\boldsymbol{W_{dir(r)}}$ denotes learnable weight matrices for all relations, each relation and each relation directions. $\boldsymbol{W_s}$ is a learnable weight matrix for self-loop edges. $\boldsymbol{\alpha_r}$ is a learnable weight scalar for each relation. For KBGAT, $[\cdot \parallel \cdot]$ denote vector concatenation operation. For CompGCN, $\phi$ is defined as composition operators.

## C IMPLEMENTATION DETAILS

**atan2 function.** The atan2 function used § 5 in Eq. (7) is defined as follows:

$$\operatorname{atan2}(y,x) = \begin{cases} \arctan(\frac{y}{x}) & if\ x > 0, \\ \arctan(\frac{y}{x}) + \pi & if\ x < 0\ and\ y \geq 0, \\ \arctan(\frac{y}{x}) - \pi & if\ x < 0\ and\ y < 0, \\ \frac{\pi}{2} & if\ x = 0\ and\ y > 0, \\ -\frac{\pi}{2} & if\ x = 0\ and\ y < 0, \\ 0 & if\ x = 0\ and\ y = 0. \end{cases} \tag{13}$$

**LtCaG model.** Learn-to-Conceptualize-and-Generalize (LtCaG) model is a non-parametric model serving as one baseline method for TaxoKG completion task. The inference process of LtCaG can be explicitly depicted in the following equations:

$$\begin{aligned} &\mathcal{P}(< dog, CapableOf, bark >) = \\ &\quad 0.5 * \frac{\sum_v \mathcal{P}(< v, isA, dog >)\mathcal{P}(< v, CapableOf, bark >)}{\sum_v \mathcal{P}(< v, isA, dog >)} \\ &\quad + 0.5 * \frac{\sum_v \mathcal{P}(< dog, isA, v >)\mathcal{P}(< v, CapableOf, bark >)}{\sum_v \mathcal{P}(< dog, isA, v >)} \end{aligned} \tag{14}$$

$$\begin{aligned} &\mathcal{P}(< papillon, isA, dog >) = \\ &\quad 0.5 * \frac{\sum_{e,v} \mathcal{P}(< papillon, e, v >)\mathcal{P}(< dog, e, v >)}{\sum_{e,v} 1 - (1 - \mathcal{P}(< papillon, e, v >))(1 - \mathcal{P}(< dog, e, v >))} \\ &\quad + 0.5 * \frac{\sum_{e,v} \mathcal{P}(< v, e, papillon >)\mathcal{P}(< v, e, dog >)}{\sum_{e,v} 1 - (1 - \mathcal{P}(< v, e, papillon >))(1 - \mathcal{P}(< v, e, dog >))} \end{aligned} \tag{15}$$

**Hyperparameters for Baselines and HakeGCN** We implement HakeGCN using PyTorch[3] and DGL[4]. For compared methods, implementations are either from original authors (HAKE[5], CompGCN[6]) or dedicated replication (TransE, DistMult, HolE, R-GCN). We optimize HakeGCN and baselines through the Adam or RAdam [20] optimizer with learning rate $lr \in \{$1e-3, 3e-4, 1e-4$\}$ chosen by hyperparameter tuning on validation sets. For regularization, we choose an $l2$ penalty on all learnable parameters except PReLU layers and bias in fully-connected layers, with weights $C_{l2} \in \{0,$ 5e-5$\}$. Other hyperparameters include: token embedding size ($\{200, 300, 500\}$), entity and relation embedding size ($\{200, 500, 600, 800, 1000\}$), dropout ratio ($\{0.1, 0.3, 0.5\}$), negative sampling size ($\{1, 8, 32, 64, 128, 256\}$), batch size ($\{128, 256, 512, 1024\}$), epoch size ($\{200, 400, 800, 1200\}$).

For HakeGCN specific hyperparameters, we select the margin $\gamma$ in Eq. (11) from $\{5, 8, 9, 10, 12\}$, and the temperature $\alpha$ in Eq. (12) from $\{0.5, 1.0, 1.5\}$. We use 2 GCN layers for the GCN encoder module to balance more high-order evidence with the over-smoothing issue, and we set all GCN layers embedding sizes the same as the entity and relation embedding sizes. For the taxonomy-based neighborhood sampling, the weight of keeping neighbors on the taxonomy is chosen from $\{1, 1.5, 2.0, 3.0, 5.0\}$, while the weight of keeping other neighbors is 1.

## D CASE STUDIES OF NEIGHBORS FOR PREDICTING CONCEPTS AND RELATIONS

To have a more intuitive sense about the mutual enhancement of taxonomy and KG for TaxoKG completion, we show some examples of the neighbors used by HakeGCN in the AutoTAXO concept prediction task (Table 8) and the OpenKG relation prediction task (Table 9), where the check mark "✓", the question mark "?", or the cross mark indicate neighbors are beneficial, neutral or harmful for the prediction task, and "-" denotes concept itself serving as subject or object in the corresponding KG triplet. As we expect, neighbors from the taxonomy are mostly helpful for predicting the KG relations, and vice versa. For instance, when predicting

[3] PyTorch: https://pytorch.org/
[4] DGL: https://www.dgl.ai/
[5] HAKE: https://github.com/MIRALab-USTC/KGE-HAKE
[6] CompGCN: https://github.com/malllabiisc/CompGCN

**Table 5: Statistics of the six datasets in TaxoKG-Bench.**

| Dataset | # entity | # concept | # pair | # mention | # predicate | # triplet |
|---|---|---|---|---|---|---|
| MSCG × ReVerb | 5.6/1.0/3.6(K) | 1.8/0.5/1.4(K) | 6.4/1.2/4.0(K) | 12.8/3.8/7.0(K) | 10.3/2.2/4.8(K) | 59.7/3.7/11.2(K) |
| SEMedical × ReVerb | 256/48/163 | 261/131/219 | 256/48/163 | 7.3/1.3/2.9(K) | 6.1/0.9/2.3(K) | 21.3/1.3/4.0(K) |
| SEMusic × ReVerb | 412/76/262 | 335/229/283 | 412/76/262 | 7.5/2.1/4.1(K) | 8.9/1.7/3.7(K) | 41.2/2.6/7.7(K) |
| MSCG × OPIEC | 6.3/1.1/4.0(K) | 1.8/0.6/1.4(K) | 7.6/1.4/4.8(K) | 5.5/1.8/3.2(K) | 3.2/0.4/0.9(K) | 51.2/3.2/9.6(K) |
| SEMedical × OPIEC | 238/44/151 | 256/136/209 | 238/44/151 | 1432/255/564 | 508/75/199 | 2239/176/499 |
| SEMusic × OPIEC | 443/81/282 | 363/256/305 | 443/82/282 | 3.6/1.2/2.3(K) | 1.4/0.3/0.6(K) | 15.9/1.5/3.9(K) |

**Table 6: Percentages of unseen entities, concepts and relations in the testing set of the six datasets.**

| Dataset | Unseen Ent | Unseen Cept | Unseen Rel |
|---|---|---|---|
| MSCG × ReVerb | 24.7% | 39.6% | 8.8% |
| SEMedical × ReVerb | 14.4% | 11.4% | 15.5% |
| SEMusic × ReVerb | 3.6% | 3.2% | 11.4% |
| MSCG × OPIEC | 47.3% | 30.0% | 39.8% |
| SEMedical × OPIEC | 18.1% | 9.6% | 15.1% |
| SEMusic × OPIEC | 4.0% | 0.7% | 6.0% |

**Table 7: Summary of message passing functions in existing relational-GCN models.**

| Model | Message Passing Function |
|---|---|
| R-GCN [28] | $\boldsymbol{W_r h_u} + \boldsymbol{W_s h_v}$ |
| KBGAT [23] | $\boldsymbol{W}[\boldsymbol{h_v} \parallel \boldsymbol{h_u} \parallel \boldsymbol{h_r}]$ |
| SCAN [30] | $\boldsymbol{W} \alpha_r \boldsymbol{h_u} + \boldsymbol{W_s h_v}$ |
| VR-GCN [45] | $\boldsymbol{W}((\boldsymbol{h_v} - \boldsymbol{h_r}) + (\boldsymbol{h_u} + \boldsymbol{h_r}))$ |
| CompGCN [38] | $\boldsymbol{W}_{dir(r)} \phi(\boldsymbol{h_u}, \boldsymbol{h_r})$ |

**Table 8: KG neighbors used in taxonomy concept prediction.**

| Concept | KG Neighbors |
|---|---|
| technique | (make from, recycled material, -) ✓ |
| | (architecture, be a thing of, -) ✓ |
| | (-, be apply, biology) ✓ |
| | (-, mean of, expression) ✗ |
| disease | (-, have reach, epidemic proportion) ✓ |
| | (-, can be treat in, a number of way) ✓ |
| | (two, die of, -) ✓ |
| | (alcohol, can cause, -) ✓ |
| rock music | (-, be about, attitude) ✓ |
| | (-, will start, a new era) ? |
| | (-, be a style of, music) ? |
| | (videos, recently tag with, -) ✗ |

**Table 9: Taxonomy neighbors used in KG relation prediction.**

| Relation | Taxonomy Neighbors |
|---|---|
| be marry to | control ✗, family name ✓, guest ? |
| die from | illness ✓, disease ✓, disorder ✓ |
| listen to | work of art ?, musical work ✓, piece of music ✓ |

the concept *disease* in Table 8, neighbors *(-, have reach, epidemic proportion)*, *(two, die of, -)*, and *(alcohol, can cause, -)* are supporting the correct prediction, although the neighbor *(-, can be treat in, a number of way)* may introduce some confusing evidence. On the other side, concepts *illness, disease, disorder* are helpful for predicting the relation *die from*. Therefore, the case studies clearly support our key insight about the mutual enhancement, and they shed a light on the future direction to investigate how to distinguish helpful and harmful neighbors towards further enhanced TaxoKG completion.

## E EFFICIENCY EVALUATION

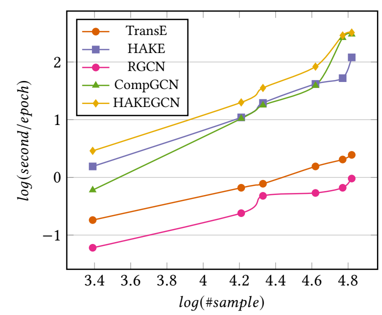


**Figure 6: Model efficiency comparison in log scale.**

We implement HakeGCN and all compared methods in Python and execute them on a server with two 48 cores Intel Xeon CPUs (768GB RAM), using one NVIDIA GeForce GTX 1080 Ti GPU (each with 24GB RAM). Figure 6 shows the runtimes of different models under various training sample sizes. HakeGCN shares similar time-complexity with HAKE and CompGCN. Although TransE and RGCN are more efficient, their performances are far from satisfactory. The slight extra time cost of HakeGCN is introduced by neighbor information aggregation and population, polar coordinate projection, and graph sampling.