NEURAL INFORMATION PROCESSING SYSTEMS

Interstellar: Searching Recurrent Architecture for Knowledge Graph Embedding

Yongqi Zhang, Quanming Yao, Lei Chen

Presenter: Zhanke Zhou

zhankezhou@hust.edu.cn

2021.04.01

Paper: https://neurips.cc/virtual/2020/public/poster_722caafb4825ef5d8670710fa29087cf.html

Code: https://github.com/AutoML-4Paradigm/Interstellar

Outline

- Background
- Interstellar
  - Motivation
  - Method
  - Experiments
- Key takeaways and research directions

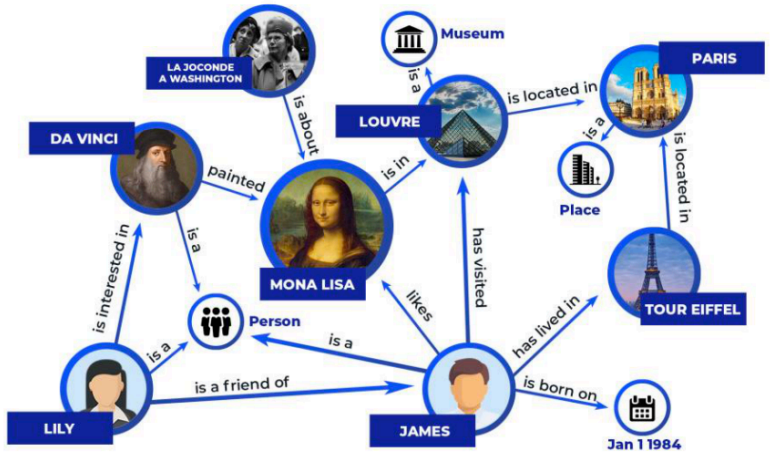
Background – Knowledge Graph (KG)

A knowledge graph

- Mainly describe real world entities and relations, organized in a graph
- Allows potentially interacting arbitrary entities with each other

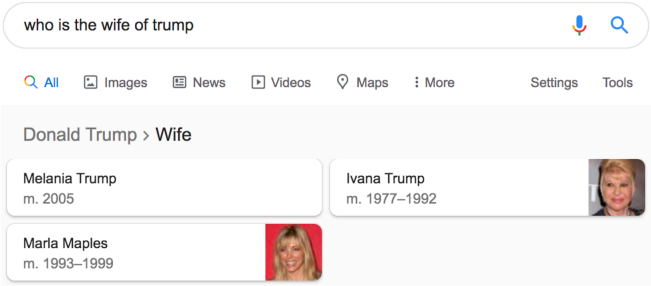
Preliminaries

- Graph representation: $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{F})$
- Entities $\mathcal{E}$
  - real world objects or concepts
- Relations $\mathcal{R}$
  - interactions between entities
- Facts $\mathcal{F}$
  - the basic unit in form of $(\mathrm{s}, \mathrm{r}, \mathrm{o})$
  - (subject entity, relation, object entity)

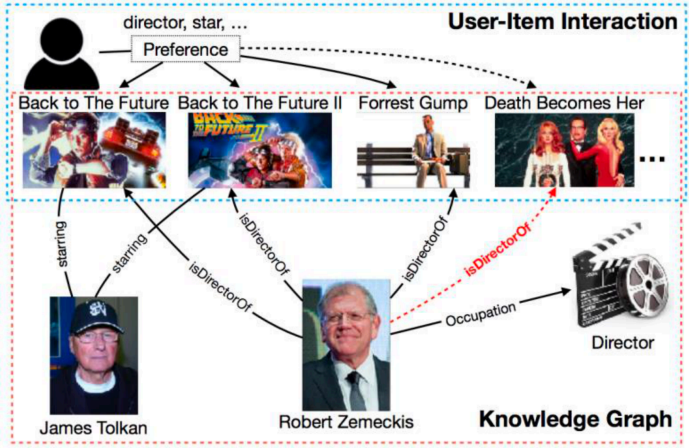


Applications

KGQA:



Recommendation:

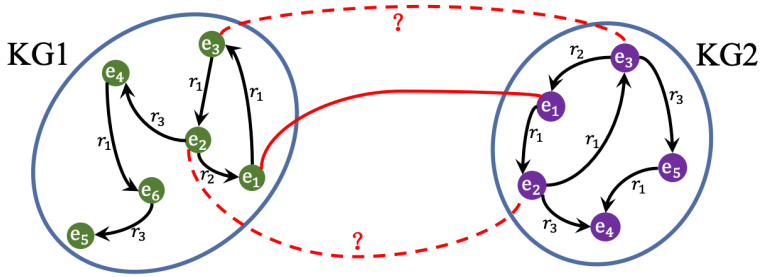


Background – Knowledge Graph Embedding (KGE)

- Knowledge Graph Embedding (KGE)
  - Encode entities and relations in KG into low-dimensional vectors space
  - while capturing nodes' and edges' connection properties

- Most KGE models based on embeddings define a scoring function ϕ to estimate the plausibility of any fact (s, r, o) using their embeddings:

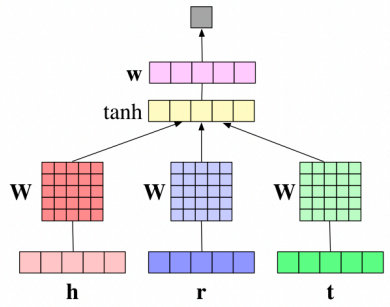
$$\phi(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$$



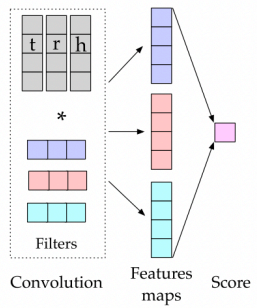
- Subtasks: Link Prediction & Entity Alignment
  - LP is the task of exploiting the existing facts to infer missing ones
    - addressing KG incompleteness
  - EA aims to align entities in different KGs referring the same instance
    - for matching and disambiguation

Background – Existing methods $\phi(s, r, o)$

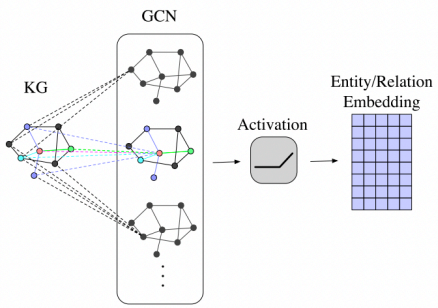
type	model		unit function
triplet-based	TransE [7]		$\mathbf{v}_t = \mathbf{s}_t + \mathbf{r}_t, \mathbf{h}_t = 0$
	ComplEx [49]		$\mathbf{v}_t = \mathbf{s}_t \otimes \mathbf{r}_t, , \mathbf{h}_t = 0$
GCN-based	R-GCN [40]		$\mathbf{s}_t = \sigma(\mathbf{s}_{t-1} + \sum_{s' \in \mathcal{N}(s)} \mathbf{W}_t^{(r)} \mathbf{s}'_{t-1})$
	GCN-Align [53]		$\mathbf{s}_t = \sigma(\mathbf{s}_{t-1} + \sum_{s' \in \mathcal{N}(s)} \mathbf{W}_t \mathbf{s}'_{t-1})$
path-based	PTransE [27]	add	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \mathbf{h}_{t-1} + \mathbf{r}_t$
		multiply	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \mathbf{h}_{t-1} \odot \mathbf{r}_t$
		RNN	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{r}_t, \mathbf{h}_{t-1})$
	Chains [11]		$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{s}_t, \mathbf{r}_t, \mathbf{h}_{t-1})$
	RSN [18]		$\mathbf{v}_t = \mathbf{W}_1 \mathbf{s}_t + \mathbf{W}_2 \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{r}_t, \text{cell}(\mathbf{s}_t, \mathbf{h}_{t-1}))$
	Interstellar		a searched recurrent network



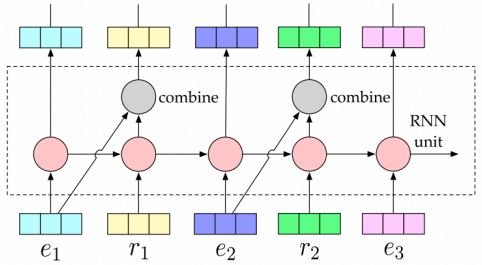
(a) MLP.



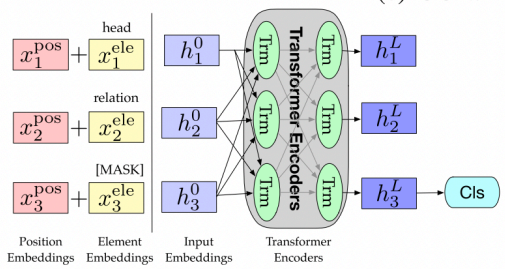
(b) CNN.



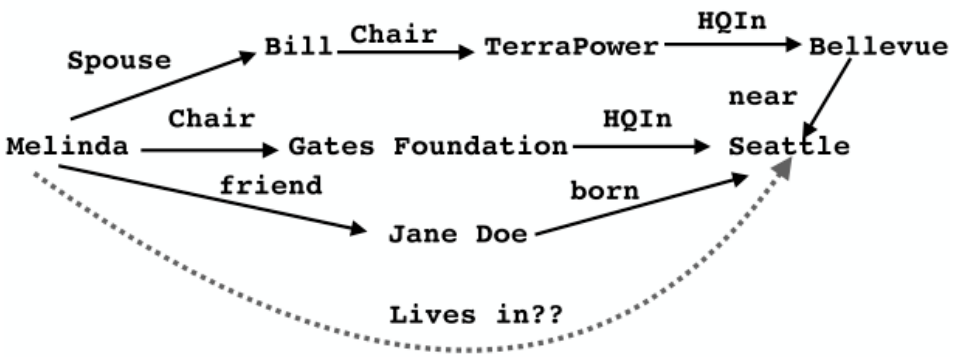
(c) GCN.



(d) RSN.



(e) Transformer.



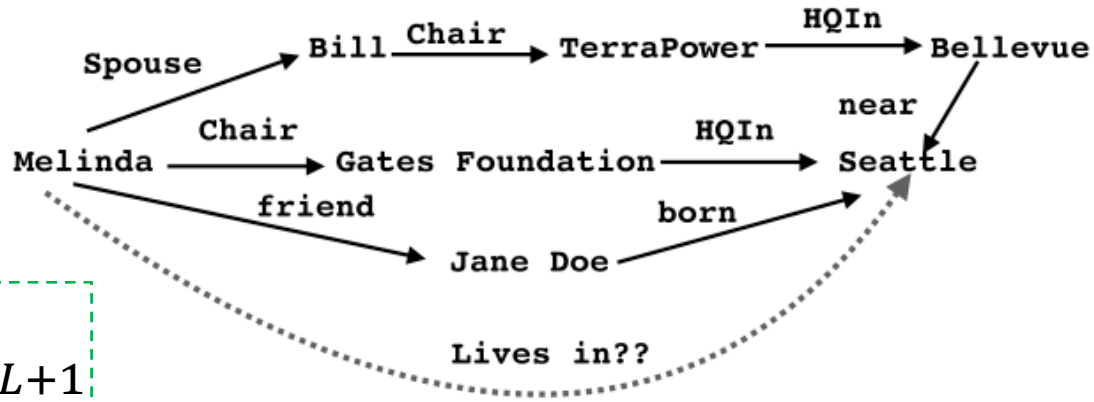
Background – Existing methods

Triples (s, r, o)

consecutively connected

Relational path

$$s_1 \xrightarrow{r_1} s_2 \xrightarrow{r_2} s_3 \cdots s_L \xrightarrow{r_L} s_{L+1}$$



short-term information inside triplets.

composition of relations.

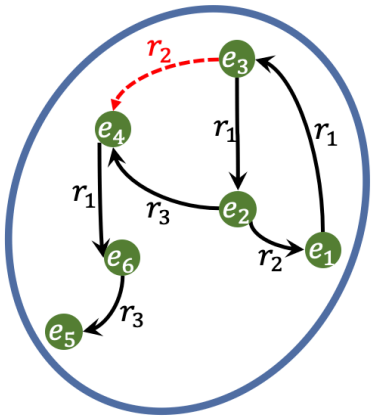
long-term information across triplets.

Relational path is an important and effective data structure that

- can capture the dependency among a sequence of triplets
- can preserve both short-term and long-term information in KG

Background – Knowledge Graph Embedding (KGE)

- Training
  - S^+: *positive samples* S^-: *negative samples*
  - Objectives: $max\ \phi^+(S^+) + \phi^-(S^-)$



$e_3 \xrightarrow{r_2} ?$

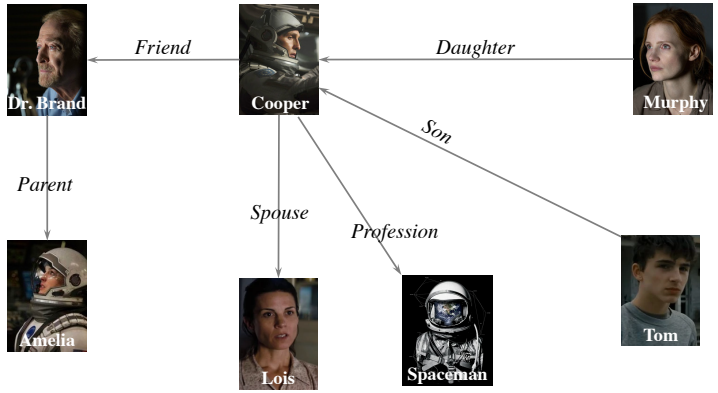
- Prediction
  - For link prediction, given an incomplete triple $(h, r, ?)$
  - the missing tail is inferred as the entity that results in the highest score:

$$t = \underset{e \in \mathcal{E}}{\operatorname{argmax}}\ \phi(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{e})$$

- Evaluation
  - Three common metrics
  - ***q*:** the rank of correct entity

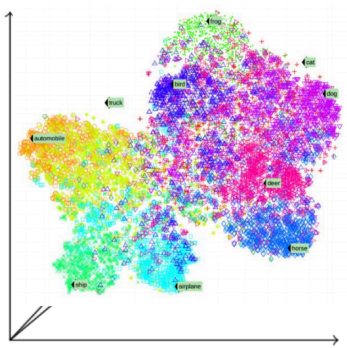
$$MR = \frac{1}{|Q|} \sum_{q \in Q} q \qquad MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{q} \qquad H@K = \frac{|\{q \in Q : q \leq K\}|}{|Q|}$$

Background – Knowledge Graph Embedding (KGE)



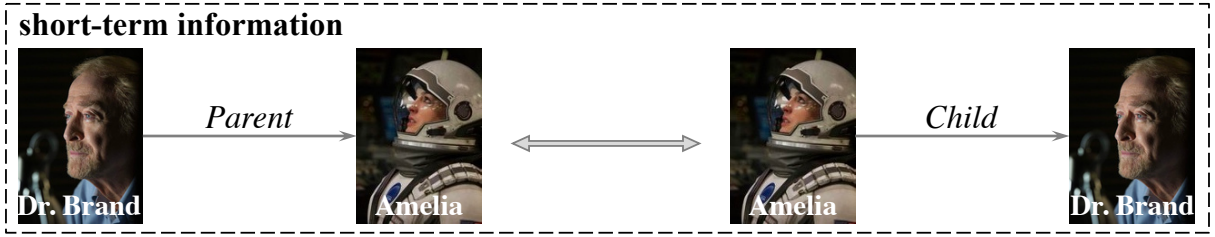
vectorize

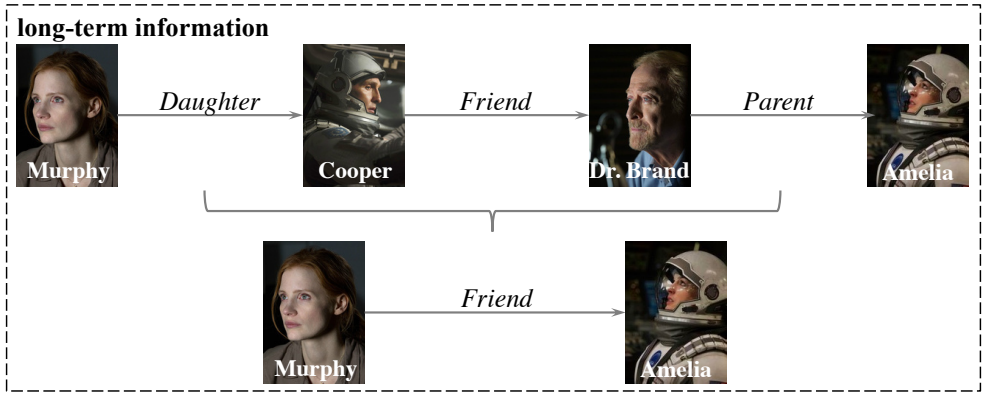
entities & relations



Main Challenge: How to preserve important information in the embeddings?

- Semantical / Topological information

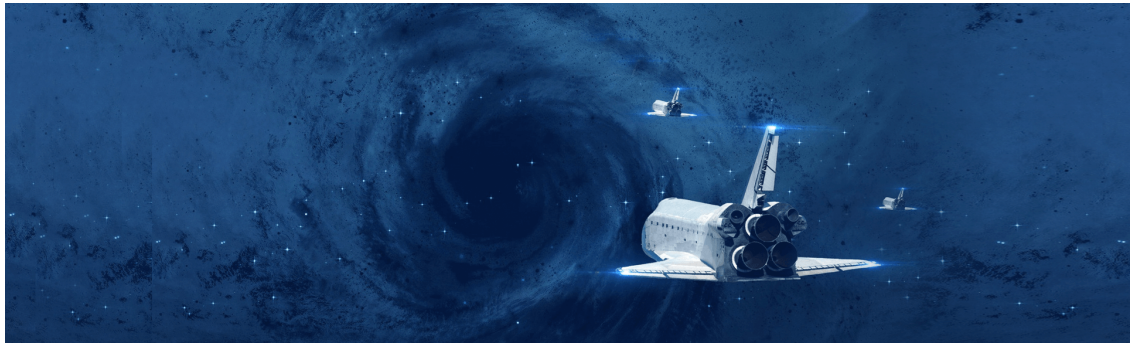




Outline

- Background
- Interstellar
  - Motivation
  - Method
  - Experiments
- Key takeaways and research directions

Interstellar – Motivation



Disadvantages of current KGE models

- Pure KGE model (based on single triplet)
  - cannot capture the information among multiple triplets
- RNN-based model (based on paths)
  - while it can capture the long-term information along steps
  - it will overlook domain-specific properties like the semantics inside each triplet without a customized architecture
- GCN-based model (based on (sub)graphs)
  - do not scale well since the entire KG needs to be processed and it has large sample complexity
  - GCN has been theoretically proved to have worse generalization guarantee than RNN in sequence learning tasks

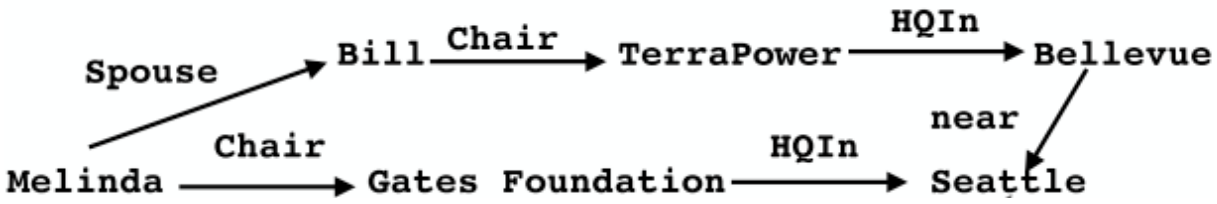
AutoML

- Inspired by the success of neural architecture search (NAS)
- However, designing the search space for recurrent neural network attracts little attention
  - the searched architectures mainly focus on cells rather than connections among cells

Interstellar

- propose to search recurrent architectures as the *Interstellar* to learn from the relational path

Interstellar – Definition

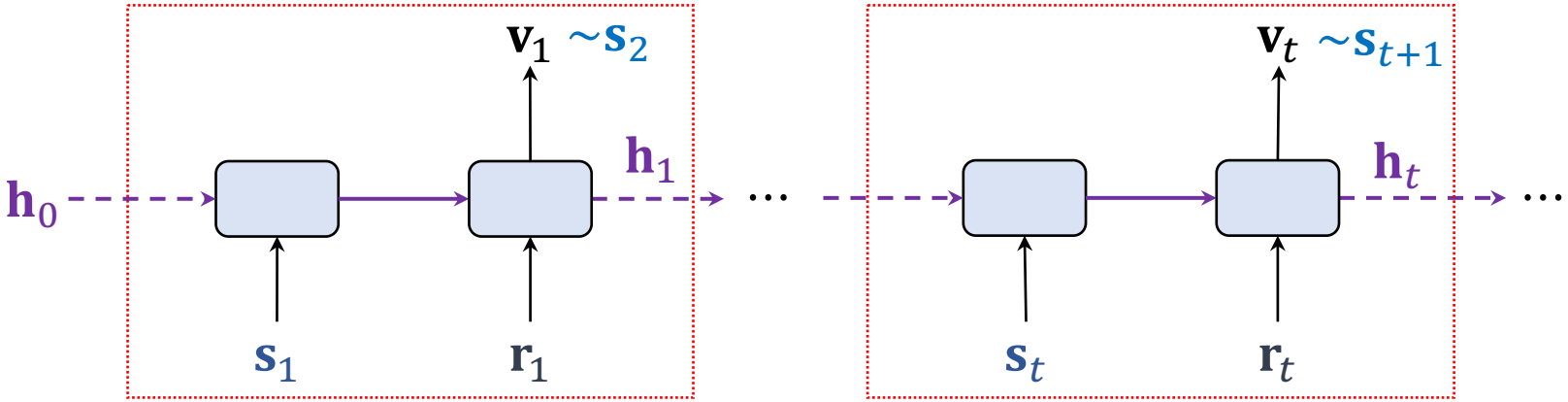


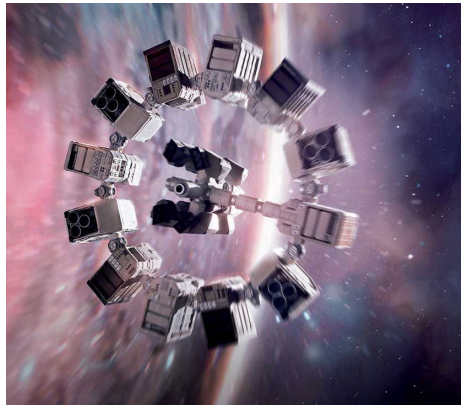
Relational path

$$s_1 \xrightarrow{r_1} s_2 \xrightarrow{r_2} \cdots s_t \xrightarrow{r_t} \cdots s_L \xrightarrow{r_L} s_{L+1}$$

The Interstellar recurrently processes the triples in path by

$$[\mathbf{v}_t, \mathbf{h}_t] = f(\mathbf{s}_t, \mathbf{r}_t, \mathbf{h}_{t-1}), \qquad \forall t = 1 \ldots L$$



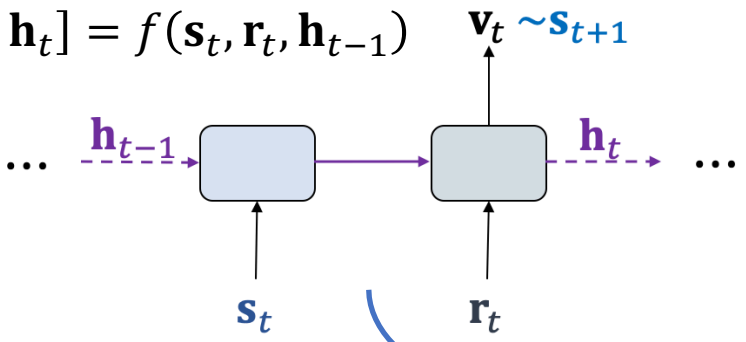


How to adaptively model $f(\cdot)$ in different KG scenarios?

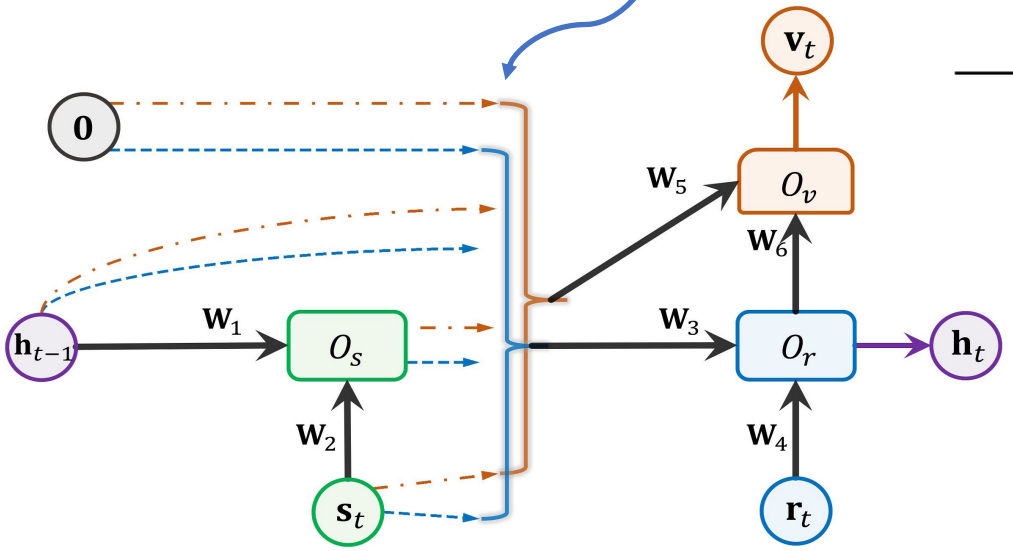
Interstellar – Search Space

Compared with standard RNNs, which recurrently model each input vectors, the Interstellar models the relational path with triplets as basic unit.
In this way, we can determine how to model short-term information inside each triplet and what long-term information should be passed along the triplets.

$$[\mathbf{v}_t, \mathbf{h}_t] = f(\mathbf{s}_t, \mathbf{r}_t, \mathbf{h}_{t-1})$$



search the connections
i.e., the dashed lines

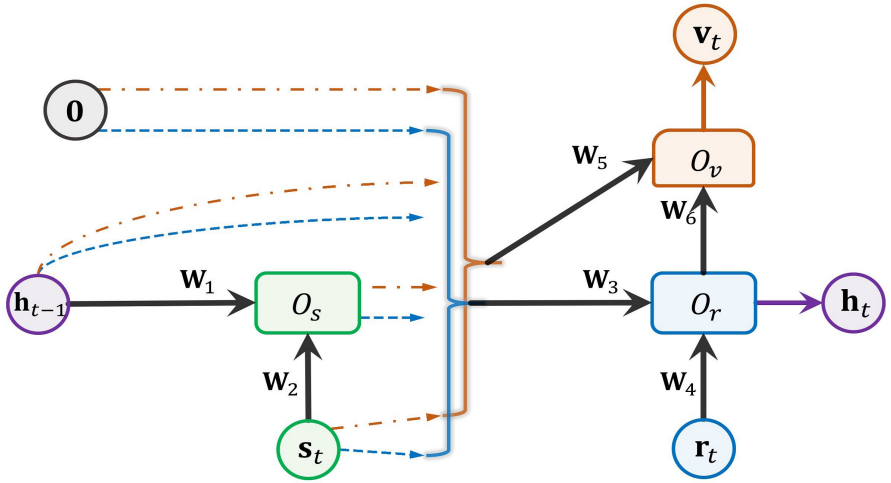


type	model		unit function	complexity
triplet-based	TransE [7]		$\mathbf{v}_t = \mathbf{s}_t + \mathbf{r}_t, \mathbf{h}_t = 0$	$O(md)$
	ComplEx [49]		$\mathbf{v}_t = \mathbf{s}_t \otimes \mathbf{r}_t, , \mathbf{h}_t = 0$	$O(md)$
GCN-based	R-GCN [40]		$\mathbf{s}_t = \sigma(\mathbf{s}_{t-1} + \sum_{s' \in \mathcal{N}(s)} \mathbf{W}_t^{(r)} \mathbf{s}'_{t-1})$	$O(\lvert\mathcal{E}\rvert\lvert\mathcal{R}\rvert d)$
	GCN-Align [53]		$\mathbf{s}_t = \sigma(\mathbf{s}_{t-1} + \sum_{s' \in \mathcal{N}(s)} \mathbf{W}_t \mathbf{s}'_{t-1})$	$O(\lvert\mathcal{E}\rvert d)$
path-based	PTransE [27]	add	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \mathbf{h}_{t-1} + \mathbf{r}_t$	$O(mLd)$
		multiply	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \mathbf{h}_{t-1} \odot \mathbf{r}_t$	$O(mLd)$
		RNN	$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{r}_t, \mathbf{h}_{t-1})$	$O(mLd^2)$
	Chains [11]		$\mathbf{v}_t = \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{s}_t, \mathbf{r}_t, \mathbf{h}_{t-1})$	$O(mLd^2)$
	RSN [18]		$\mathbf{v}_t = \mathbf{W}_1 \mathbf{s}_t + \mathbf{W}_2 \mathbf{h}_t, \mathbf{h}_t = \text{cell}(\mathbf{r}_t, \text{cell}(\mathbf{s}_t, \mathbf{h}_{t-1}))$	$O(mLd^2)$
	Interstellar		a searched recurrent network	$O(mLd^2)$

Detailed search space

macro-level	connections	$\mathbf{h}_{t-1}, O_s, \mathbf{0}, \mathbf{s}_t$
$\hat{\boldsymbol{\alpha}} \in \hat{\mathcal{A}}$	combinators	$+, \odot, \otimes$, gated
micro-level	activation	identity, tanh, sigmoid
$\check{\boldsymbol{\alpha}} \in \check{\mathcal{A}}$	weight matrix	$\{\mathbf{W}_i\}_{i=1}^{6}, \mathbf{I}$

Interstellar – Search Space



Detailed search space

macro-level	connections	$\mathbf{h}_{t-1}, O_s, \mathbf{0}, \mathbf{s}_t$	$4^2 \times 4^3 = 1024$
$\hat{\boldsymbol{\alpha}} \in \hat{\mathcal{A}}$	combinators	$+, \odot, \otimes$, gated	
micro-level	activation	identity, tanh, sigmoid	$3^2 \times 2^6 = 576$
$\check{\boldsymbol{\alpha}} \in \check{\mathcal{A}}$	weight matrix	$\{\mathbf{W}_i\}_{i=1}^{6}, \mathbf{I}$	

6×10^5 candidates in total

How to efficiently search the architectures? 🤔

Bi-level Optimization Problem:

$$\boldsymbol{\alpha}^* = \arg\max_{\boldsymbol{\alpha} \in \mathcal{A}} \mathcal{M}\left(f(\boldsymbol{F}^*; \boldsymbol{\alpha}), \mathcal{G}_{\text{val}}\right)$$

$$\text{s.t.} \quad \boldsymbol{F}^* = \arg\min_{\boldsymbol{F}} \mathcal{L}\left(f(\boldsymbol{F}; \boldsymbol{\alpha}), \mathcal{G}_{\text{tra}}\right)$$

α: model architecture
F: learnable weights
M: measurement on G_{val}
L: loss on G_{tra}

Interstellar – Search Algorithm

How to efficiently search the architectures?

Search appropriate $\alpha \in \mathcal{A}$ that maximize the validation performance

$$\boldsymbol{\alpha}^* = \arg\max_{\boldsymbol{\alpha}\in\mathcal{A}} \mathcal{M}\left(f(\boldsymbol{F}^*;\boldsymbol{\alpha}), \mathcal{G}_{\text{val}}\right), \quad \text{s.t.} \quad \boldsymbol{F}^* = \arg\min_{\boldsymbol{F}} \mathcal{L}\left(f(\boldsymbol{F};\boldsymbol{\alpha}), \mathcal{G}_{\text{tra}}\right)$$

Two common search approaches:

F: learnable weights

M: measurement on G_{val}

Stand-alone approach:

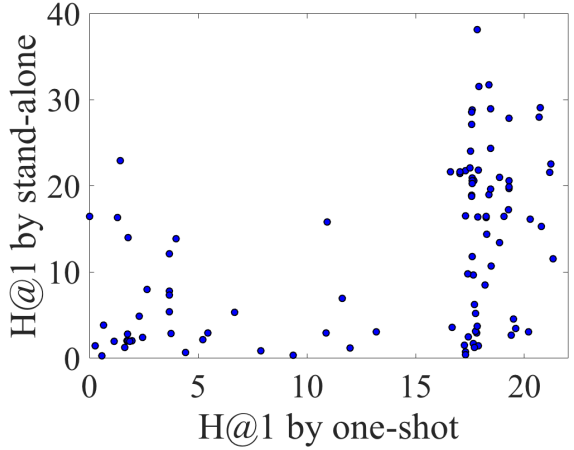
- $\mathcal{M}$ is accurate
- $\boldsymbol{F}^*$ needs high cost

[Zoph and Le 2017]

One-shot approach:

- $\mathcal{M}$ is not always reliable
- $\boldsymbol{F}^*$ is shared and efficient

[Pham et al. 2018, Liu et al. 2019]



Correlation is weak.

Is it possible to take advantage of both?

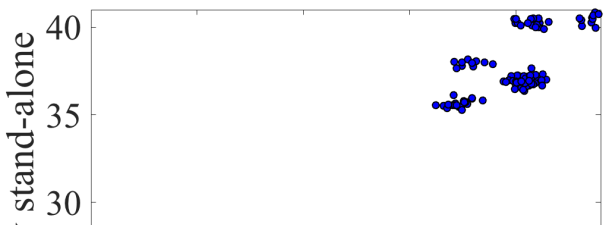
Interstellar – Search Algorithm

Hybrid search algorithm

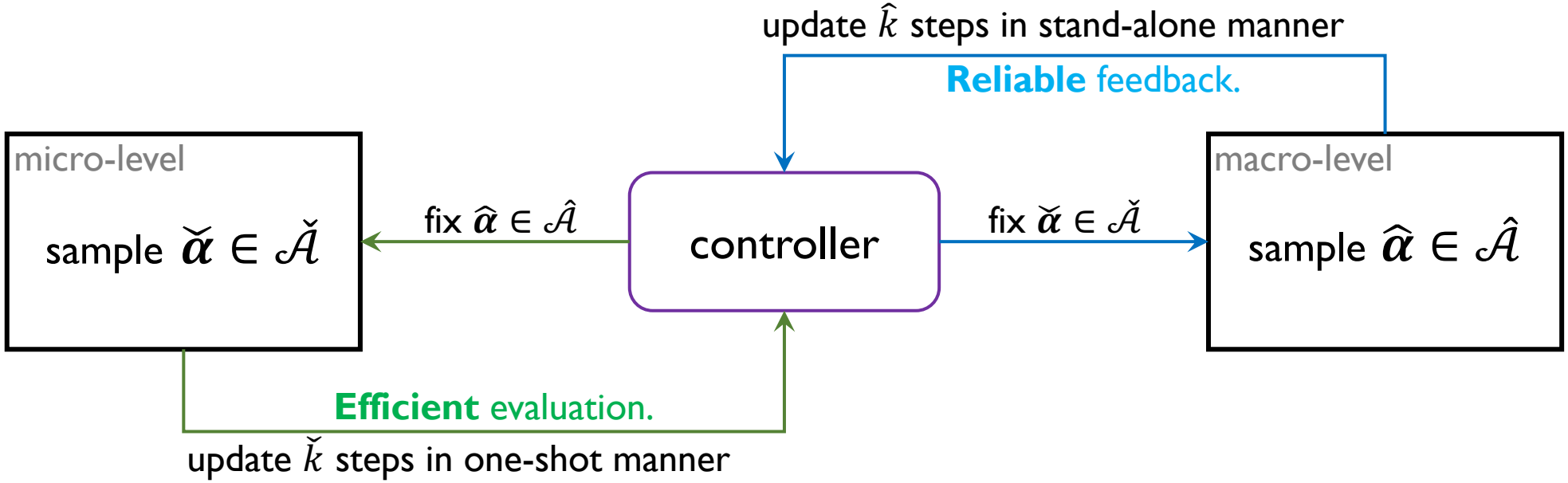
Algorithm 1 Proposed search recurrent architecture as the Interstellar algorithm.

Require: search space $\mathcal{A} \equiv \mathcal{A}_1 \cup \mathcal{A}_2$ in Figure 2, controller c for sampling $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_2, \boldsymbol{\alpha}_1]$.

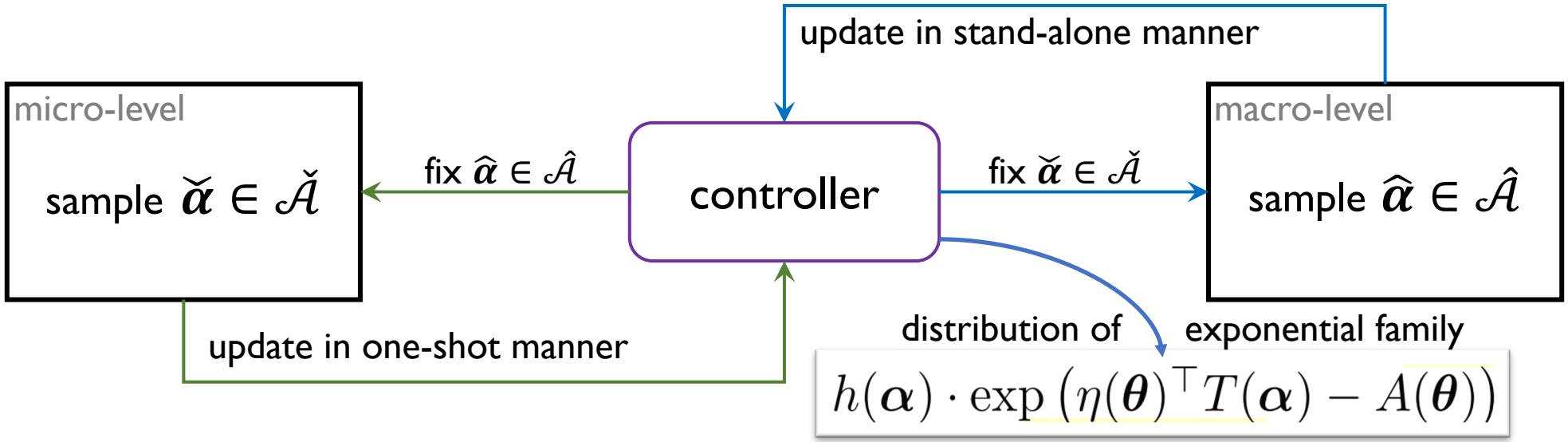
1: **repeat**
2: sample the *micro-level* architecture $\boldsymbol{\alpha}_2 \in \mathcal{A}_2$ by c;
3: update the controller c for k_1 steps using Algorithm 2 (in *stand-alone* manner);
4: sample the *macro-level* architecture $\boldsymbol{\alpha}_1 \in \mathcal{A}_1$ by c;
5: update the controller c for k_2 steps using Algorithm 3 (in *one-shot* manner);
6: **until** termination
7: Fine-tune the hyper-parameters for the best architecture $\boldsymbol{\alpha}^* = [\boldsymbol{\alpha}_1^*, \boldsymbol{\alpha}_2^*]$ sampled from c.
8: **return** $\boldsymbol{\alpha}^*$ and the fine-tuned hyper-parameters.



macro-level	connections	$\mathbf{h}_{t-1}, O_s, \mathbf{0}, \mathbf{s}_t$
$\hat{\boldsymbol{\alpha}} \in \hat{\mathcal{A}}$	combinators	$+, \odot, \otimes$, gated
micro-level	activation	identity, tanh, sigmoid
$\check{\boldsymbol{\alpha}} \in \check{\mathcal{A}}$	weight matrix	$\{\mathbf{W}_i\}_{i=1}^{6}, \mathbf{I}$



Interstellar – Search Algorithm



- Using policy gradient to optimize the controller

$$\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\alpha} \sim p_{\boldsymbol{\theta}}(\boldsymbol{\alpha})} \left[\mathcal{M}\left(f(\boldsymbol{F}^*; \boldsymbol{\alpha}), \mathcal{G}_{\text{val}}\right)\right].$$

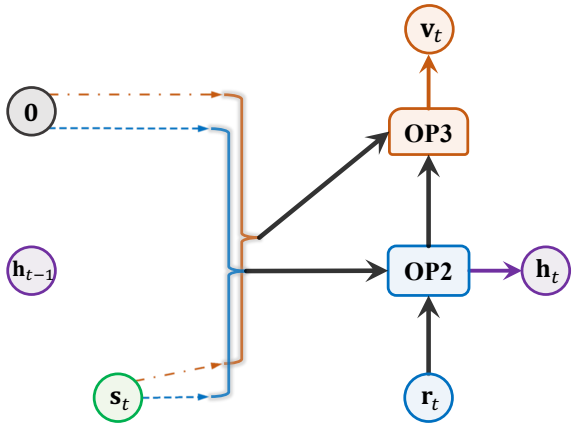
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \rho \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}\left[\mathcal{M}(f(F; \boldsymbol{\alpha}), \mathcal{G}_{\text{val}}) \nabla_{\boldsymbol{\theta}} \ln\left(p_{\boldsymbol{\theta}_t}(\boldsymbol{\alpha})\right)\right]$$

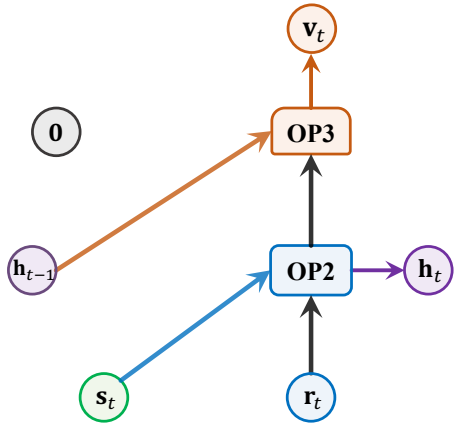
Outline

- Background
- Interstellar
  - Motivation
  - Method
  - Experiments
- Key takeaways and research directions

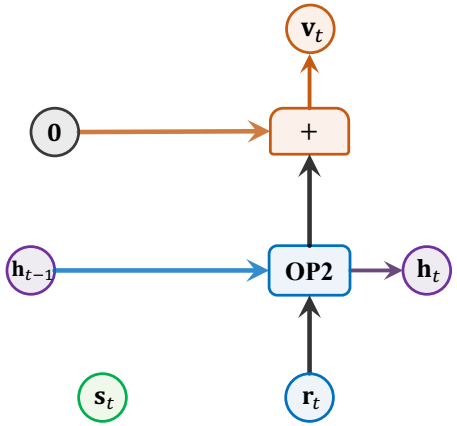
Interstellar – Experiments



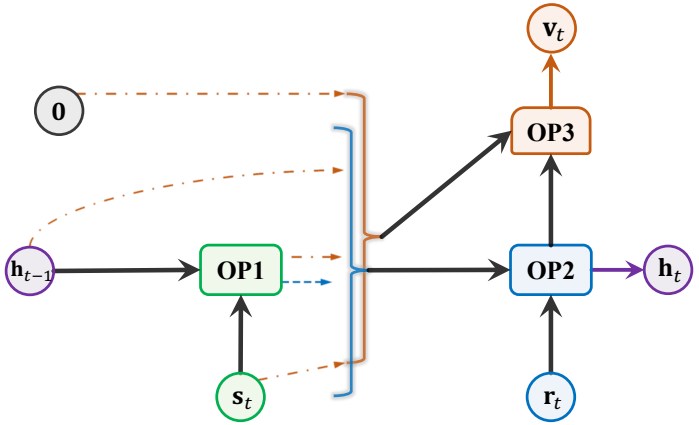
P1: single triplet



P2: two triples



P3: relation composition



P4: fully recurrent

data	tasks
S1	neighbor ∧ locatedin → locatedin locatedin ∧ locatedin → locatedin
S2	neighbor ∧ locatedin → locatedin
S3	neighbor ∧ locatedin ∧ locatedin → locatedin

harder | longer

Table 3: Performance on Countries dataset.

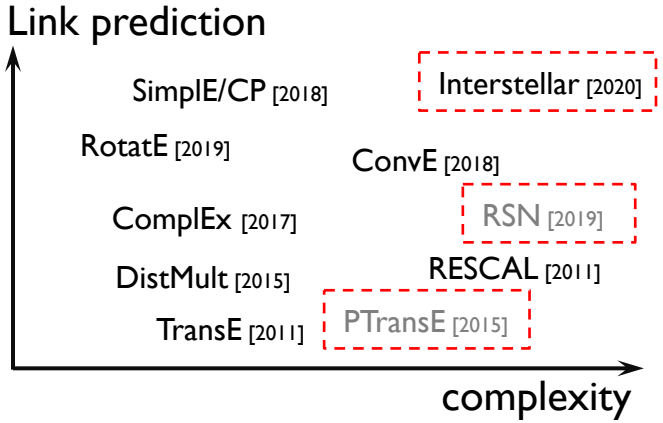
	S1	S2	S3
P1	0.998±0.001	0.997±0.002	0.933±0.031
P2	**1.000±0.000**	0.999±0.001	0.952±0.023
P3	0.992±0.001	**1.000±0.000**	0.961±0.016
P4	0.977±0.028	0.984±0.010	**0.964±0.015**
Interstellar	**1.000±0.000**	**1.000±0.000**	**0.968± 0.007**

Example
train: Serbia, neighbor, croatia
train: Serbia, locatedin, southern_europe
test: Serbia, locatedin, europe

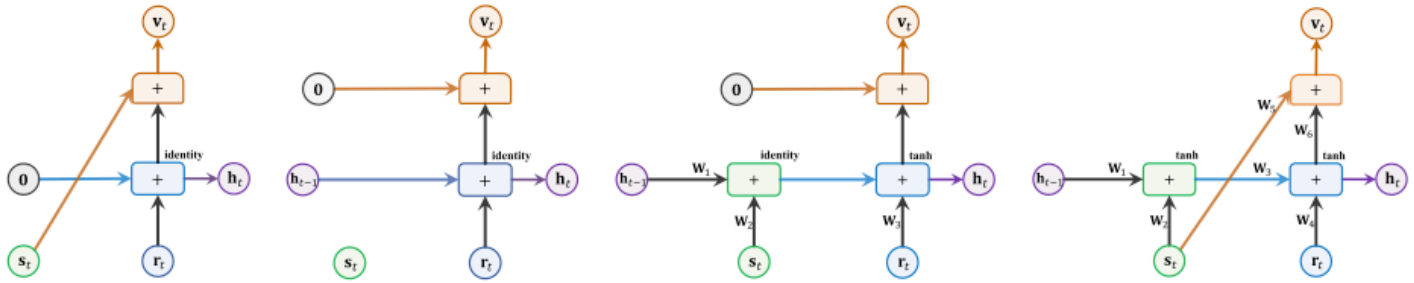
Model design should be task dependent 🤔

Interstellar – Link prediction task

models	WN18-RR			FB15k-237		
	H@1	H@10	MRR	H@1	H@10	MRR
TransE	12.5	44.5	0.18	17.3	37.9	0.24
ComplEx	41.4	49.0	0.44	22.7	49.5	0.31
RotatE*	43.6	54.2	0.47	**23.3**	50.4	**0.32**
R-GCN	-	-	-	15.1	41.7	0.24
PTransE	27.2	46.4	0.34	20.3	45.1	0.29
RSN	38.0	44.8	0.40	19.2	41.8	0.27
Interstellar	**44.0**	**54.8**	**0.48**	**23.3**	**50.8**	**0.32**

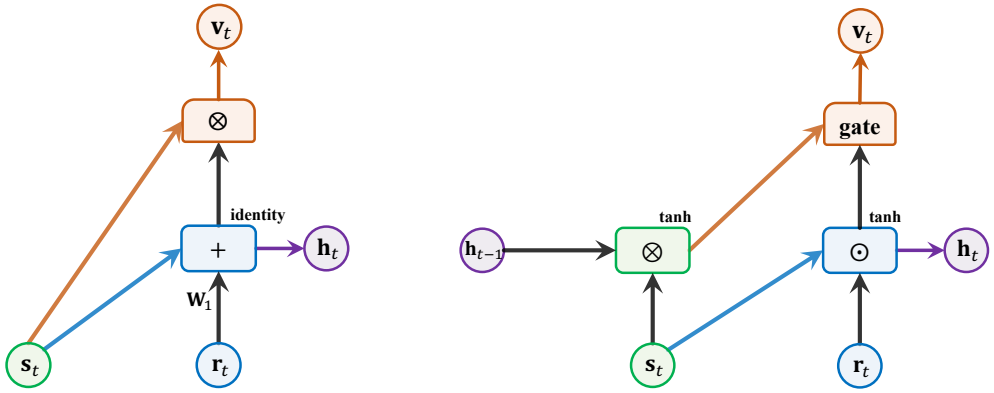


Existing models:



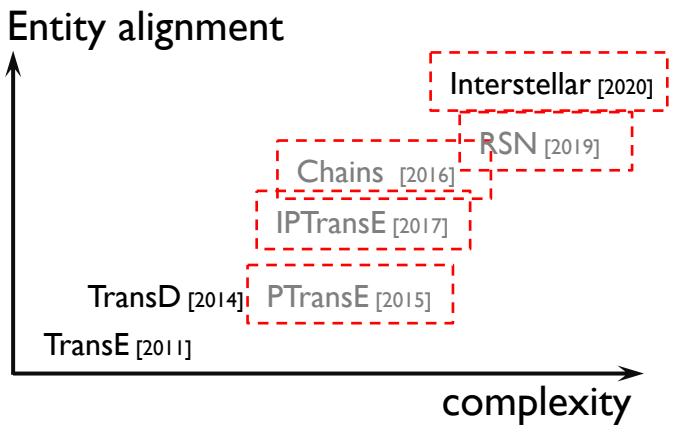
(a) TransE. (b) PTransE. (c) Chains. (d) RSN.

Searched models:



WN18RR FB15k-237

Interstellar – Entity alignment task



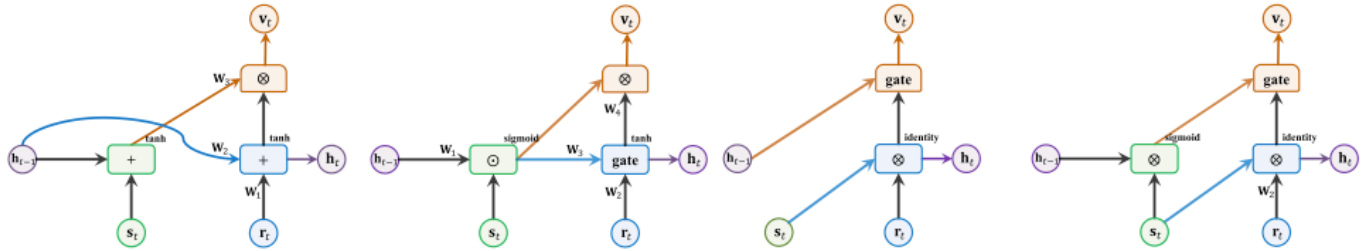
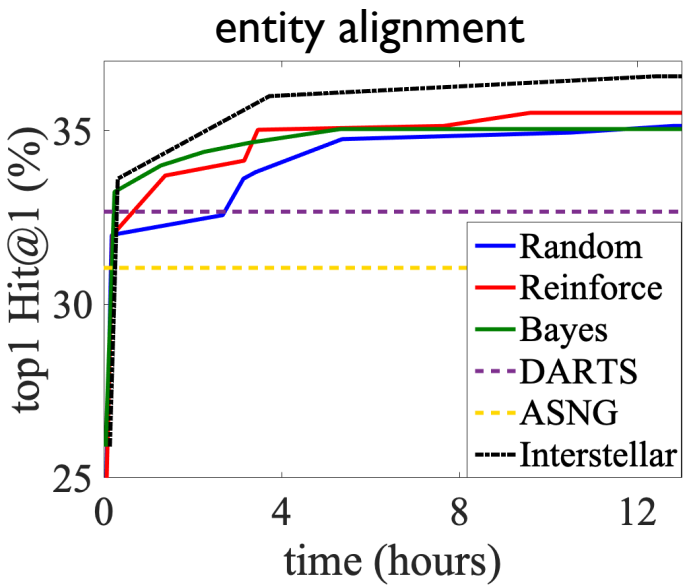


Figure 12: Graphical representation of the searched recurrent network f on each datasets in entity alignment task (Normal version).

models		DBP-WD			DBP-YG			EN-FR			EN-DE		
		H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
triplet	TransE	18.5	42.1	0.27	9.2	24.8	0.15	16.2	39.0	0.24	20.7	44.7	0.29
	TransD*	27.7	57.2	0.37	17.3	41.6	0.26	21.1	47.9	0.30	24.4	50.0	0.33
	BootEA*	32.3	63.1	0.42	31.3	62.5	0.42	31.3	62.9	0.42	44.2	70.1	0.53
GCN	GCN-Align	17.7	37.8	0.25	19.3	41.5	0.27	15.5	34.5	0.22	25.3	46.4	0.22
	VR-GCN	19.4	55.5	0.32	20.9	55.7	0.32	16.0	50.8	0.27	24.4	61.2	0.36
	R-GCN	8.6	31.4	0.16	13.3	42.4	0.23	7.3	31.2	0.15	18.4	44.8	0.27
path	PTransE	16.7	40.2	0.25	7.4	14.7	0.10	7.3	19.7	0.12	27.0	51.8	0.35
	IPTransE*	23.1	51.7	0.33	22.7	50.0	0.32	25.5	55.7	0.36	31.3	59.2	0.41
	Chains	32.2	60.0	0.42	35.3	64.0	0.45	31.4	60.1	0.41	41.3	68.9	0.51
	RSN*	38.8	65.7	0.49	40.0	67.5	0.50	34.7	63.1	0.44	48.7	72.0	0.57
	Interstellar	**40.7**	**71.2**	**0.51**	**40.2**	**72.0**	**0.51**	**35.5**	**67.9**	**0.46**	**50.1**	**75.6**	**0.59**

Interstellar – Efficiency



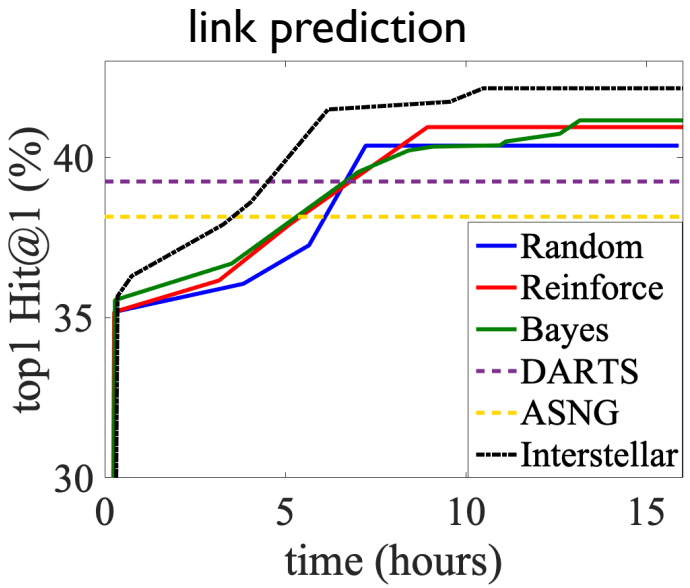


Table 6: Comparison of searching and fine-tuning time (in hours) in Algorithm 1.

procedure		entity alignment		link prediction	
		Normal	Dense	WN18-RR	FB15k-237
search	macro-level (line 2-3)	9.9±1.5	14.9±0.3	11.7±1.9	23.2±3.4
	micro-level (line 4-5)	4.2±0.2	7.5±0.6	6.3± 0.9	5.6±0.4
fine-tune (line 7)		11.6±1.6	16.2±2.1	44.3±2.3	67.6±4.5

Outline

- Background
- Interstellar
  - Motivation
  - Method
  - Experiments
- Key takeaways and research directions

Key takeaways and Research directions

- *Interstellar* analyzes the difficulty and importance of using the relational path to learn the short-term and long-term information in KG.

- *Interstellar* formulates the KGE task as a NAS problem and propose a domain-specific search space. Different from searching RNN cells, the recurrent network is specifically designed for KG tasks.
- This is the first work applying neural architecture search (NAS) methods on KG tasks.

- *Interstellar* identifies the problems of adopting stand-alone and one-shot search algorithms for our search space, and designs a hybrid-search algorithm to search efficiently.

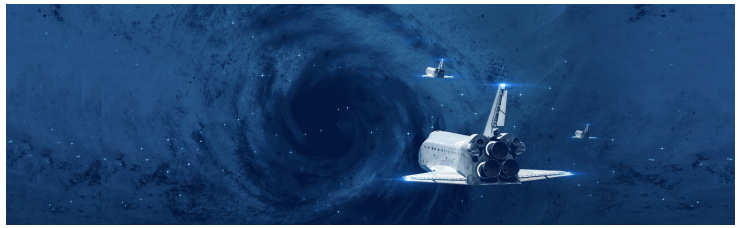
Contributions of *Interstellar*:

- Analyze the difficult and importance of learning from relational path
- Setup problem by NAS
  - Novel search space
  - Hybrid search algorithm

Key takeaways and Research directions

Towards Automated Knowledge Graph Embedding

- Analyze problems
  - high requirements of human intelligence
    - e.g., model design & training & fine-tuning
  - model performance is data-dependent
- Summary of existing models → General search space “拆解”
  - cover existing models
- References of existing search algorithms → Improved algorithm “组装”
  - efficient and accurate evaluation
- Combine the search space and algorithm → Better performance/adaptivity
  - trade off between exploration and exploitation



Key takeaways and Research directions

General effectiveness and efficiency evaluation results

		FB15k				WN18				FB15k-237				WN18RR				YAGO3-10			
		H@1	H@10	MR	MRR	H@1	H@10	MR	MRR	H@1	H@10	MR	MRR	H@1	H@10	MR	MRR	H@1	H@10	MR	MRR
Tensor Decomposition Models	DistMult	73.61	86.32	173	0.784	72.60	94.61	675	0.824	22.44	49.01	199	0.313	39.68	50.22	5913	0.433	41.26	66.12	1107	0.501
	ComplEx	81.56	90.53	34	0.848	94.53	95.50	3623	0.949	25.72	52.97	202	0.349	42.55	52.12	4907	0.458	50.48	70.35	1112	0.576
	ANALOGY	65.59	83.74	126	0.726	92.61	94.42	808	0.934	12.59	35.38	476	0.202	35.82	38.00	9266	0.366	19.21	45.65	2423	0.283
	SimplE	66.13	83.63	138	0.726	93.25	94.58	759	0.938	10.03	34.35	651	0.179	38.27	42.65	8764	0.398	35.76	63.16	2849	0.453
	HolE	75.85	86.78	211	0.800	93.11	94.94	650	0.938	21.37	47.64	186	0.303	40.28	48.79	8401	0.432	41.84	65.19	6489	0.502
	TuckER	72.89	88.88	39	0.788	94.64	95.80	510	0.951	25.90	53.61	162	0.352	42.95	51.40	6239	0.459	46.56	68.09	2417	0.544
Geometric Models	TransE	49.36	84.73	45	0.628	40.56	94.87	279	0.646	21.72	49.65	209	0.31	2.79	49.52	3936	0.206	40.57	67.39	1187	0.501
	STransE	39.77	79.60	69	0.543	43.12	93.45	208	0.656	22.48	49.56	357	0.315	10.13	42.21	5172	0.226	3.28	7.35	5797	0.049
	CrossE	60.08	86.23	136	0.702	73.28	95.03	441	0.834	21.21	47.05	227	0.298	38.07	44.99	5212	0.405	33.09	65.45	3839	0.446
	TorusE	68.85	83.98	143	0.746	94.33	95.44	525	0.947	19.62	44.71	211	0.281	42.68	53.35	4873	0.463	27.43	47.44	19455	0.342
	RotatE	73.93	88.10	42	0.791	94.30	96.02	274	0.949	23.83	53.06	178	0.336	42.60	57.35	3318	0.475	40.52	67.07	1827	0.498
Deep Learning Models	ConvE	59.46	84.94	51	0.688	93.89	95.68	413	0.945	21.90	47.62	281	0.305	38.99	50.75	4944	0.427	39.93	65.75	2429	0.488
	ConvKB	11.44	40.83	324	0.211	52.89	94.89	202	0.709	13.98	41.46	309	0.230	5.63	52.50	3429	0.249	32.16	60.47	1683	0.420
	ConvR	70.57	88.55	70	0.773	94.56	95.85	471	0.950	25.56	52.63	251	0.346	43.73	52.68	5646	0.467	44.62	67.33	2582	0.527
	CapsE	1.93	21.78	610	0.087	84.55	95.08	233	0.890	7.34	35.60	405	0.160	33.69	55.98	720	0.415	0.00	0.00	60676	0.000
	RSN	72.34	87.01	51	0.777	91.23	95.10	346	0.928	19.84	44.44	248	0.280	34.59	48.34	4210	0.395	42.65	66.43	1339	0.511
	AnyBURL	81.09	87.86	288	0.835	94.63	95.96	233	0.951	24.03	48.93	480	0.324	44.93	55.97	2530	0.485	45.83	66.07	815	0.528

Family and Group		Model
Tensor Decomposition Models	Bilinear	DistMult
		ComplEx
		Analogy
		SimplE
	Non-bilinear	HolE
		TuckER
Geometric Models	Pure Translation	TransE
	Additional Embeddings	STransE
		CrossE
	Roto-translation	TorusE
		RotatE
Deep Learning Models	Convolution	ConvE
		ConvKB
		ConvR
	Capsule	CapsE
	Recurrent	RSN

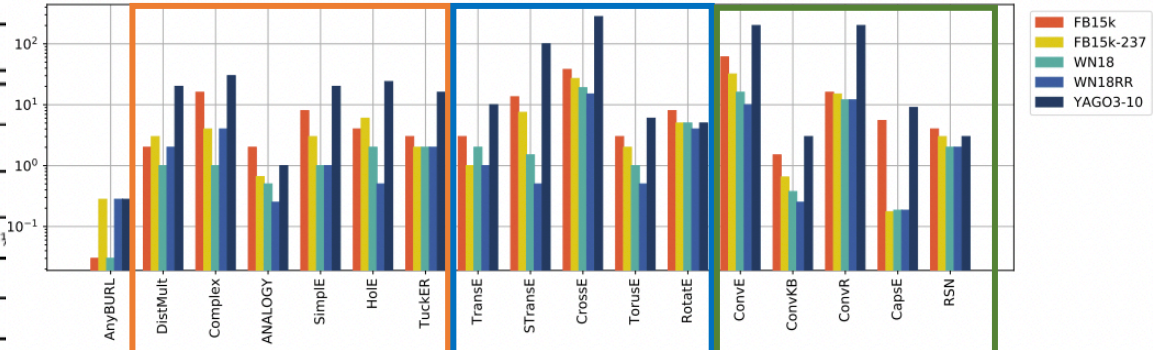


Fig. 5. Training times in hours for each LP model on each dataset. Y axis is in logscale.

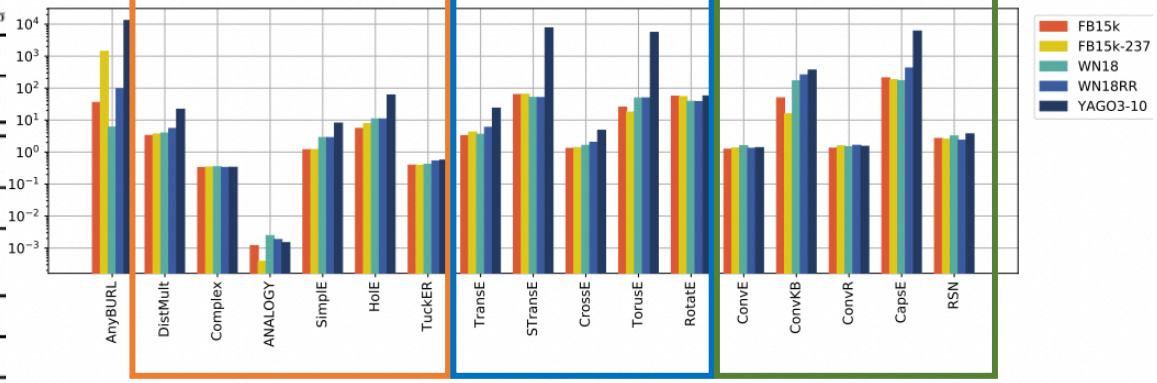


Fig. 6. Prediction times in milliseconds for each LP model on each dataset. Y axis is in logscale.

No best models 🤔

Key takeaways and Research directions

OPEN GRAPH BENCHMARK

Future research directions

- Conduct experiments on large-scale datasets
  - Open Graph Benchmark

Name	Package	#Nodes	#Edges*
ogbl-citation2	>=1.2.4	2,927,963	30,561,187
ogbl-wikikg2	>=1.2.4	2,500,604	17,137,181
ogbl-biokg	>=1.2.0	93,773	5,088,434

- Model architecture design
  - Leverage structural information such as path or subgraph
  - Capture transferability among datasets and models

- Benchmarking the KGE models
  - Insightful way to analyze model performance
  - Standard approach to evaluate the different models
  - Efficient hyper-parameter optimization method
  - Involving more KG tasks

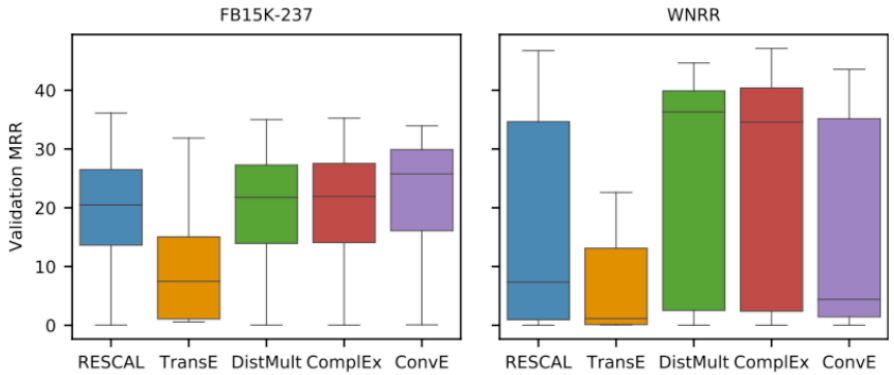
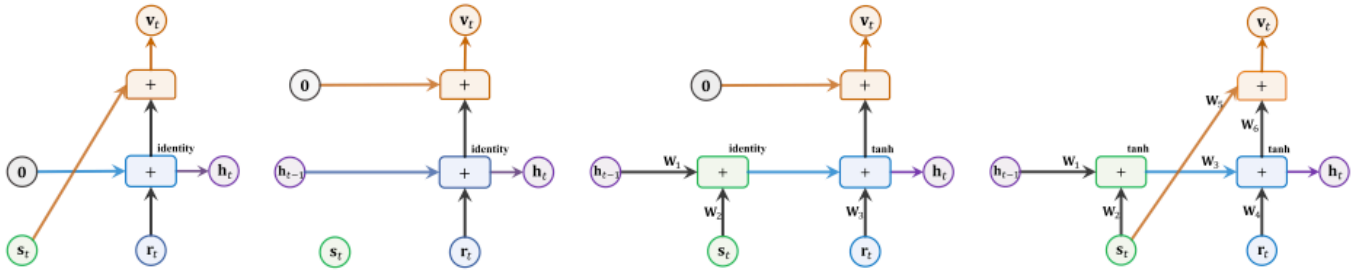


Figure 1: Distribution of filtered MRR (%) on validation data over the quasi-random hyperparameter configurations explored in our study.

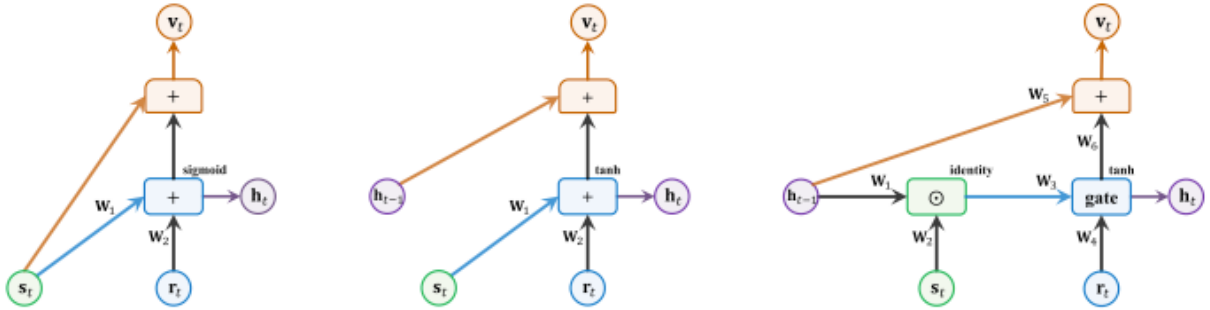
Q&A

Thanks for your attention!

Appendix

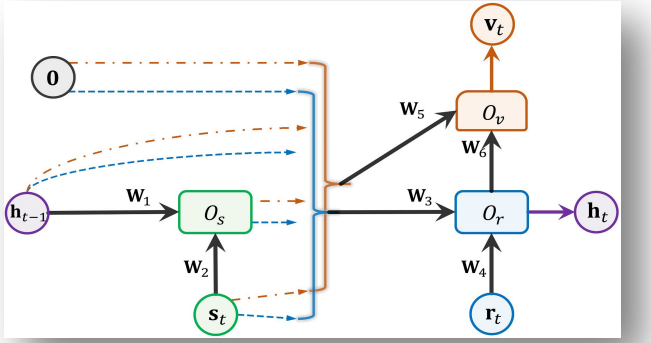


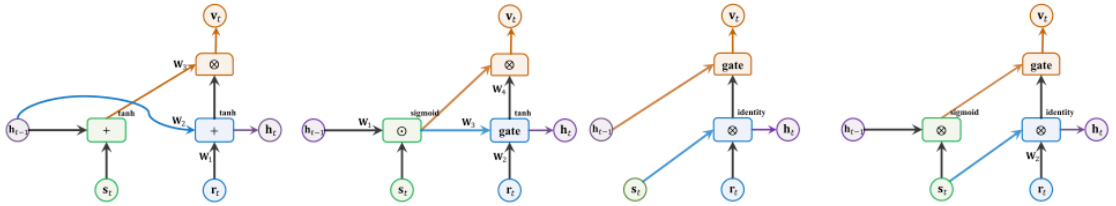
(a) TransE. (b) PTransE. (c) Chains. (d) RSN.



(a) S1 (b) S2 (c) S3

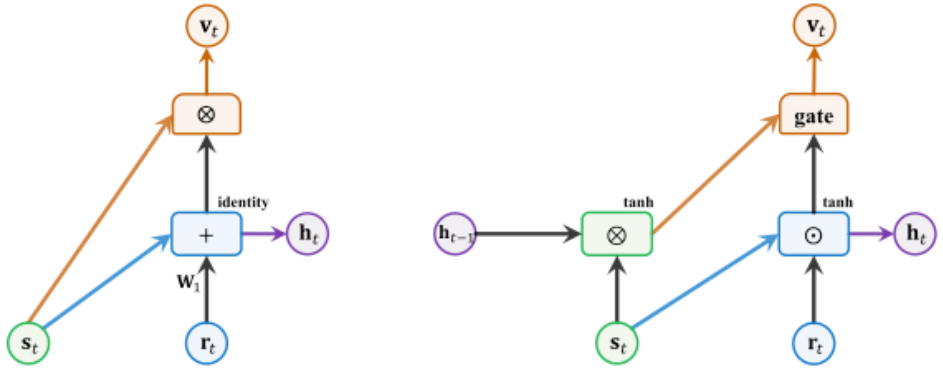
Figure 11: Graphical representation of the searched f on countries dataset.





(a) DBP-WD (b) DBP-YG (c) EN-FR (d) EN-DE

Figure 12: Graphical representation of the searched recurrent network f on each datasets in entity alignment task (Normal version).



(a) WN18-RR (b) FB15k-237

Appendix

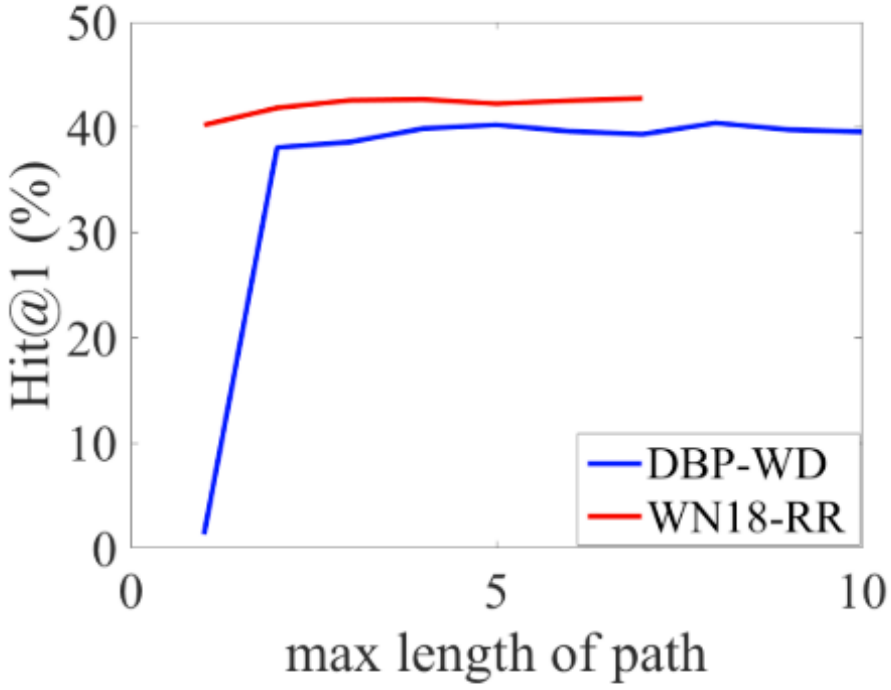


Figure 9: The influence of path length.

long-term information is important for the entity alignment task
while short-term information is important for link prediction task.

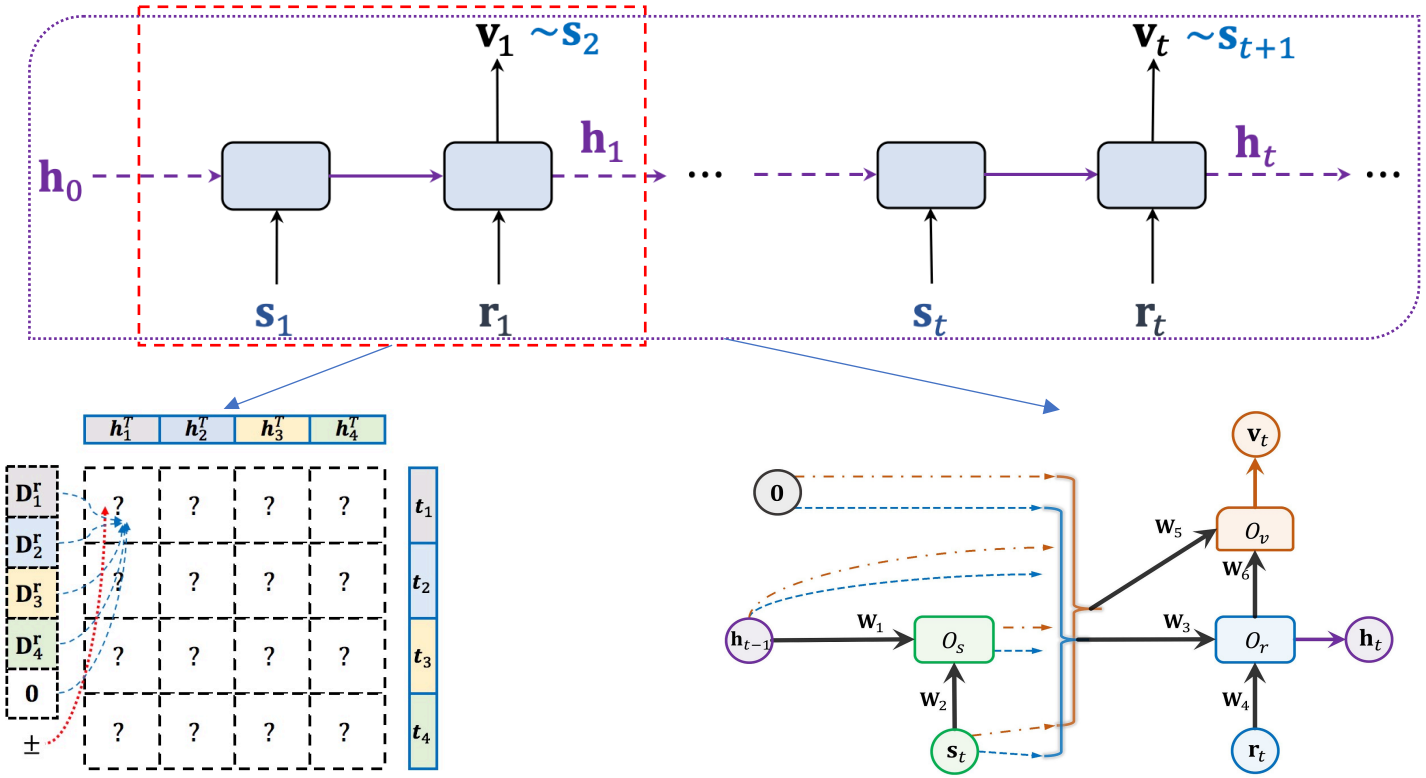
Appendix

Calculation of policy gradient

$$
\begin{aligned}
\nabla_\theta E_x[f(x)] &= \nabla_\theta \sum_x p(x) f(x) && \text{definition of expectation} \\
&= \sum_x \nabla_\theta p(x) f(x) && \text{swap sum and gradient} \\
&= \sum_x p(x) \frac{\nabla_\theta p(x)}{p(x)} f(x) && \text{both multiply and divide by } p(x) \\
&= \sum_x p(x) \nabla_\theta \log p(x) f(x) && \text{use the fact that } \nabla_\theta \log(z) = \frac{1}{z} \nabla_\theta z \\
&= E_x[f(x) \nabla_\theta \log p(x)] && \text{definition of expectation}
\end{aligned}
$$

Appendix

- AutoSF + Interstellar

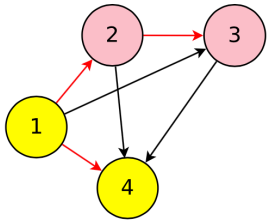


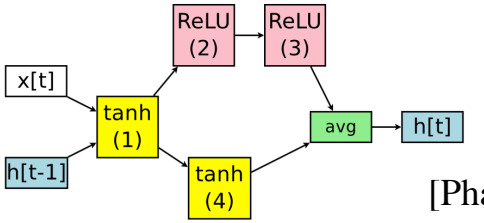
- Logic rule (e.g. DRUM) + Interstellar
- Subgraph processor for KG

Appendix

Search space:

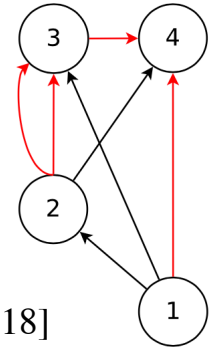
RNN

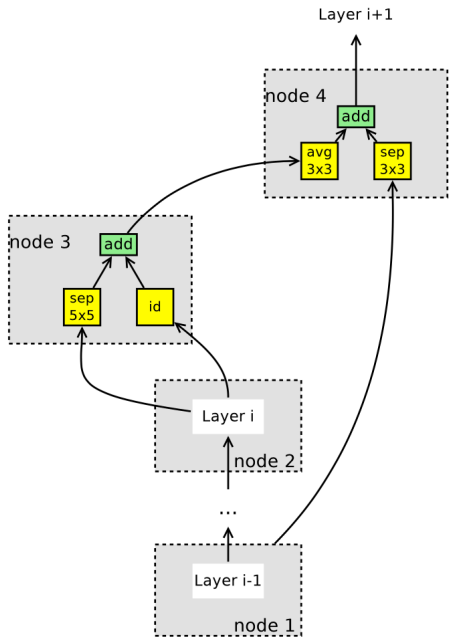




[Pham et. al. 2018]

CNN





...

Search algorithm

- Random, Reinforce, Bayes, Gradient-descent, etc

Evaluation method

- Stand-alone: train and evaluate separately
- One-shot: supernet with parameter-sharing