SparseTransX: Efficient Training of Translation-Based Knowledge Graph Embeddings Using Sparse Matrix Operations

Md Saidul Hoque Anik, Ariful Azad

MLSys 2025 (Poster #7)

Outline

1. Background
2. Training KG Embeddings
3. Sparse Formulation of TransE
    a. Benefits of Sparse Formulation
    b. Extension to other models
4. SparseTransX Framework
5. Experiments and Results
    a. Speedups and Memory Optimization
    b. Accuracy

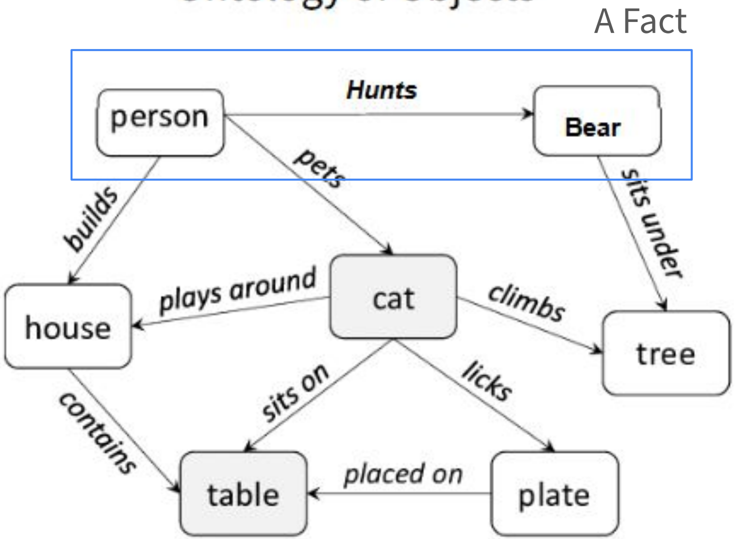
Knowledge Graph

- A specialized graph structure to express ontology
- Directed graph with nodes as entities and edge as relations
- Stored as list of triplets

Head	Rel	Tail
Human	hunts	Bear
Bear	isA	Animal
Human	pets	Cat
...	...	...

- Becoming increasingly popular due to the rise of large language models (LLMs) and usage in GraphRAG for context

Ontology of Objects



Knowledge Graphs

Motivation

Neural Networks

MLP, CNN, RNN, LLM

↓

Matrix Multiplication

Motivation

Neural Networks

MLP, CNN, RNN, LLM

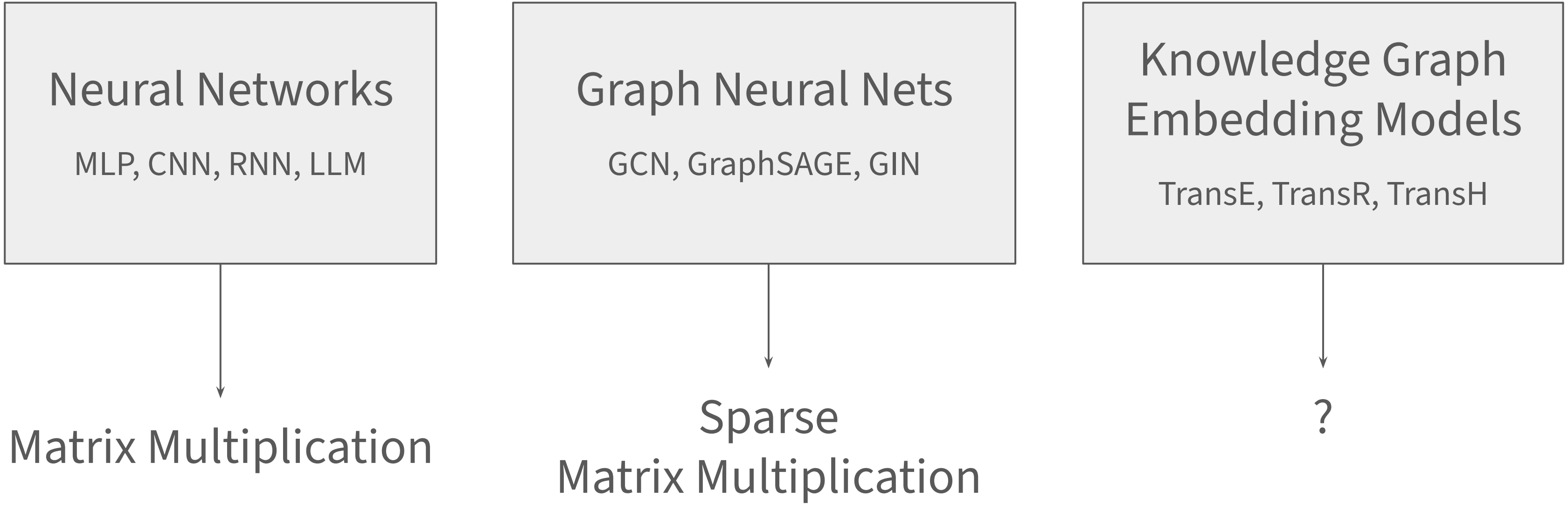
→ Matrix Multiplication

Graph Neural Nets

GCN, GraphSAGE, GIN

→ Sparse Matrix Multiplication

Motivation


Neural Networks
MLP, CNN, RNN, LLM
Graph Neural Nets
GCN, GraphSAGE, GIN
Knowledge Graph Embedding Models
TransE, TransR, TransH
Matrix Multiplication
Sparse Matrix Multiplication
?

Knowledge Graph Embedding Training

Training Dataset

Head	Rel	Tail
Human	hunts	Bear
Bear	isA	Animal
Human	pets	Cat
...	...	...

Knowledge Graph Embedding Training

Training Dataset

Head	Rel	Tail
Human	hunts	Bear
Bear	isA	Animal
Human	pets	Cat
...	...	...

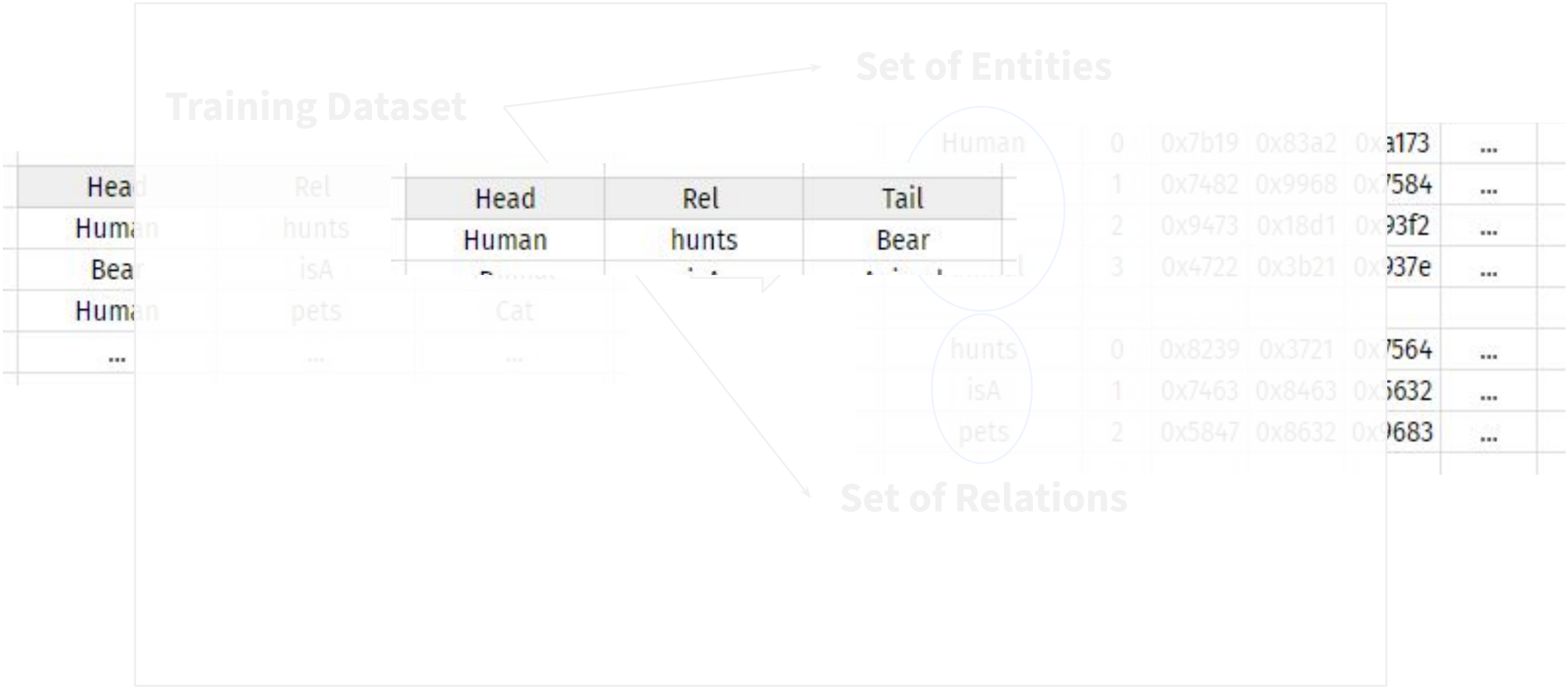
Set of Entities

Human	0	0x7b19	0x83a2	0xa173	...
Bear	1	0x7482	0x9968	0x7584	...
Cat	2	0x9473	0x18d1	0x93f2	...
Animal	3	0x4722	0x3b21	0x937e	...

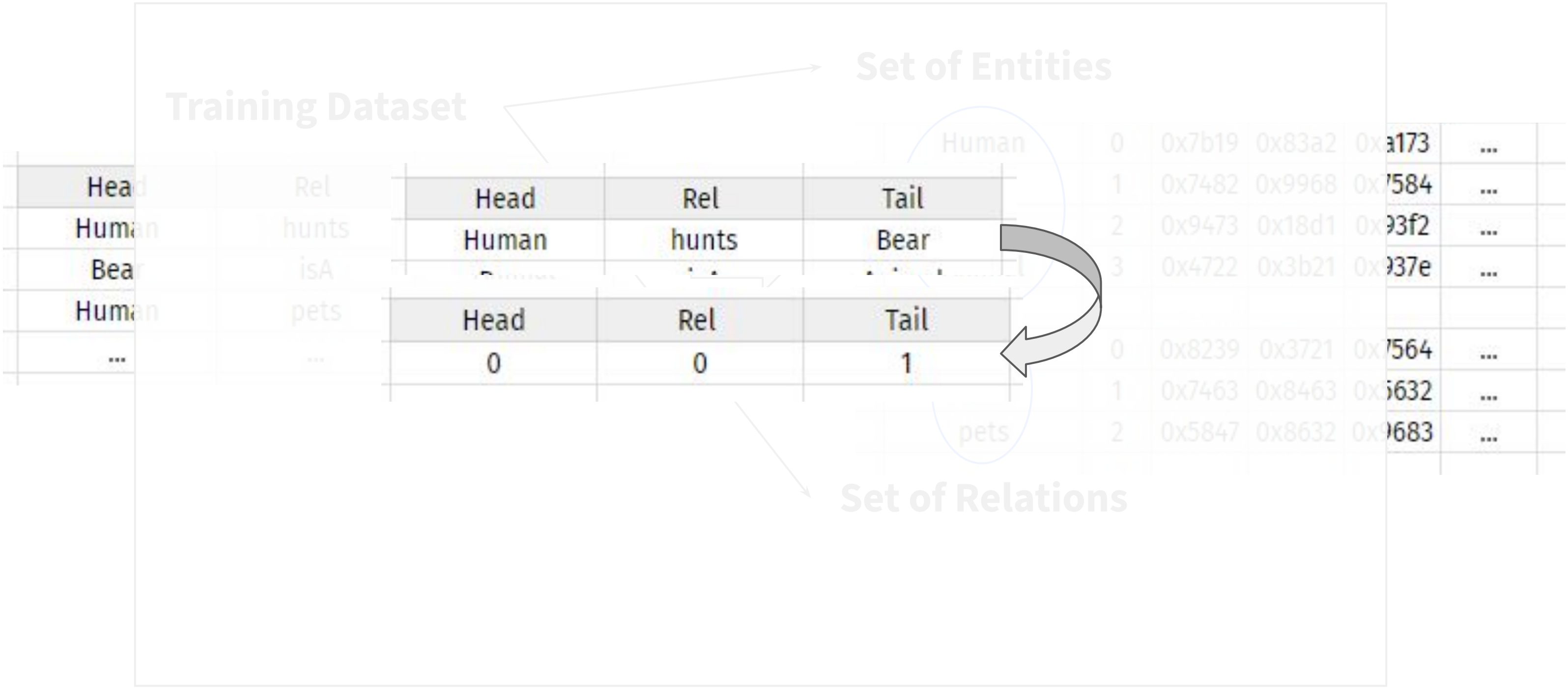
hunts	0	0x8239	0x3721	0x7564	...
isA	1	0x7463	0x8463	0x5632	...
pets	2	0x5847	0x8632	0x9683	...

Set of Relations

Knowledge Graph Embedding Training



Knowledge Graph Embedding Training

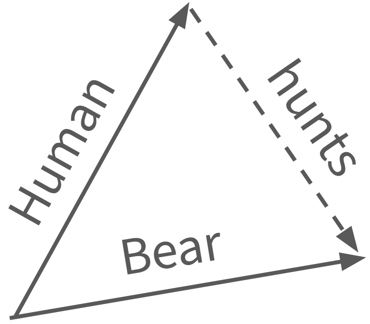


KG Embedding Training (TransE)

Bordes et al., 2013

TransE score function:

$$\forall(\vec{h}, \vec{t}, \vec{r}) \in U,$$
$$\vec{h} + \vec{r} \approx \vec{t}$$
$$\implies \vec{h} + \vec{r} - \vec{t} \approx \vec{0}$$

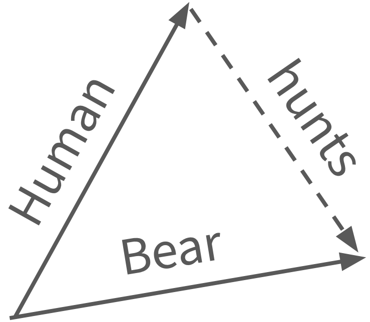


KG Embedding Training (TransE)

Bordes et al., 2013

TransE score function:

$$\forall(\vec{h},\vec{t},\vec{r}) \in U,$$
$$\vec{h}+\vec{r}\approx\vec{t}$$
$$\implies \vec{h}+\vec{r}-\vec{t}\approx\vec{0}$$



Head	Rel	Tail
0	0	1

Embedding Lookup

Ent/Rel	ID	Vector Embeddings			
Human	0	0x7b19	0x83a2	0xa173	...
Bear	1	0x7482	0x9968	0x7584	...
Cat	2	0x9473	0x18d1	0x93f2	...
Animal	3	0x4722	0x3b21	0x937e	...
hunts	0	0x8239	0x3721	0x7564	...
isA	1	0x7463	0x8463	0x5632	...
pets	2	0x5847	0x8632	0x9683	...

KG Embedding Training (TransE)

Bordes et al., 2013

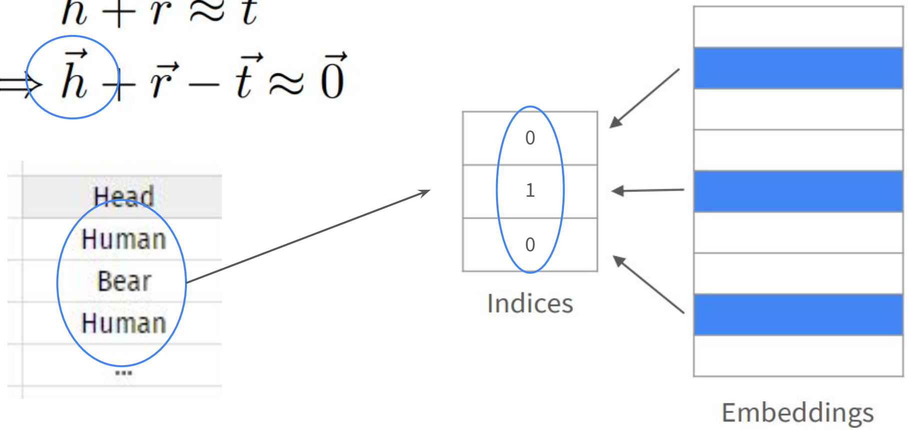
TransE score function:

$$\forall(\vec{h}, \vec{t}, \vec{r}) \in U,$$

$$\vec{h} + \vec{r} \approx \vec{t}$$

$$\implies \vec{h} + \vec{r} - \vec{t} \approx \vec{0}$$

Forward Propagation
Gather Operation



KG Embedding Training (TransE)

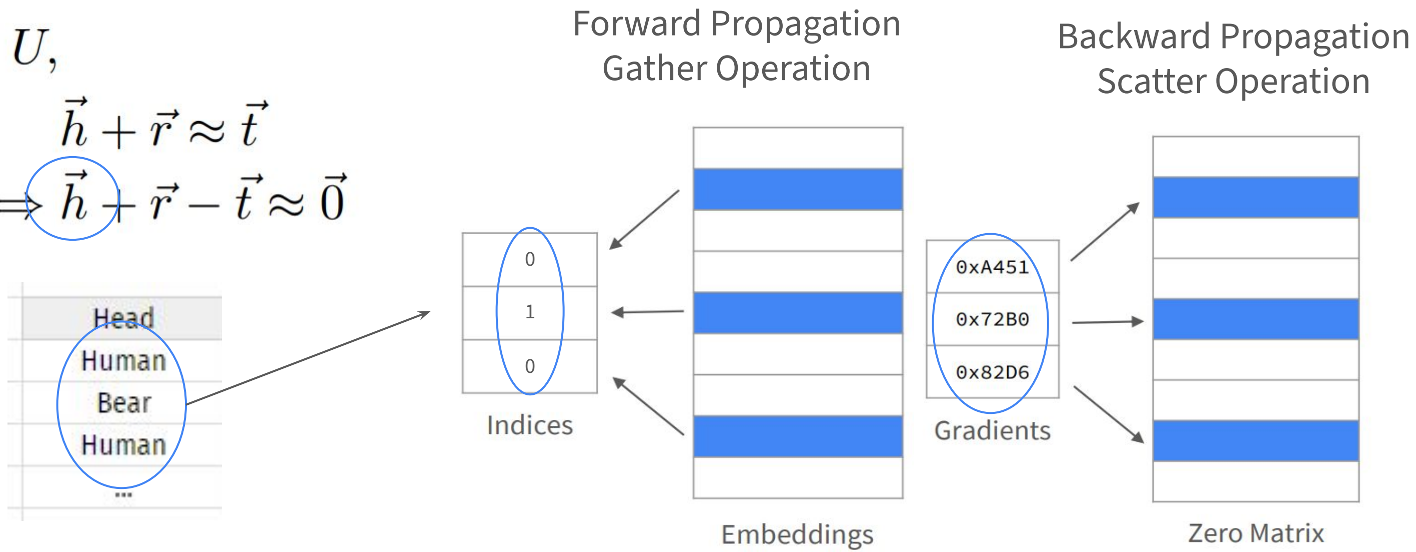
Bordes et al., 2013

TransE score function:

$$\forall(\vec{h}, \vec{t}, \vec{r}) \in U,$$

$$\vec{h} + \vec{r} \approx \vec{t}$$

$$\Longrightarrow \vec{h} + \vec{r} - \vec{t} \approx \vec{0}$$

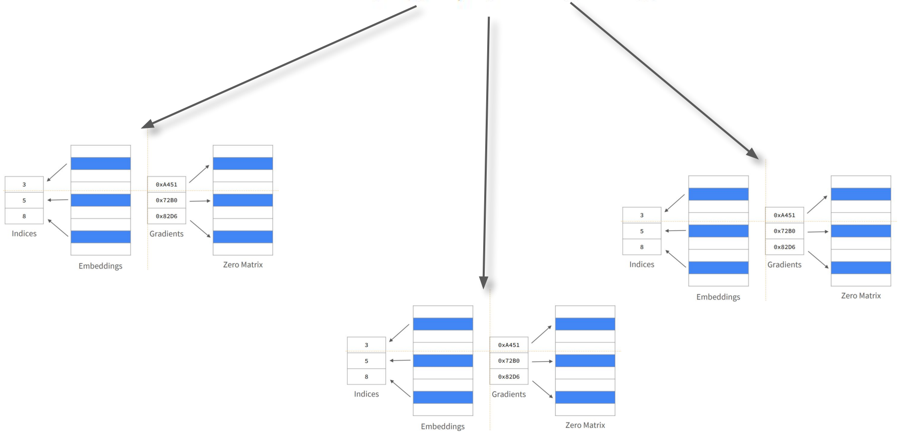


KG Embedding Training (TransE)

Bordes et al., 2013

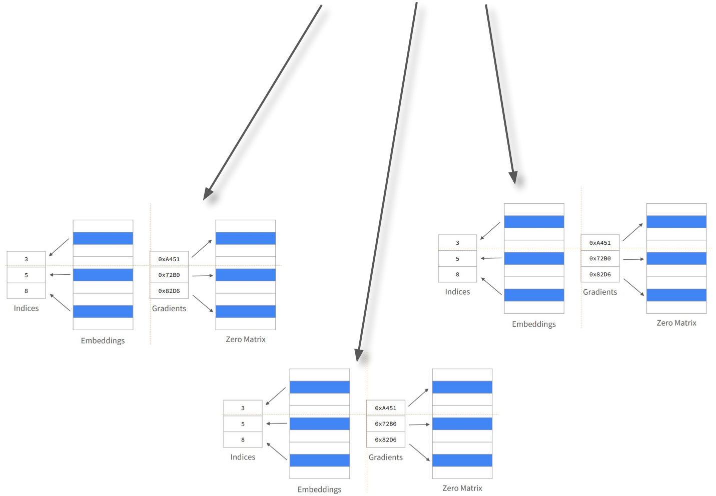
Positive score:

$$\forall(\vec{h},\vec{t},\vec{r}) \in U,$$
$$\vec{h}+\vec{r} \approx \vec{t}$$
$$\implies \vec{h}+\vec{r}-\vec{t} \approx \vec{0}$$

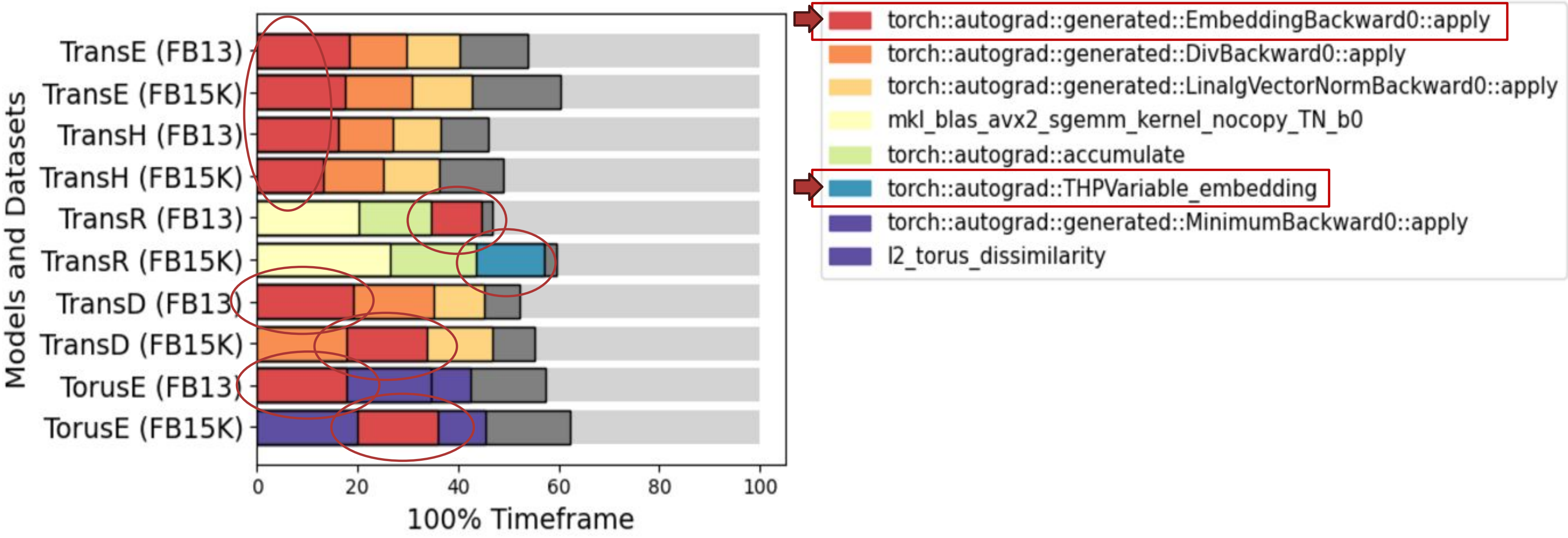


Negative score:

$$\forall(\vec{h},\vec{t},\vec{r}) \in U,$$
$$\vec{h}_1+\vec{r} \approx \vec{t}_1$$
$$\implies \vec{h}_1+\vec{r}-\vec{t}_1 \approx \vec{0}$$

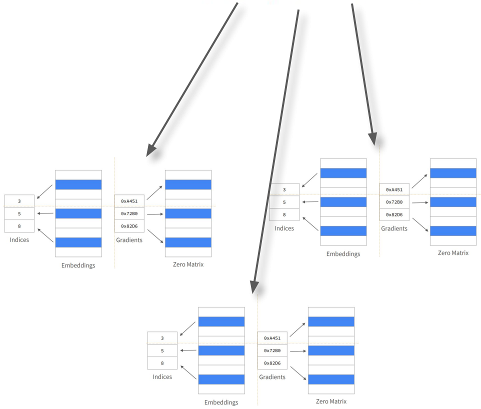


Training Bottleneck



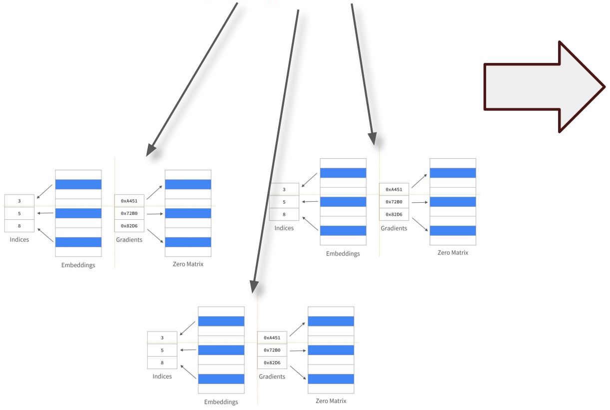
Sparse Matrix Formulation (Our Approach)

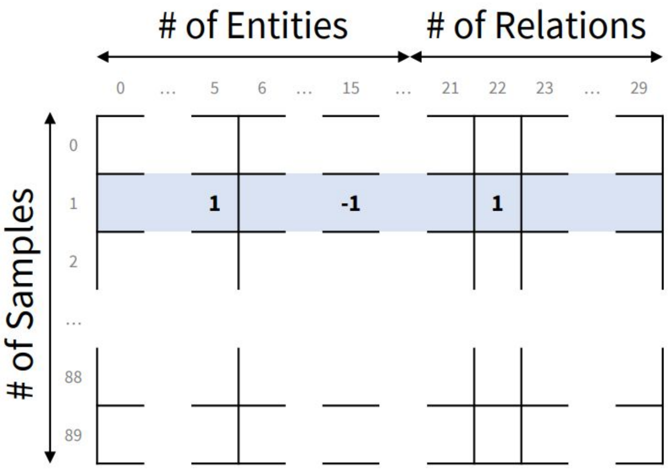
$$\forall(\vec{h}, \vec{t}, \vec{r}) \in U,$$
$$\vec{h} + \vec{r} \approx \vec{t}$$
$$\implies \vec{h} + \vec{r} - \vec{t} \approx \vec{0}$$



Sparse Matrix Formulation (Our Approach)

$$\forall(\vec{h},\vec{t},\vec{r}) \in U,$$
$$\vec{h}+\vec{r} \approx \vec{t}$$
$$\implies \vec{h}+\vec{r}-\vec{t} \approx \vec{0}$$

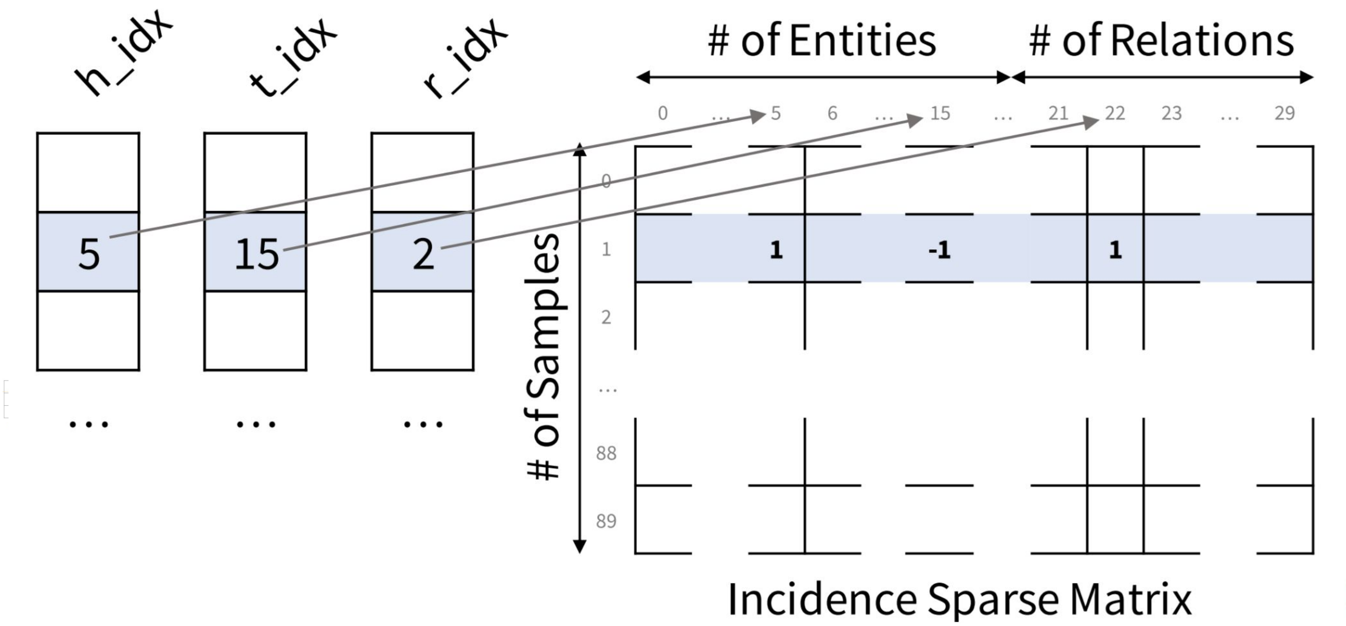




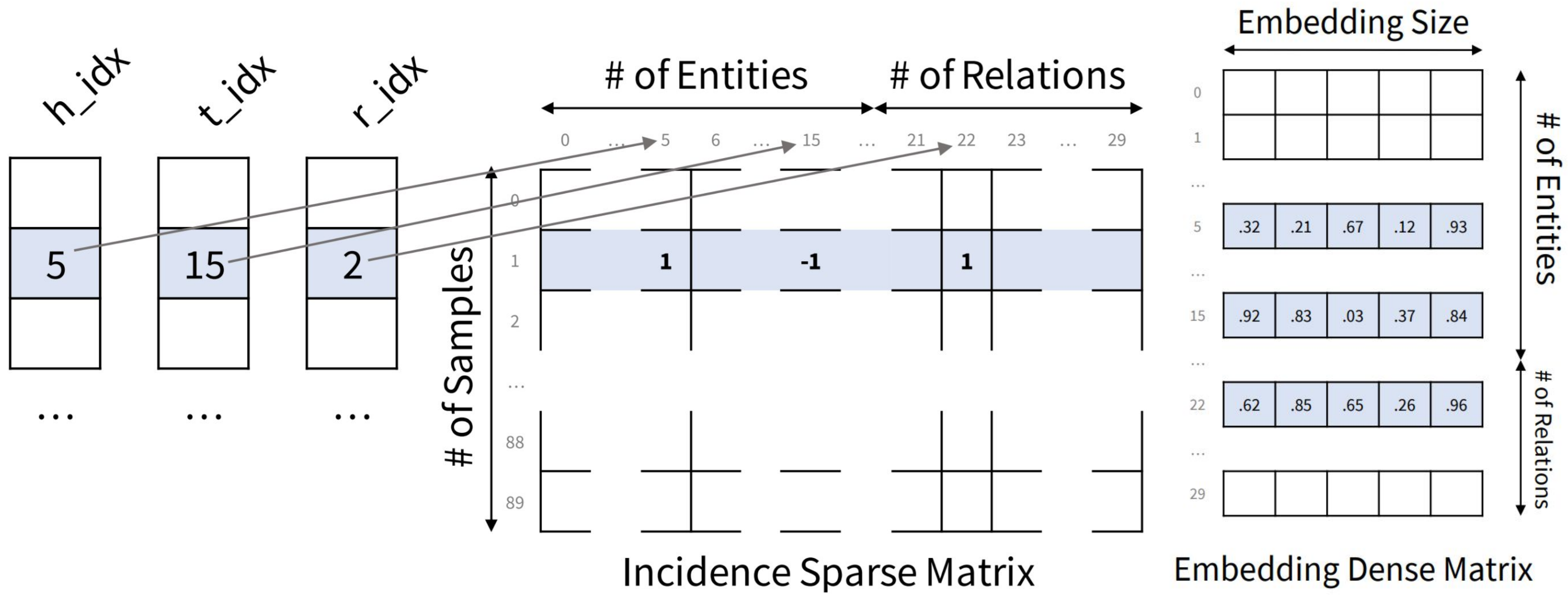
Incidence Sparse Matrix

Sparse Matrix Formulation (Our Approach)

$$\vec{h} + \vec{r} - \vec{t} \approx \vec{0}$$



Sparse Matrix Formulation (Our Approach)

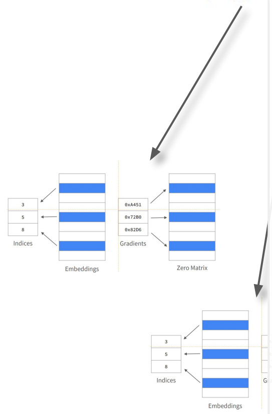


Sparse Matrix Formulation (Our Approach)

$\forall(\vec{h},\vec{t},\vec{r}) \in U,$

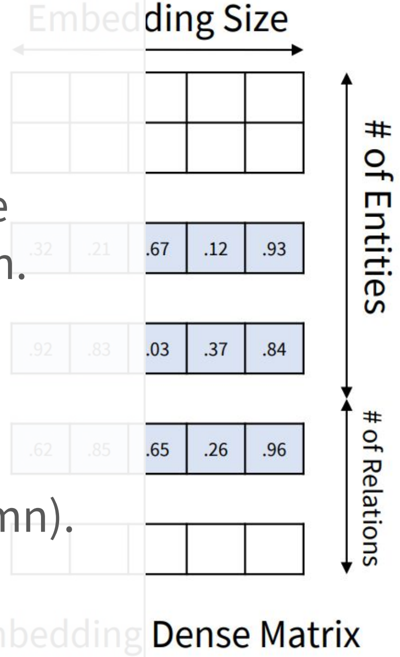
$$\vec{h} + \vec{r} \approx \vec{t}$$

$$\implies \vec{h} + \vec{r} - \vec{t} \approx 0$$



Benefits

- ✓ Reduced usage of intermediate variables.
- ✓ Many high-performance SpMM available. Can use hardware acceleration such as SIMD vectorization.
    - ✓ Backward computation is also an SpMM
- ✓ Matrix remains sparse even for very dense graph.
- ✓ Sparsity pattern is very regular. Complexity is O(mn).
- ✓ Not a square matrix. Can split horizontally.



Comparison

Comparison: Sparse vs. Non-sparse

```python
def non_sparse_approach(h_idx, t_idx, r_idx):
    h = ent_emb[h_idx]
    t = ent_emb[t_idx]
    r = rel_emb[r_idx]
    _score = h + r - t
    return norm(_score)
```

Comparison

Comparison: Sparse vs. Non-sparse

```python
def non_sparse_approach(h_idx, t_idx, r_idx):
    h = ent_emb[h_idx]
    t = ent_emb[t_idx]
    r = rel_emb[r_idx]
    _score = h + r - t
    return norm(_score)

def sparse_approach(adj_mat):
    _score = spmm(adj_mat, all_emb)
    return norm(_score)
```

Generalization

Other Translational Models

Model	Scoring Function
TransE (Bordes et al., 2013)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransH (Wang et al., 2014)	$\|\|\vec{h_\perp} + \vec{d_r} - \vec{t_\perp}\|\|$
TransR (Lin et al., 2015)	$\|\|M_r\vec{h} + \vec{r} - M_r\vec{t}\|\|$
TorusE (Ebisu & Ichise, 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransA (Xiao et al., 2015)	$\|\vec{h} + \vec{r} - \vec{t}\|^T W_r \|\vec{h} + \vec{r} - \vec{t}\|$
TransC (Lv et al., 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|_2^2$
TransM (Fan et al., 2014)	$w_r\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
DistMult (Yang et al., 2014)	$\|\|\vec{h} \odot \vec{r} \odot \vec{t}\|\|$
ComplEx (Trouillon et al., 2016)	$\|\|\vec{h} \odot \vec{r} \odot \bar{\vec{t}}\|\|$
RotatE (Sun et al., 2019)	$\|\|\vec{h} \odot \vec{r} - \vec{t}\|\|$

Generalization

Other Translational Models

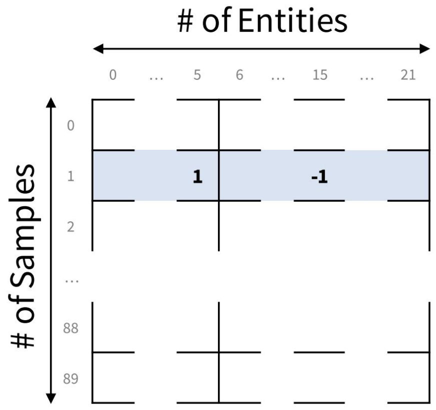
Model	Scoring Function
TransE (Bordes et al., 2013)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransH (Wang et al., 2014)	$\|\|\vec{h_\perp} + \vec{d_r} - \vec{t_\perp}\|\|$
TransR (Lin et al., 2015)	$\|\|M_r\vec{h} + \vec{r} - M_r\vec{t}\|\|$
TorusE (Ebisu & Ichise, 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransA (Xiao et al., 2015)	$\|\vec{h} + \vec{r} - \vec{t}\|^T W_r \|\vec{h} + \vec{r} - \vec{t}\|$
TransC (Lv et al., 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|_2^2$
TransM (Fan et al., 2014)	$w_r\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
DistMult (Yang et al., 2014)	$\|\|\vec{h} \odot \vec{r} \odot \vec{t}\|\|$
ComplEx (Trouillon et al., 2016)	$\|\|\vec{h} \odot \vec{r} \odot \bar{\vec{t}}\|\|$
RotatE (Sun et al., 2019)	$\|\|\vec{h} \odot \vec{r} - \vec{t}\|\|$

Generalization

Other Translational Models

Model	Scoring Function
TransE (Bordes et al., 2013)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransH (Wang et al., 2014)	$\|\|\vec{h_\perp} + \vec{d_r} - \vec{t_\perp}\|\|$
TransR (Lin et al., 2015)	$\|\|M_r\vec{h} + \vec{r} - M_r\vec{t}\|\|$
TorusE (Ebisu & Ichise, 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransA (Xiao et al., 2015)	$\|\vec{h} + \vec{r} - \vec{t}\|^T W_r \|\vec{h} + \vec{r} - \vec{t}\|$
TransC (Lv et al., 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|_2^2$
TransM (Fan et al., 2014)	$w_r\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
DistMult (Yang et al., 2014)	$\|\|\vec{h} \odot \vec{r} \odot \vec{t}\|\|$
ComplEx (Trouillon et al., 2016)	$\|\|\vec{h} \odot \vec{r} \odot \bar{\vec{t}}\|\|$
RotatE (Sun et al., 2019)	$\|\|\vec{h} \odot \vec{r} - \vec{t}\|\|$

Solution: Only compute (h-t) using SpMM



Incidence Sparse Matrix

Generalization

Other Translational Models

Model	Scoring Function
TransE (Bordes et al., 2013)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransH (Wang et al., 2014)	$\|\|\vec{h_\perp} + \vec{d_r} - \vec{t_\perp}\|\|$
TransR (Lin et al., 2015)	$\|\|M_r\vec{h} + \vec{r} - M_r\vec{t}\|\|$
TorusE (Ebisu & Ichise, 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransA (Xiao et al., 2015)	$\|\vec{h} + \vec{r} - \vec{t}\|^T W_r \|\vec{h} + \vec{r} - \vec{t}\|$
TransC (Lv et al., 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|_2^2$
TransM (Fan et al., 2014)	$w_r\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
DistMult (Yang et al., 2014)	$\|\|\vec{h} \odot \vec{r} \odot \vec{t}\|\|$
ComplEx (Trouillon et al., 2016)	$\|\|\vec{h} \odot \vec{r} \odot \bar{\vec{t}}\|\|$
RotatE (Sun et al., 2019)	$\|\|\vec{h} \odot \vec{r} - \vec{t}\|\|$

Generalization

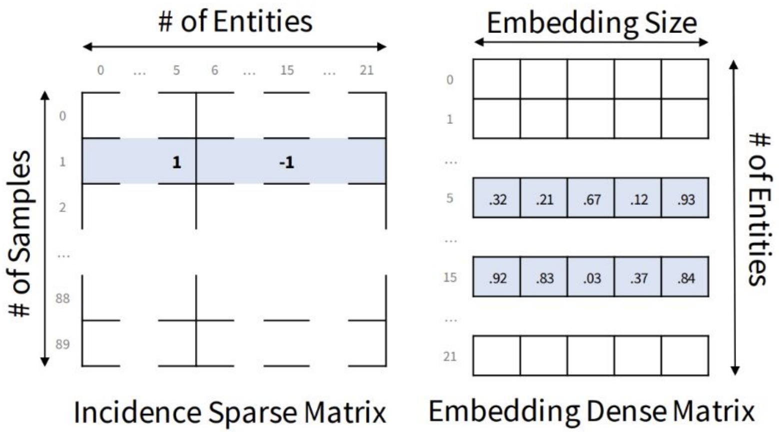
Other Translational Models

Model	Scoring Function
TransE (Bordes et al., 2013)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransH (Wang et al., 2014)	$\|\|\vec{h_\perp} + \vec{d_r} - \vec{t_\perp}\|\|$
TransR (Lin et al., 2015)	$\|\|M_r\vec{h} + \vec{r} - M_r\vec{t}\|\|$
TorusE (Ebisu & Ichise, 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
TransA (Xiao et al., 2015)	$\|\vec{h} + \vec{r} - \vec{t}\|^T W_r \|\vec{h} + \vec{r} - \vec{t}\|$
TransC (Lv et al., 2018)	$\|\|\vec{h} + \vec{r} - \vec{t}\|\|_2^2$
TransM (Fan et al., 2014)	$w_r\|\|\vec{h} + \vec{r} - \vec{t}\|\|$
DistMult (Yang et al., 2014)	$\|\|\vec{h} \odot \vec{r} \odot \vec{t}\|\|$
ComplEx (Trouillon et al., 2016)	$\|\|\vec{h} \odot \vec{r} \odot \bar{\vec{t}}\|\|$
RotatE (Sun et al., 2019)	$\|\|\vec{h} \odot \vec{r} - \vec{t}\|\|$

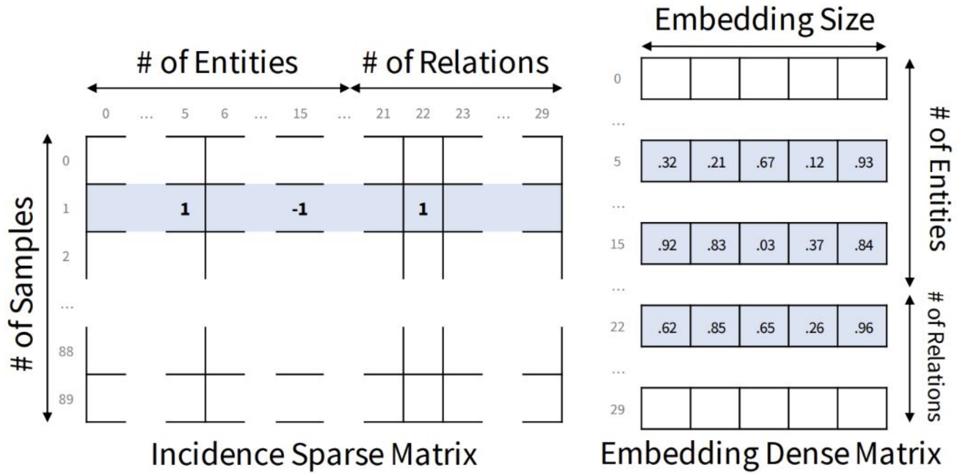
Solution: Use Semiring

Generalization

(h-t) and (h+r-t) computation



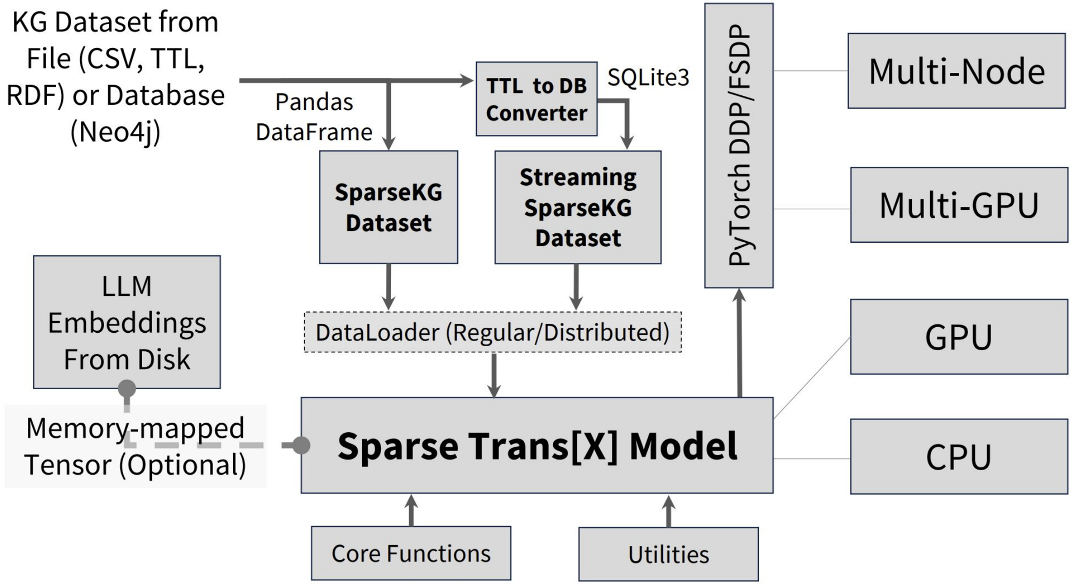
(a) $(h-t)$ Computation. All cells in the highlighted row are zero except h-idx and t-idx. For the highlighted row, h-idx = 5, t-idx = 15, entity-count = 22.



(b) $(h+r-t)$ Computation. All cells in the highlighted row are zero except h-idx, t-idx, and r-idx + entity-count. For the highlighted row, h-idx = 5, t-idx = 15, entity-count = 20, r-idx = 2.

SparseTransX Framework

https://github.com/HipGraph/**SpTransX**



Experimental Setup

Models

Training configuration for CPU and GPU

MODEL	EMBEDDING	BATCH
TRANSE	1024	12×32768
TORUSE	1024	12×32768
TRANSR	128	2×32768
TRANSH	ENT=128,REL=128	32768

Training Loop (Single CPU/GPU)

1. 200 epochs
2. Learning Rate: 0.0004
3. Dissimilarity: L2
4. MarginRankingLoss
   a. Margin: 0.5

Frameworks:

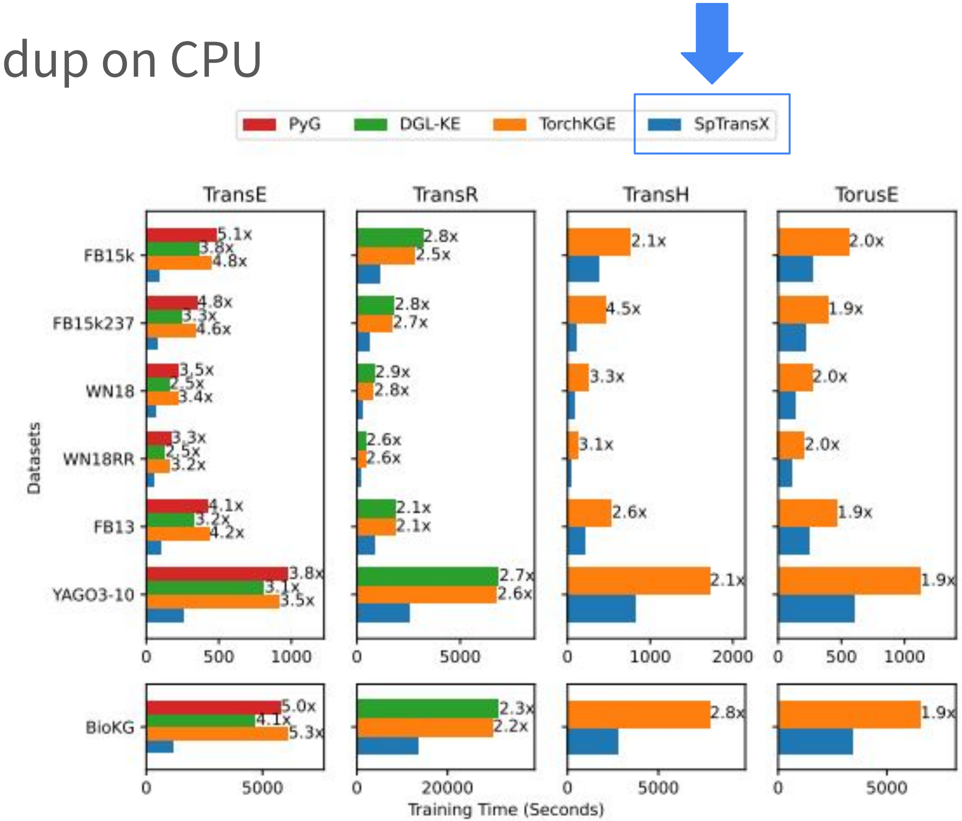
1. DGL-KE
2. PyTorch Geometric
3. TorchKGE
4. **SparseTransX**

Measured:

1. Total Training Time
2. GPU Peak Memory
3. Cache Miss Rate
4. FLOPs Count
5. Hits@10 Accuracy*

Results

Speedup on CPU



[200 epochs]

Datasets

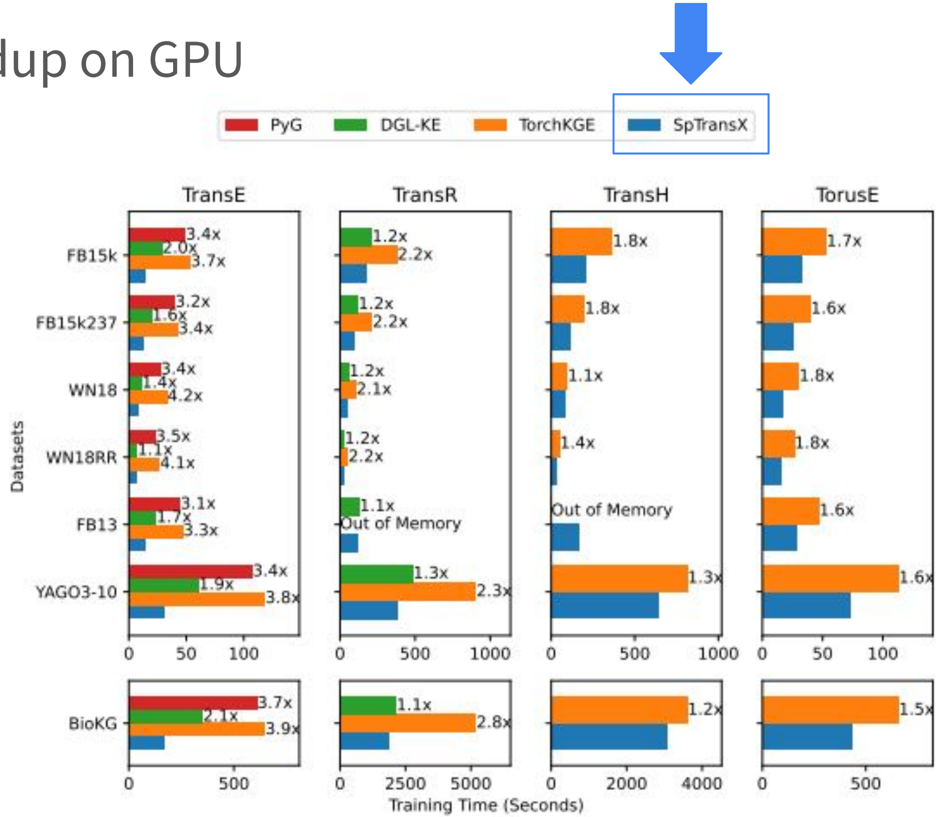
DATASET	ENTITY	RELATIONS	TRAINING TRIPLETS
FB15K	14951	1345	483142
FB15K237	14541	237	272115
WN18	40943	18	141442
WN18RR	40943	11	86835
FB13	67399	15342	316232
YAGO3-10	123182	37	1079040
BIOKG	93773	51	4762678

1.9x-5.3x Speedup

[NERSC Perlmutter system]
AMD EPYC 7763 (Milan) CPU with 64 cores and 512GB DDR4 memory

Results

Speedup on GPU



[200 epochs]

Datasets

DATASET	ENTITY	RELATIONS	TRAINING TRIPLETS
FB15K	14951	1345	483142
FB15K237	14541	237	272115
WN18	40943	18	141442
WN18RR	40943	11	86835
FB13	67399	15342	316232
YAGO3-10	123182	37	1079040
BIOKG	93773	51	4762678

1.1x-4.2x Speedup

[NERSC Perlmutter system]
1 x NVIDIA A100-SXM4 GPU with 40 GB VRAM

Results

Average GPU Memory Usage

MODEL	SPTRANSX	TORCHKGE	DGL-KE	PYG
TRANSE	**5.61**	13.55	11.37	13.54
TRANSR	**13.65**	20.42	30.73	-
TRANSH	**0.28**	3.1	-	-
TORUSE	**12.03**	15.87	-	-

1.3x to 11.1x improvement

Results

Accuracy

9 runs, 200-1000 epochs, WN18RR dataset

Average Hits@10

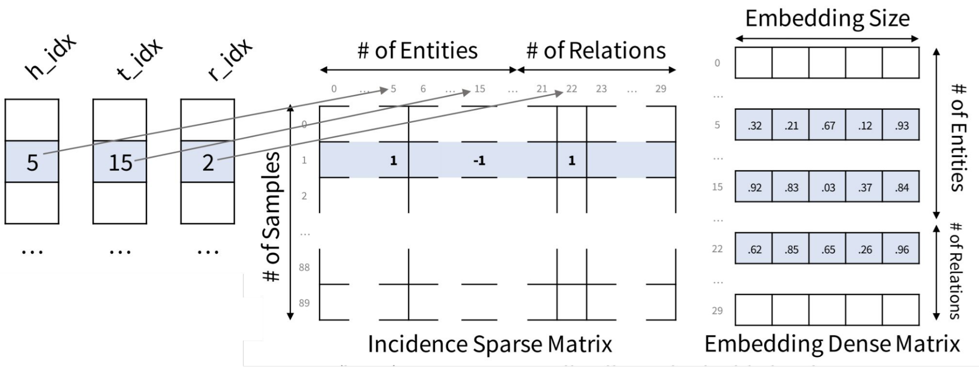
Model	TorchKGE	SpTransX
TransE	0.79 ± 0.001700	0.79 ± 0.002667
TransR	0.29 ± 0.005735	0.33 ± 0.006154
TransH	0.76 ± 0.012285	0.79 ± 0.001832
TorusE	0.73 ± 0.003258	0.73 ± 0.002780

Summary

Formulated KG embedding training using SpMM

1. SpMM is well researched and has more efficient implementations
2. Generalized the formulation to many KG emb models
3. Compact model training → reduces GPU memory usage, speedup in CPU and GPU
4. The sparse training does not affect accuracy

Training loop modification: Preprocessed Dataloader, SpMM as part of model training

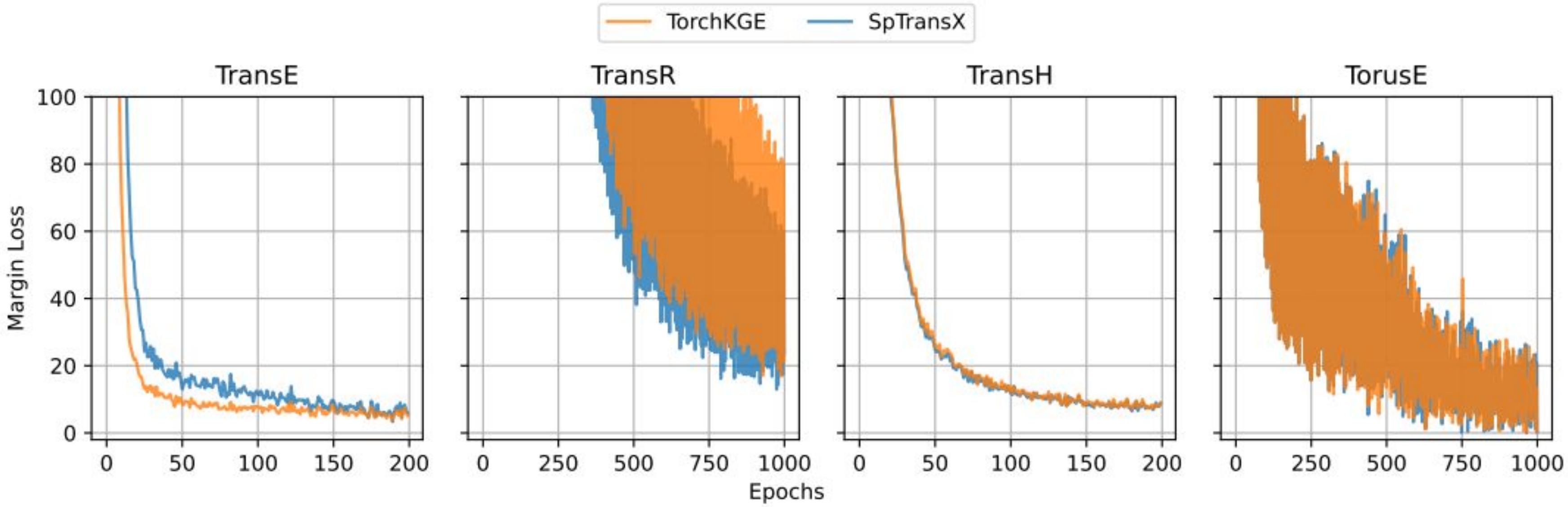


https://github.com/HipGraph/**SpTransX**

Thank You!

Results

Loss Curve



Results

Average FLOPs Count (factor of $\times 10^{10}$)

Model	SpTransX	TorchKGE	DGL-KE	PyG
TransE	**220**	483.87	293.06	483.82
TransR	**567.37**	1157.94	874.67	-
TransH	**9.66**	19.58	-	-
TorusE	**289.99**	387.93	-	-

Results

Average Cache Miss Rate (in %)

Model	SpTransX	TorchKGE	DGL-KE	PyG
TransE	**26.54**	29.37	29.99	29.04
TransR	**17.02**	19.20	29.54	-
TransH	10.43	**9.75**	-	-
TorusE	**21.53**	22.94	-	-

Distributed Training

Distributed Data-Parallel (DDP) training time for the COVID19 dataset
(60,820 entities, 62 relations, and 1,032,939 triplets)

Table: Scaling TransE model on COVID-19 dataset

Number of GPUs	500 epoch time (seconds)
4	706.38
8	586.03
16	340.00
32	246.02
64	179.95

Distributed Training

Training time of TransE model with Freebase dataset (250M triplets, 77M entities. 74K relations, batch size 393K) on 32 NVIDIA A100 GPUs. **FSDP enables model training with larger embedding when DDP fails.**

Table: Average Time of 15 Epochs (in seconds)

Embedding Size	DDP (Distributed Data Parallel)	FSDP (Fully Sharded Data Parallel)
16	65.07 ± 1.641	63.35 ± 1.258
20	Out of Memory	96.44 ± 1.490

Backpropagation of SpMM

Proof that $\frac{\partial C}{\partial X} = A^T$

To see why $\frac{\partial C}{\partial X} = A^T$ is used in the gradient calculation, we can consider the following small matrix multiplication without loss of generality.

$$A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$$

$$C = \begin{bmatrix} c_1 & c_2 \\ c_3 & c_4 \end{bmatrix}$$

Where $C = AX$, thus-

$$c_1 = f(x_1, x_3)$$
$$c_2 = f(x_2, x_4)$$
$$c_3 = f(x_1, x_3)$$
$$c_4 = f(x_2, x_4)$$

Backpropagation of SpMM

Therefore-

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial x_1} &= \frac{\partial \mathfrak{L}}{\partial c_1} \times \frac{\partial c_1}{\partial x_1} + \frac{\partial \mathfrak{L}}{\partial c_2} \times \frac{\partial c_2}{\partial x_1} + \frac{\partial \mathfrak{L}}{\partial c_3} \times \frac{\partial c_3}{\partial x_1} + \frac{\partial \mathfrak{L}}{\partial c_4} \times \frac{\partial c_4}{\partial x_1} \\
&= \frac{\partial \mathfrak{L}}{\partial c_1} \times \frac{\partial \mathfrak{c}_1}{\partial x_1} + 0 + \frac{\partial \mathfrak{L}}{\partial c_3} \times \frac{\partial \mathfrak{c}_3}{\partial x_1} + 0 \\
&= a_1 \times \frac{\partial \mathfrak{L}}{\partial c_1} + a_3 \times \frac{\partial \mathfrak{L}}{\partial c_3}
\end{aligned}$$

Similarly-

$$\begin{aligned}
\frac{\partial \mathfrak{L}}{\partial x_2} &= a_1 \times \frac{\partial \mathfrak{L}}{\partial c_2} + a_3 \times \frac{\partial \mathfrak{L}}{\partial c_4} \\
\frac{\partial \mathfrak{L}}{\partial x_3} &= a_2 \times \frac{\partial \mathfrak{L}}{\partial c_1} + a_4 \times \frac{\partial \mathfrak{L}}{\partial c_3} \\
\frac{\partial \mathfrak{L}}{\partial x_4} &= a_2 \times \frac{\partial \mathfrak{L}}{\partial c_2} + a_4 \times \frac{\partial \mathfrak{L}}{\partial c_4}
\end{aligned}$$

Backpropagation of SpMM

This can be expressed as a matrix equation in the following manner-

$$\frac{\partial \mathfrak{L}}{\partial X} = \frac{\partial C}{\partial X} \times \frac{\partial \mathfrak{L}}{\partial C}$$

$$\implies \begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial x_1} & \frac{\partial \mathfrak{L}}{\partial x_2} \\ \frac{\partial \mathfrak{L}}{\partial x_3} & \frac{\partial \mathfrak{L}}{\partial x_4} \end{bmatrix} = \frac{\partial C}{\partial X} \times \begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial c_1} & \frac{\partial \mathfrak{L}}{\partial c_2} \\ \frac{\partial \mathfrak{L}}{\partial c_3} & \frac{\partial \mathfrak{L}}{\partial c_4} \end{bmatrix}$$

By comparing the individual partial derivatives computed earlier, we can say-

$$\begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial x_1} & \frac{\partial \mathfrak{L}}{\partial x_2} \\ \frac{\partial \mathfrak{L}}{\partial x_3} & \frac{\partial \mathfrak{L}}{\partial x_4} \end{bmatrix} = \begin{bmatrix} a_1 & a_3 \\ a_2 & a_4 \end{bmatrix} \times \begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial c_1} & \frac{\partial \mathfrak{L}}{\partial c_2} \\ \frac{\partial \mathfrak{L}}{\partial c_3} & \frac{\partial \mathfrak{L}}{\partial c_4} \end{bmatrix}$$

$$\implies \begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial x_1} & \frac{\partial \mathfrak{L}}{\partial x_2} \\ \frac{\partial \mathfrak{L}}{\partial x_3} & \frac{\partial \mathfrak{L}}{\partial x_4} \end{bmatrix} = A^T \times \begin{bmatrix} \frac{\partial \mathfrak{L}}{\partial c_1} & \frac{\partial \mathfrak{L}}{\partial c_2} \\ \frac{\partial \mathfrak{L}}{\partial c_3} & \frac{\partial \mathfrak{L}}{\partial c_4} \end{bmatrix}$$

$$\implies \frac{\partial \mathfrak{L}}{\partial X} = A^T \times \frac{\partial \mathfrak{L}}{\partial C}$$

$$\therefore \frac{\partial C}{\partial X} = A^T \quad \square$$