CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

A Deep Learning Blueprint for Relational Databases

TRL @ NeurIPS 2023

Jan Neumann

Faculty of Electrical Engineering, CTU in Prague

November 2023

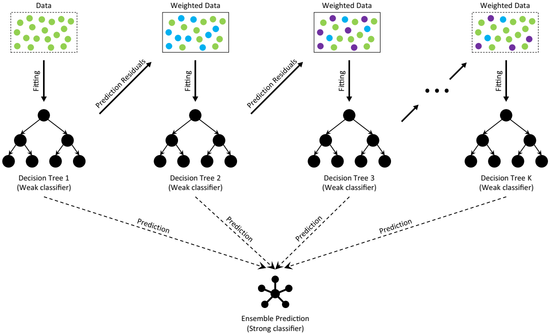
Background

Tabular Data

Example:

Acct District	Acct Since	Date	Amount	Status
Prague	1993-02-26	1994-01-05	80952	A
Tabor	1995-04-07	1996-04-29	30276	B
Prague	1993-02-26	1997-12-08	30276	A
Strakonice	1997-08-18	1998-10-14	318480	D
Strakonice	1997-08-08	1998-04-19	110736	C
...	...	...	...	...

For such prediction tasks, standard statistical models still dominate, due to their superior performance [1].



Gradient-Boosted Decision Trees [2]. Figure taken from [3].

Can You Use Deep Learning?

- Most methods based on the Transformer architecture [4]
- Examples:
  - TabTransformer (2020) [5]
  - TabPFN (2023) [6]

However...

What if our data is relational – more than 1 table, with foreign keys?

Tabular vs. Relational Data

Account Id	Date	Amount	Status
2	1994-01-05	80952	A
19	1996-04-29	30276	B
2	1997-12-08	30276	A
37	1998-10-14	318480	D
38	1998-04-19	110736	C
...	...	...	...

Account Id	District Id	Frequency	Date Created	...
2	1	Monthly	1993-02-26	...
19	21	Monthly	1995-04-07	...
37	20	Monthly	1997-08-18	...
38	20	Weekly	1997-08-08	...
...	...	...	...	...

District Id	District	Location	...
1	Prague	Prague	...
20	Strakonice	S Bohemia	...
21	Tabor	S Bohemia	...
...	...	...	...

Tabular vs. Relational Data

Acct District	Acct Since	Date	Amount	Status
Prague	1993-02-26	1994-01-05	80952	A
Tabor	1995-04-07	1996-04-29	30276	B
Prague	1993-02-26	1997-12-08	30276	A
Strakonice	1997-08-18	1998-10-14	318480	D
Strakonice	1997-08-08	1998-04-19	110736	C
...	...	...	...	...

Tabular vs. Relational Data

Acct District	Acct Since	Date	Amount	Status
Prague	1993-02-26	1994-01-05	80952	A
Tabor	1995-04-07	1996-04-29	30276	B
Prague	1993-02-26	1997-12-08	30276	A
Strakonice	1997-08-18	1998-10-14	318480	D
Strakonice	1997-08-08	1998-04-19	110736	C
...	...	...	...	...

Tabular vs. Relational Data

Account Id	Date	Amount	Status
2	1994-01-05	80952	A
19	1996-04-29	30276	B
2	1997-12-08	30276	A
37	1998-10-14	318480	D
38	1998-04-19	110736	C
...	...	...	...

Account Id	District	Frequency	Date Created	...
2	Prague	Monthly	1993-02-26	...
19	Tabor	Monthly	1995-04-07	...
37	Strakonice	Monthly	1997-08-18	...
38	Strakonice	Weekly	1997-08-08	...
...	...	...	...	...

Tabular vs. Relational Data

Account Id	Date	Amount	Status
2	1994-01-05	80952	A
19	1996-04-29	30276	B
2	1997-12-08	30276	A
37	1998-10-14	318480	D
38	1998-04-19	110736	C
...	...	...	...

Account Id	District Id	Frequency	Date Created	...
2	1	Monthly	1993-02-26	...
19	21	Monthly	1995-04-07	...
37	20	Monthly	1997-08-18	...
38	20	Weekly	1997-08-08	...
...	...	...	...	...

District Id	District	Location	...
1	Prague	Prague	...
20	Strakonice	S Bohemia	...
21	Tabor	S Bohemia	...
...	...	...	...

Tabular vs. Relational Data

Account Id	Date	Amount	Status
2	1994-01-05	80952	A
19	1996-04-29	30276	B
2	1997-12-08	30276	A
37	1998-10-14	318480	D
38	1998-04-19	110736	C
...	...	...	...

Account Id	District Id	Frequency	Date Created	...
2	1	Monthly	1993-02-26	...
19	21	Monthly	1995-04-07	...
37	20	Monthly	1997-08-18	...
38	20	Weekly	1997-08-08	...
...	...	...	...	...

District Id	District	Location	...
1	Prague	Prague	...
20	Strakonice	S Bohemia	...
21	Tabor	S Bohemia	...
...	...	...	...

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure
- Propositionalization [7]

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure
- Propositionalization [7]
  - This approach dominates the industry [8, 9]

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure
- Propositionalization [7]
  - This approach dominates the industry [8, 9]
  - Less expensive

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure
- Propositionalization [7]
  - This approach dominates the industry [8, 9]
  - Less expensive
  - Helps the predictor understand the original relational structure

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

- Naively: Universal relation – join all tables
  - Expensive!
  - Difficult to observe the complex original data structure
- Propositionalization [7]
  - This approach dominates the industry [8, 9]
  - Less expensive
  - Helps the predictor understand the original relational structure
  - Loss of information :(

How Do We Train On Relational Data?

First idea: **Convert to tabular, then use tabular learners**

Either expensive, or principially suboptimal!

End-to-end Deep Learning?

- Can We Fully Preserve The Data Structure?

End-to-end Deep Learning?

- Can We Fully Preserve The Data Structure?
- Can we utilize the ability of deep learning to find its own optimal latent representation of the data?

End-to-end Deep Learning?

- Can We Fully Preserve The Data Structure?
- Can we utilize the ability of deep learning to find its own optimal latent representation of the data?
- **Graph Neural Networks** [10]
- **Transformer architecture** [4]

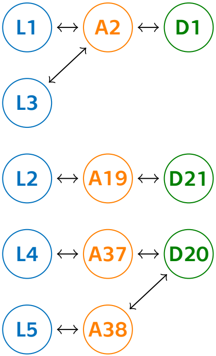
End-to-end Deep Learning?

- Can We Fully Preserve The Data Structure?
- Can we utilize the ability of deep learning to find its own optimal latent representation of the data?
- **Graph Neural Networks** [10]
- **Transformer architecture** [4]
- Incorporate both intra-relational (attribute) and inter-relational (foreign key) structure within the message-passing scheme

Our Proposal

Message Passing on Orig. Example

Two-level Multi-relational Hypergraph



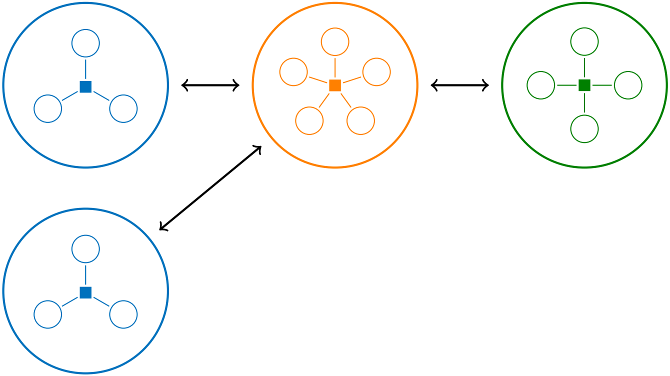
Account Id	Date	Amount	Status
2	1994-01-05	80952	A
19	1996-04-29	30276	B
2	1997-12-08	30276	A
37	1998-10-14	318480	D
38	1998-04-19	110736	C
...	...	...	...

Account Id	District Id	Frequency	Date Created	...
2	1	Monthly	1993-02-26	...
19	21	Monthly	1995-04-07	...
37	20	Monthly	1997-08-18	...
38	20	Weekly	1997-08-08	...
...	...	...	...	...

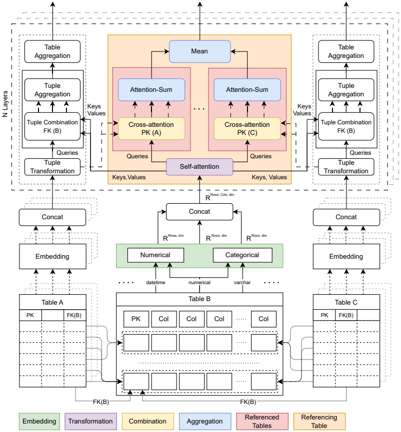
District Id	District	Location	...
1	Prague	Prague	...
20	Strakonice	S Bohemia	...
21	Tabor	S Bohemia	...
...	...	...	...

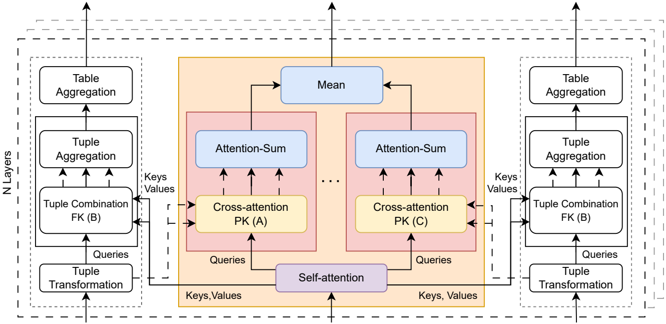
Message Passing on Orig. Example

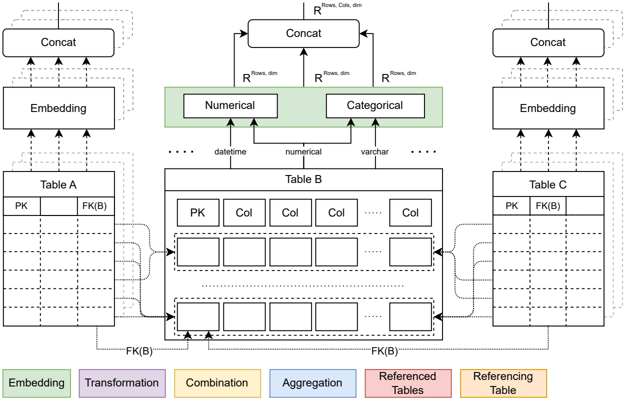
Two-level Multi-relational Hypergraph



A Deep Learning Blueprint for Relational Databases
TRL @ NeurIPS 2023




N Layers
Table Aggregation
Tuple Aggregation
Tuple Combination FK (B)
Keys Values
Queries
Tuple Transformation
Mean
Attention-Sum
Cross-attention PK (A)
Cross-attention PK (C)
Queries
Self-attention
Keys,Values
Keys, Values


Concat
Embedding
Table A
PK
FK(B)
$R^{Rows, Cols, dim}$
Concat
$R^{Rows, dim}$
$R^{Rows, dim}$
$R^{Rows, dim}$
Numerical
Categorical
datetime
numerical
varchar
Table B
PK
Col
Col
Col
Col
Concat
Embedding
Table C
PK
FK(B)
FK(B)
FK(B)
Embedding
Transformation
Combination
Aggregation
Referenced Tables
Referencing Table

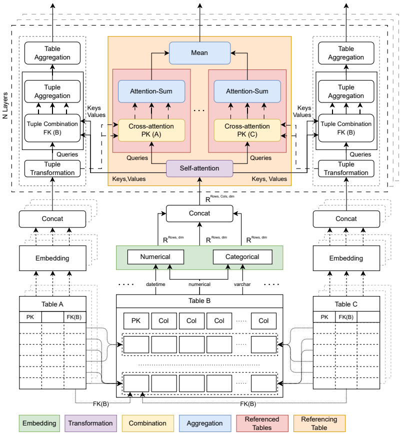


Table Aggregation
Tuple Aggregation
Tuple Combination FK (B)
Tuple Transformation
Queries
Keys Values
N Layers
Mean
Attention-Sum
Cross-attention PK (A)
Cross-attention PK (C)
Self-attention
Keys,Values
Keys, Values
Concat
Embedding
Numerical
Categorical
datetime
numerical
varchar
Table A
Table B
Table C
PK
FK(B)
Col
Embedding
Transformation
Combination
Aggregation
Referenced Tables
Referencing Table

Additional Offerings[1]

1. Load SQL databases directly

Additional Offerings[1]

1. Load SQL databases directly
2. Optionally auto-detect attribute semantics (numeric vs. categorical)

Additional Offerings[1]

1. Load SQL databases directly
2. Optionally auto-detect attribute semantics (numeric vs. categorical)
3. Per-type handling and embedding

Additional Offerings[1]

1. Load SQL databases directly
2. Optionally auto-detect attribute semantics (numeric vs. categorical)
3. Per-type handling and embedding
4. Directly usable with existing GNN and Transformer implementations

Additional Offerings[1]

1. Load SQL databases directly
2. Optionally auto-detect attribute semantics (numeric vs. categorical)
3. Per-type handling and embedding
4. Directly usable with existing GNN and Transformer implementations

Work-in-progress Python library that extends PyTorch Geometric [11].

Additional Offerings[1]

1. Load SQL databases directly
2. Optionally auto-detect attribute semantics (numeric vs. categorical)
3. Per-type handling and embedding
4. Directly usable with existing GNN and Transformer implementations

Work-in-progress Python library that extends PyTorch Geometric [11].

[1] Tested on a large library of example relational datasets [12]. Unavailable anymore at the time of writing. We are considering re-publishing the datasets ourselves.

Results

Results

category:	Tab.	Rel.[2]	Prop.	NeSy[3]	Ours		
datasets	**MLP**	**RDN-b** [15]	**getML** [9]	**CILP** [16]	**l_1**	**l_2**	**l_3**
PTE	N/A	44.94%	**100.00%**	100.00%	**100.00%**	83.05%	**100.00%**
university	81.82%	81.82%	54.55%	81.82%	**100.00%**	**100.00%**	**100.00%**
NCAA	**100.00%**	47.50%	**100.00%**	78.75%	67.92%	71.69%	67.92%
cs	N/A	63.33%	96.67%	96.67%	**100.00%**	**100.00%**	**100.00%**
UTube	N/A	84.15%	98.93%	**99.39%**	98.16%	98.16%	98.16%
mutagen	87.50%	85.71%	82.86%	92.86%	**94.59%**	**94.59%**	**94.59%**
Dunur	N/A	23.17%	**97.56%**	**97.56%**	94.54%	94.54%	94.54%
MuskSmall	N/A	77.78%	74.07%	66.67%	**83.33%**	77.77%	50.00%
WebKP	N/A	82.51%	**83.04%**	65.40%	68.57%	51.99%	65.14%
DCG	N/A	72.57%	65.17%	61.06%	73.89%	65.92%	**79.20%**
Pima	N/A	32.17%	**77.11%**	75.65%	58.82%	73.20%	74.50%
CiteSeer	N/A	**66.16%**	47.41%	37.36%	50.15%	51.51%	37.76%
Carcinogen.	N/A	53.06%	62.07%	**65.31%**	64.61%	63.07%	60.00%
Toxicology	N/A	63.73%	57.02%	**72.55%**	61.76%	67.64%	61.76%
Chess	40.91%	34.09%	33.64%	48.86%	**50.84%**	**50.84%**	**50.84%**
Atheroscler.	26.72%	18.10%	22.41%	28.45%	**33.76%**	32.46%	31.16%

[2] Statistical relational learning [13]

[3] Neuro-symbolic integration [14]

References I

[1] Ravid Shwartz-Ziv and Amitai Armon. *Tabular Data: Deep Learning Is Not All You Need*. Nov. 23, 2021. DOI: 10.48550/arXiv.2106.03253. URL: http://arxiv.org/abs/2106.03253 (visited on 11/25/2023). preprint.

[2] Jerome H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine.". In: *The Annals of Statistics* 29.5 (Oct. 2001), pp. 1189–1232. DOI: 10.1214/aos/1013203451.

[3] Haowen Deng et al. "Ensemble Learning for the Early Prediction of Neonatal Jaundice with Genetic Features". In: *BMC Medical Informatics and Decision Making* 21 (Dec. 1, 2021). DOI: 10.1186/s12911-021-01701-9.

[4] Ashish Vaswani et al. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.

[5] Xin Huang et al. *TabTransformer: Tabular Data Modeling Using Contextual Embeddings*. Dec. 11, 2020. DOI: 10.48550/arXiv.2012.06678. URL: http://arxiv.org/abs/2012.06678 (visited on 11/25/2023). preprint.

[6] Noah Hollmann et al. *TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second*. Sept. 16, 2023. DOI: 10.48550/arXiv.2207.01848. URL: http://arxiv.org/abs/2207.01848 (visited on 11/25/2023). preprint.

[7] Stefan Kramer, Nada Lavrač, and Peter Flach. "Propositionalization Approaches to Relational Data Mining". In: *Relational Data Mining*. Ed. by Sašo Džeroski and Nada Lavrač. Berlin, Heidelberg: Springer, 2001, pp. 262–291. DOI: 10.1007/978-3-662-04599-2_11.

References II

[8] The Alteryx. *Featuretools*. URL: https://www.featuretools.com/ (visited on 11/26/2023).

[9] The SQLNet Company GmbH. *getML*. URL: https://www.getml.com/ (visited on 11/26/2023).

[10] Zonghan Wu et al. "A Comprehensive Survey on Graph Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1 (Jan. 2021), pp. 4–24. DOI: 10.1109/TNNLS.2020.2978386.

[11] PyTorch Geometric Team. *PyTorch Geometric*. URL: https://pytorch-geometric.readthedocs.io/en/latest/ (visited on 11/26/2023).

[12] Jan Motl and Oliver Schulte. *The CTU Prague Relational Learning Repository*. Nov. 10, 2015. DOI: 10.48550/arXiv.1511.03086. URL: http://arxiv.org/abs/1511.03086 (visited on 11/26/2023). preprint.

[13] Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning*. MIT Press, 2007. 602 pp.

[14] Barbara Hammer and Pascal Hitzler, eds. *Perspectives of Neural-Symbolic Integration*. Red. by Janusz Kacprzyk. Vol. 77. Studies in Computational Intelligence. Berlin, Heidelberg: Springer, 2007. DOI: 10.1007/978-3-540-73954-8.

[15] Sriraam Natarajan et al. "Gradient-Based Boosting for Statistical Relational Learning: The Relational Dependency Network Case". In: *Mach Learn* 86.1 (Jan. 1, 2012), pp. 25–56. DOI: 10.1007/s10994-011-5244-9.

References III

[16] Manoel V. M. França, Gerson Zaverucha, and Artur S. d'Avila Garcez. "Fast Relational Learning Using Bottom Clause Propositionalization with Artificial Neural Networks". In: *Mach Learn* 94.1 (Jan. 1, 2014), pp. 81–104. DOI: 10.1007/s10994-013-5392-1.