Knowledge Hypergraphs
- Prediction Beyond Binary Relations

Bahare Fatemi , Perouz Taslakian , David Vazquez, David Poole
University of British Columbia, 2019

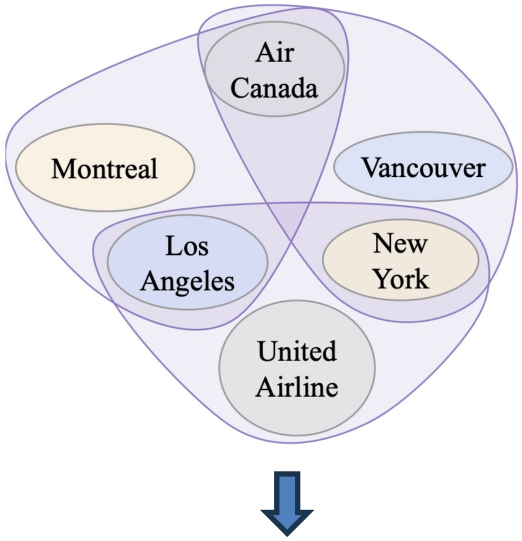
2025-02-19
HoonUi Lee

Contents

Knowledge Hypergraph

❖ Generalization of knowledge graphs

- ❑ Store facts in the form of relations among any number of entities
  - ‣ while at most two entities in Knowledge Graph
- ❑ More than **1/3** of the **entities** participate in **non-binary** relations, **61%** of the **relations** are **non-binary** in FREEBASE
  - ‣ why Knowledge Hypergraph is needed
- ❑ Two methods converting non-binary relations into binary ones
  - ‣ reification
  - ‣ star-to-clique



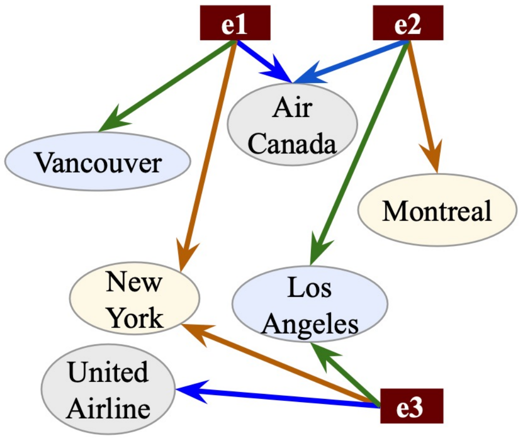
files_between(Air canada, Montreal, Los Angeles)
files_between(Air canada, Vancouver, New York)
files_between(United Airline, New York, Los Angeles)

Knowledge Hypergraph

- Reification

❖ Higher-arity to binary

- ❑ Generating a **new entity** and creating a new tuple by inserting an existing entity into the tail part
  - ‣ **no loss of information** during the transformation process



files_between(e1, Vancouver)
files_between(e1, New York)
files_between(e1, Air Canada)
...

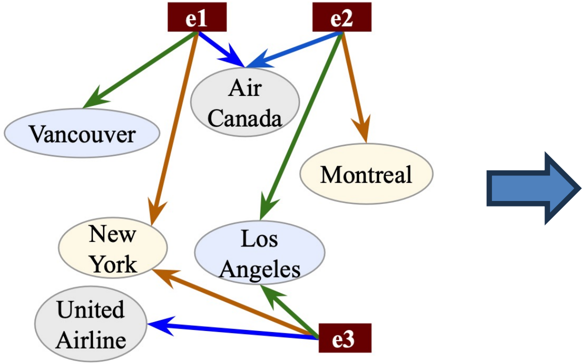
Knowledge Hypergraph

- Reification

❖ Problem occurs in test

❑ Entities that the model never encounters during training

- do not have a learned embedding for these new entities



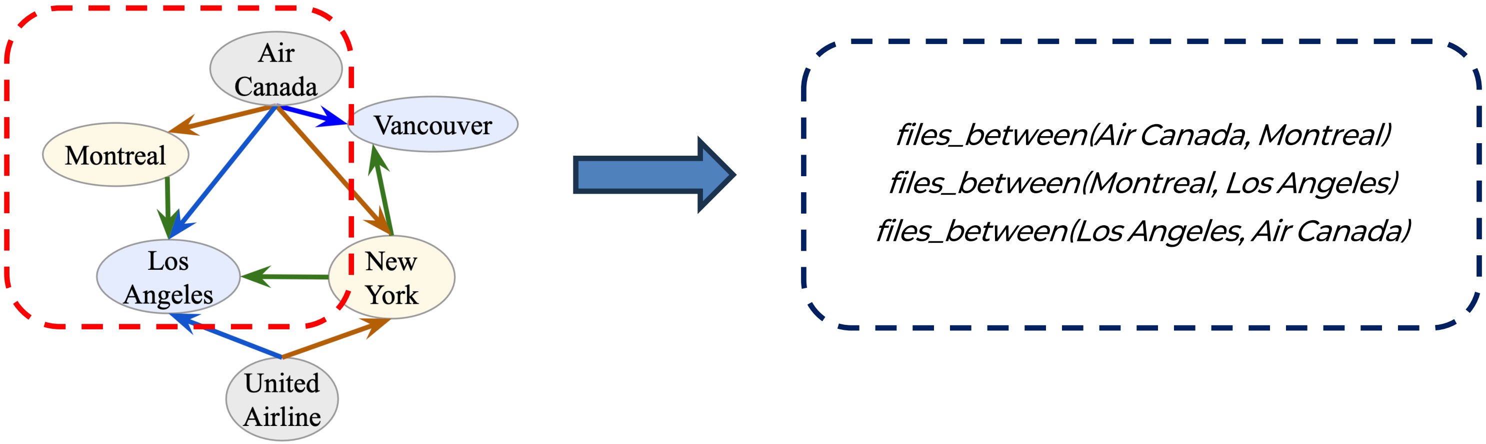
Knowledge Hypergraph

- Star-to-Clique

❖ Higher-arity to binary

❑ Converts a tuple defined on k entities into kC_2 tuples

with distinct relations between all pairwise entities in tuple



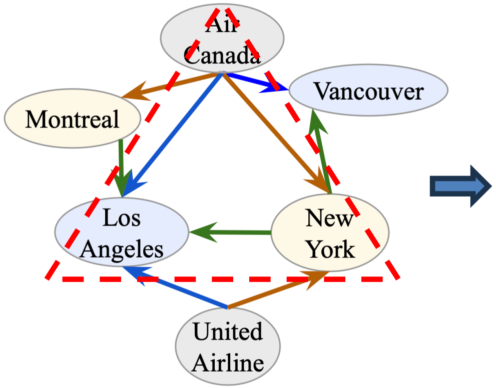
Knowledge Hypergraph

- Star-to-Clique

❖ Information loss occurs

- ❑ Information loss occurs during the conversion process
  - information that **did not exist in the original graph** may emerge

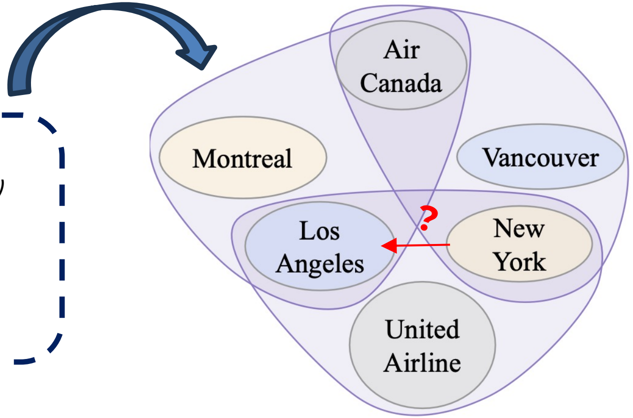
    in the process of considering all pairs



files_between(Air Canada, Los Angeles)

files_between(Air Canada, New York)

files_between(Los Angeles, New York)



Knowledge Hypergraph

- KG completion

❖ Inefficient in Knowledge Hypergraph Completion

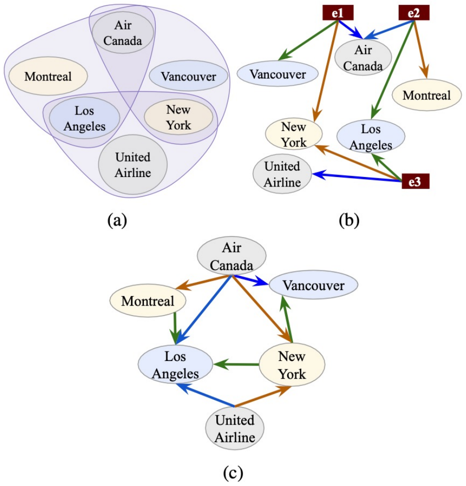
❑ Applying the existing link prediction method
to the transformed binary triples did not yield satisfactory performance

- need another methods

❑ Predictions may need to vary based on the position
that entity appears in within the tuple

- does not matter if the relation is symmetric

➔ HSimplE & HypE



(a) (b) (c)

HSimplE

- SimplE

❖ Inspired by SimplE

- ❑ SimplE learns two embedding vectors $e^{(1)}$ and $e^{(2)}$ for entity e, two embedding vectors $r^{(1)}$ and $r^{(2)}$ for relation r

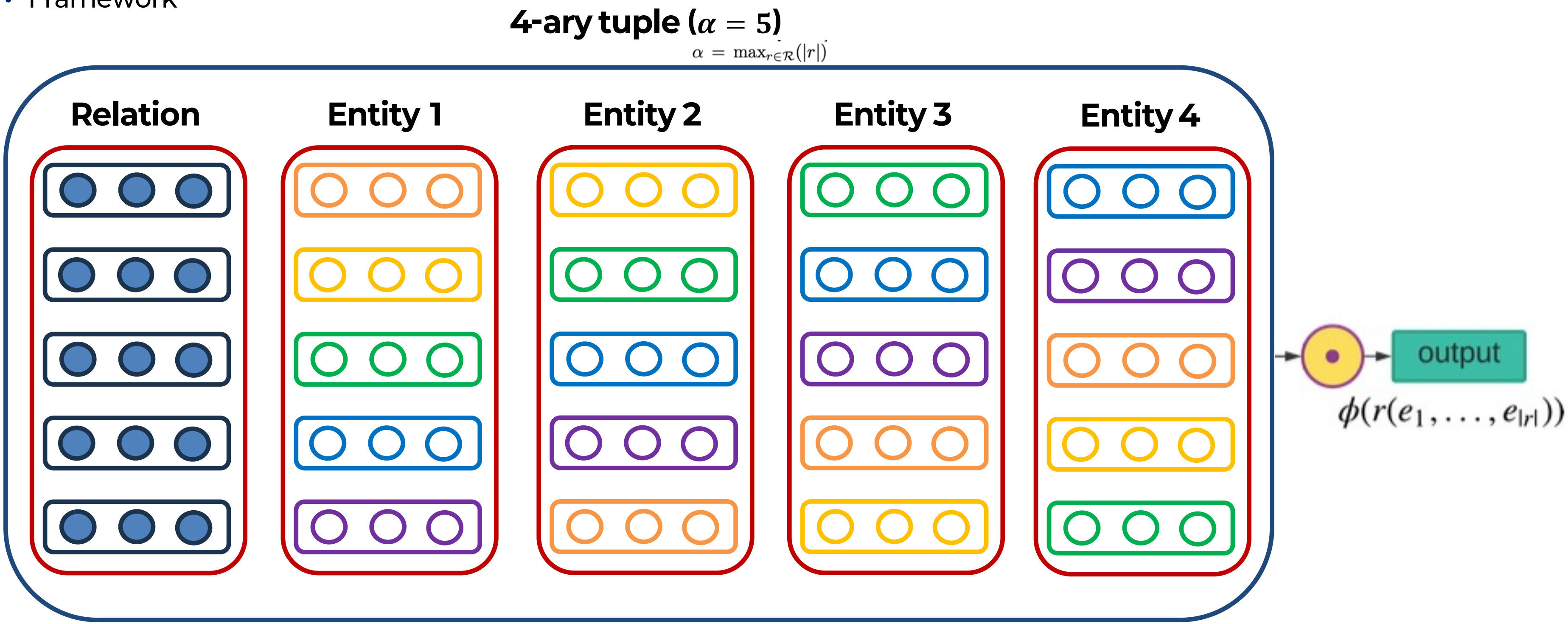
- ❑ Scoring function

$$\phi(r(e_1, e_2)) = \odot(\mathbf{r}^{(1)}, \mathbf{e}_1^{(1)}, \mathbf{e}_2^{(2)}) + \odot(\mathbf{r}^{(2)}, \mathbf{e}_2^{(1)}, \mathbf{e}_1^{(2)})$$

- ❑ Each entity learns distinct embedding vectors for the Head and Tail positions
  - Consider the directionality of relation

HSimplE

- Framework



HSimplE

- Framework

❖ Use position with shift

- **Single** vector for relation, entity (Unlike SimplE)
  - entity vector can be seen as **concatenation** of different representations of **entity's every possible position** ($e = e_1 + e_2 + e_3 + \ ...$)

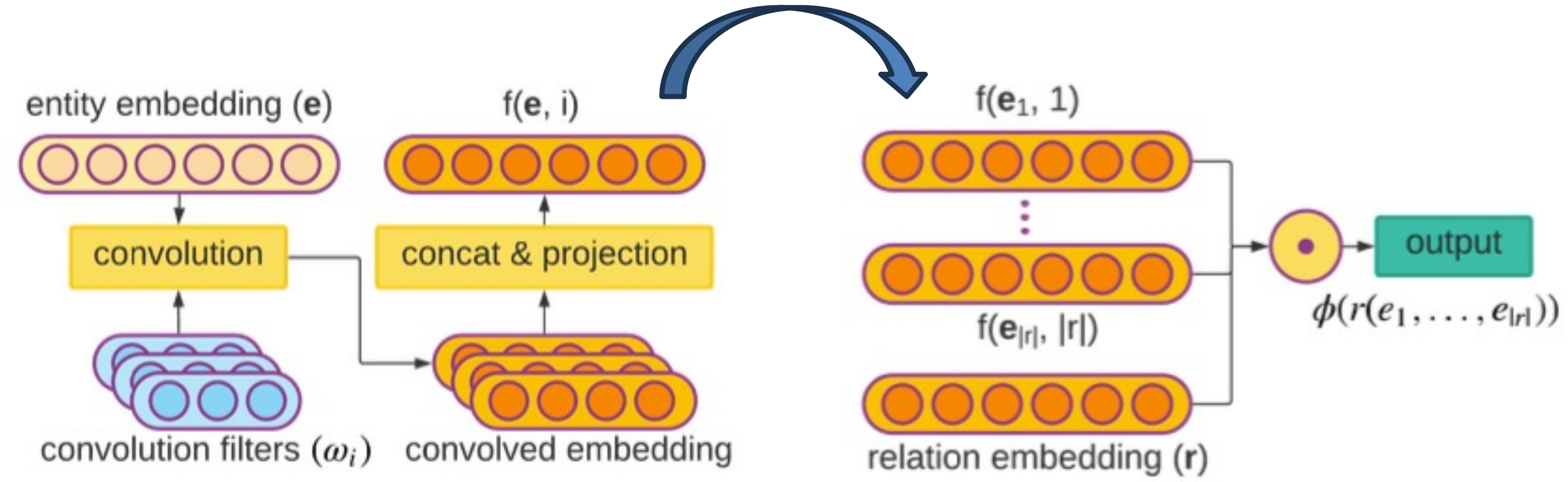
- Scoring function

$$\phi(r(e_i, e_j, \ldots, e_k)) = \odot(\mathbf{r}, \mathbf{e_i}, \mathrm{shift}(\mathbf{e_j}, \mathrm{len}(\mathbf{e_j})/\alpha), \ldots, \mathrm{shift}(\mathbf{e_k}, \mathrm{len}(\mathbf{e_k}) \cdot (\alpha - 1)/\alpha))) \quad (1)$$

  - len(e) returns length of vector e
  - shift(v, x) shifts vector v to the left by x steps
  - $\alpha = \max_{\{r \in R\}}(|r|)$

HypE

- Framework

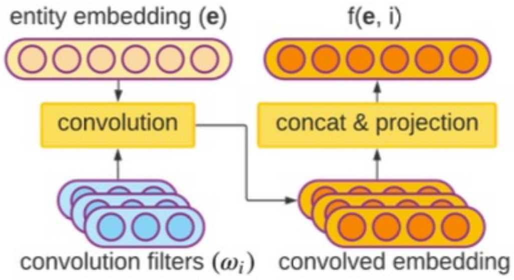


HypE

- Framework

❖ Use position with convolution filter

- ❑ **Single** vector for relation, entity
- ❑ Positional convolutional weight **filters** for **each possible position**
  - transform the embedding of each entity
  - with concatenation and projection



- ❑ $f(e,i) = concat(e * \omega_{\{i1\}}, \dots, e * \omega_{\{in\}})P$
  - n : the # of filters per position
  - P: projection matrix
  - The result of the function is an entity representation of size d

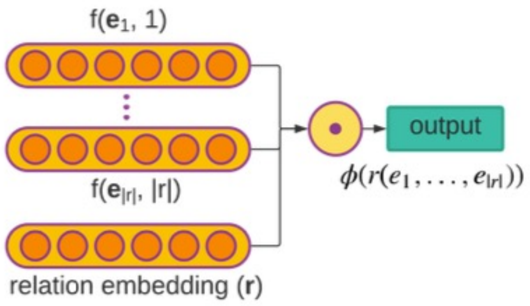
HypE

- Framework

❖ Combine with relation embedding

❑ Scoring function

$$\phi(r(e_1, \ldots, e_{|r|})) = \odot(\mathbf{r}, f(\mathbf{e_1}, 1), \ldots, f(\mathbf{e}_{|\mathbf{r}|}, |r|))$$



HypE

- vs. HSimplE

❖ Advantage of learning positional filters

❑ Learning disentangled from its position keeps entity representations simple

- HSimplE incorporates position when learning entity embeddings

❑ Robustness in test

- ability to infer new entity-position relationships
- In HSimplE, hard to generalizing and predicting entity-position pairs that did not appear during the training process

Proposed Method

- vs. Previous work

❖ Overcome risks about information

- ❑ Using **Star-to-Clique**, tuples might generate different meanings from the original KHG
- ❑ While HSimplE and HypE use original tuple, without converting

❖ Overcome limitations in test

- ❑ Using **Reification**, entity appear in test while unseen in training data inevitably
- ❑ While HSimplE and HypE, experiments can be conducted under the **transductive setting**

HSimplE & HypE

- Training

❖ Loss Function

❑ $$\mathcal{L}(\mathbf{r}, \mathbf{e}) = \sum_{x' \in \tau'_{train}} -log\left(\frac{e^{\phi(x')}}{e^{\phi(x')} + \sum_{x \in T_{neg}(x')} e^{\phi(x)}}\right)$$

- Cross-entropy loss based on Softmaxd

❑ Produce a set of negative samples of size N|r|

- |r| : the number of arguments that the relation takes
- N: ratio of negative samples (hyperparameter)

Experiment

- Dataset

			number of tuples			number of tuples with respective arity				
Dataset	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	#train	#valid	#test	#arity=2	#arity=3	#arity=4	#arity=5	#arity=6
WN18	40,943	18	141,442	5,000	5,000	151,442	0	0	0	0
FB15k	14,951	1,345	483,142	50,000	59,071	592,213	0	0	0	0
JF17K	29,177	327	77,733	–	24,915	56,322	34,550	9,509	2,230	37
FB-AUTO	3,410	8	6,778	2,255	2,180	3,786	0	215	7,212	0
M-FB15K	10,314	71	415,375	39,348	38,797	82,247	400,027	26	11,220	0

Table 5: Dataset Statistics.

- ❑ **WN18, FB15K** : with binary relation

- ❑ **JF17K, FB-AUTO, M-FB15K** : with non-binary relation

Experiment

- Knowledge Hypergraph Completion

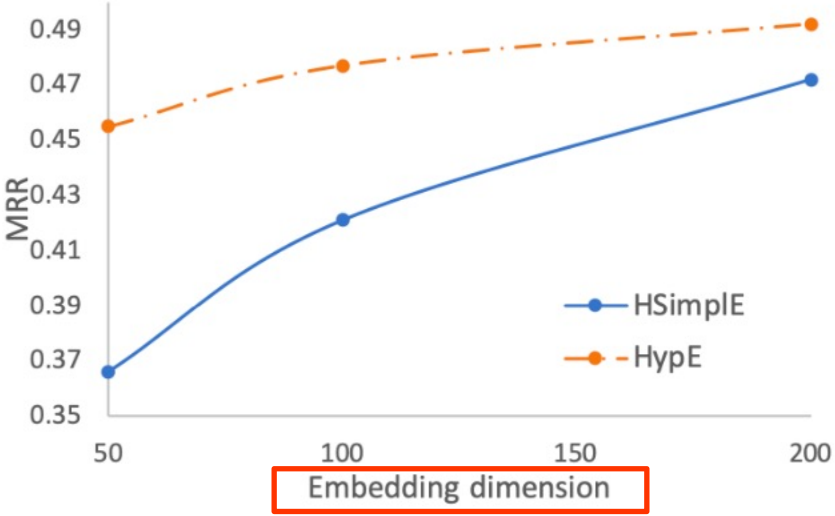
	Model	JF17K MRR	JF17K Hit@1	JF17K Hit@3	JF17K Hit@10	FB-AUTO MRR	FB-AUTO Hit@1	FB-AUTO Hit@3	FB-AUTO Hit@10	M-FB15K MRR	M-FB15K Hit@1	M-FB15K Hit@3	M-FB15K Hit@10
reification	r-SimplE	0.102	0.069	0.112	0.168	0.106	0.082	0.115	0.147	0.051	0.042	0.054	0.070
multi-arity	m-DistMult	0.463	0.372	0.510	0.634	0.784	0.745	0.815	0.845	0.705	0.633	0.740	0.844
	m-CP	0.391	0.298	0.443	0.563	0.752	0.704	0.785	0.837	0.680	0.605	0.715	0.828
	m-TransH [Wen *et al.*, 2016]	0.444	0.370	0.475	0.581	0.728	0.727	0.728	0.728	0.623	0.531	0.669	0.809
	HSimplE (Ours)	0.472	0.378	0.520	0.645	0.798	0.766	0.821	0.855	0.730	0.664	0.763	0.859
	HypE (Ours)	**0.494**	**0.408**	**0.538**	**0.656**	**0.804**	**0.774**	**0.823**	**0.856**	**0.777**	**0.725**	**0.800**	**0.881**

- r- : to binary triple with reification
- m- : extend existing method to multi-arity

❑ HypE shows better performance than HSimplE

Experiment

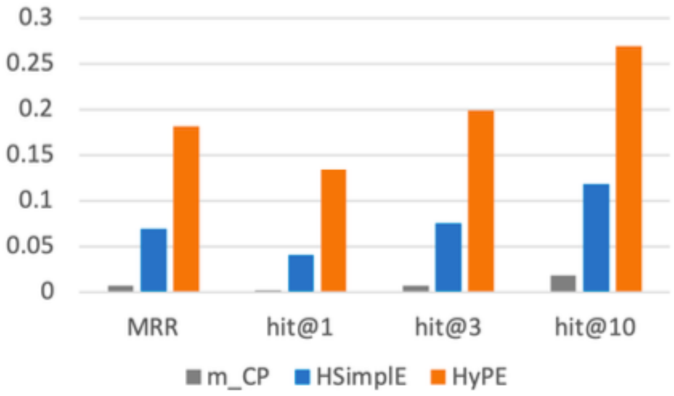
- In constrained budget



- ❑ Analyzed that, at low embedding dimensions,
  HSimplE lacked the resources to learn position information effectively
- ❑ HypE learns position information independently of the embedding representation

Experiment

- Missing position test



❏ Train and test with "Missing Position Test Set"

- entity-position that not encounter in train set, but in test set (1,806 test samples)

❏ HSimplE can infer, but HypE do better

Experiment

- Knowledge Graph Completion

	WN18				FB15k			
Model	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
CP [Hitchcock, 1927]	0.074	0.049	0.080	0.125	0.326	0.219	0.376	0.532
TransH [Wang *et al.*, 2014]	-	-	-	0.867	-	-	-	0.585
m-TransH [Wen *et al.*, 2016]	0.671	0.495	0.839	0.923	0.351	0.228	0.427	0.559
DistMult [Yang *et al.*, 2015]	0.822	0.728	0.914	0.936	0.654	0.546	0.733	0.824
HSimplE (Ours) and SimplE [Kazemi and Poole, 2018]	**0.942**	**0.939**	**0.944**	**0.947**	**0.727**	**0.660**	0.773	0.838
HypE (Ours)	0.934	0.927	0.940	0.944	0.725	0.648	**0.777**	**0.856**

❑ Both HSimplE and HypE outperformed existing embedding-based models

Experiment

Dataset	#arity=2	#arity=3	#arity=4	#arity=5	#arity=6
JF17K	56,322	34,550	9,509	2,230	37

- Ablation Study on Different Arities

Model	Arity 2	3	4-5-6	All
r-SimplE	0.478	0.025	0.017	0.168
m-DistMult	0.495	0.648	0.809	0.634
m-CP	0.409	0.563	0.765	0.560
m-TransH [Wen *et al.*, 2016]	0.411	0.617	0.826	0.596
HSimplE (Ours)	**0.497**	**0.699**	0.745	0.645
HypE (Ours)	0.466	0.693	**0.858**	**0.656**
# train tuples	36,293	18,846	6,772	61,911
# test tuples	10,758	10,736	3,421	24,915

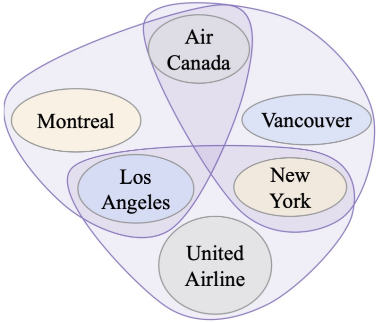
Table 4: Breakdown performance of Hit@10 across relations with different arities on JF17K dataset along with their statistics.

- ❑ Difference between r-SimplE and HSimplE in multi-arity
- ❑ Proposed methods show state-of-the-art performance

Conclusion

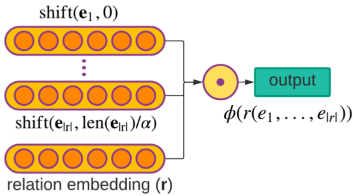
❖ Previous work

- ❑ Two methods that convert Knowledge Hypergraph to Knowledge graph
- ❑ Reification and Star-to-Clique have fundamental issue with the method itself
- ❑ Tuples are used for training, requiring the incorporation of position information



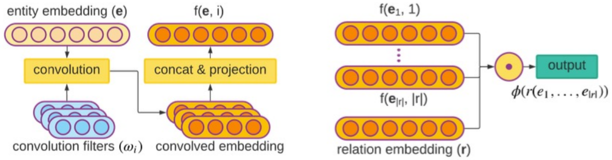
❖ HSimplE

- ❑ Learn entity embedding with position-specific information using shifts



❖ HypE

- ❑ convolution filters for each position are used to learn independently from entity embedding



❖ Experiment

- ❑ Limitations of the Existing Converting Method
- ❑ HypE shows better performance than HSimplE in missing position test set