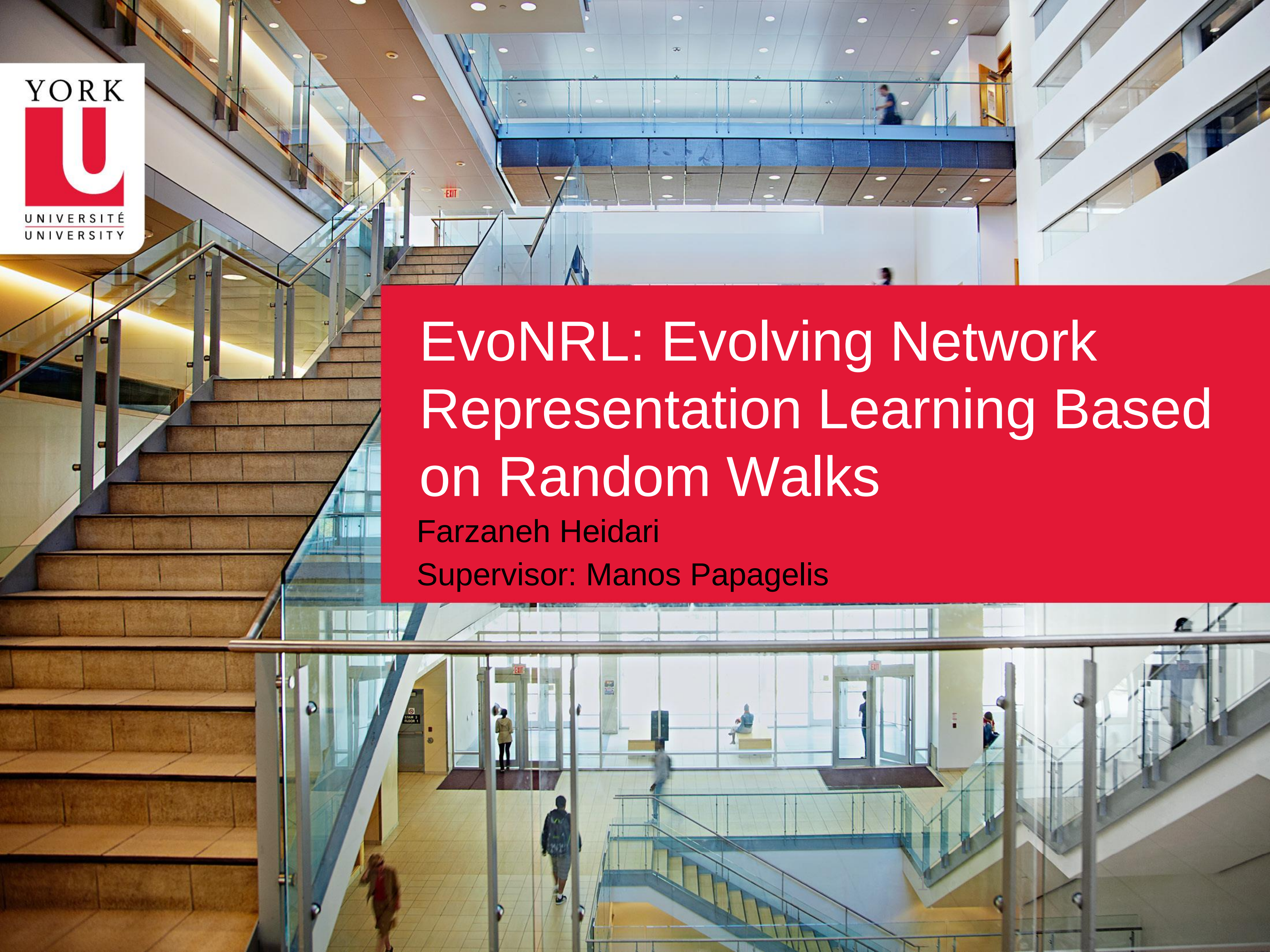
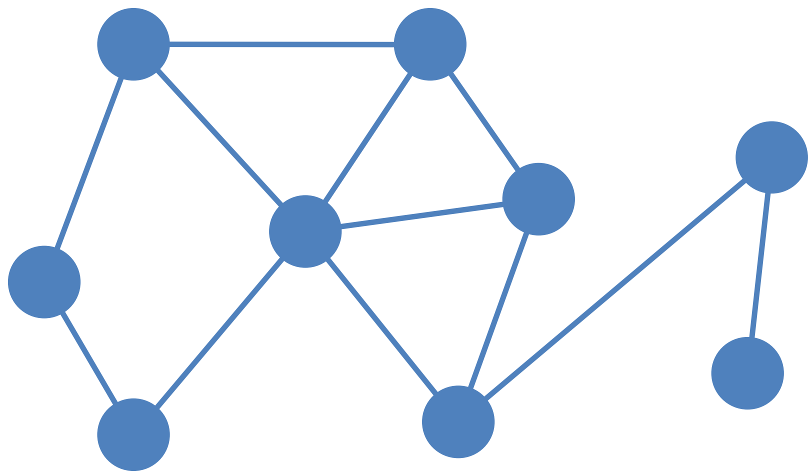


EvoNRL: Evolving Network Representation Learning Based on Random Walks

Farzaneh Heidari

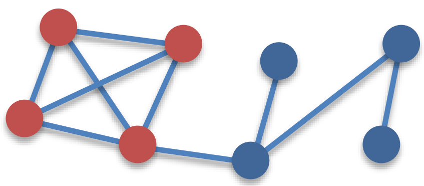
Supervisor: Manos Papagelis



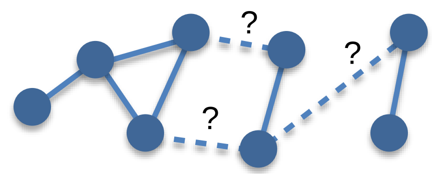
networks

(universal language for describing complex data)

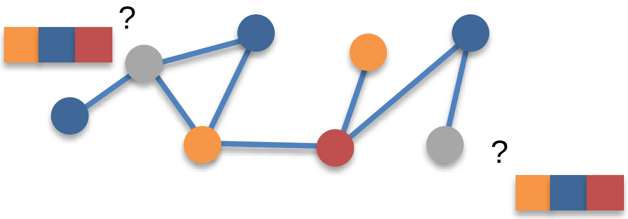
Classical ML Tasks in Networks



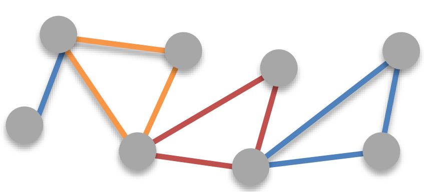
community detection



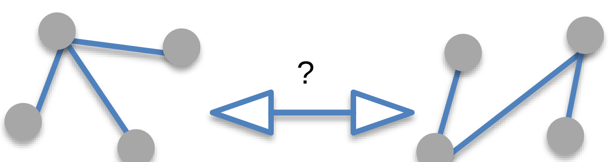
link prediction



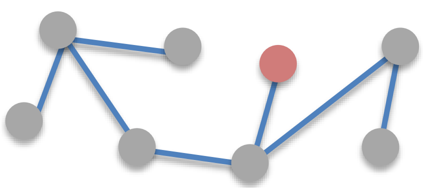
node classification



triangle count



graph similarity

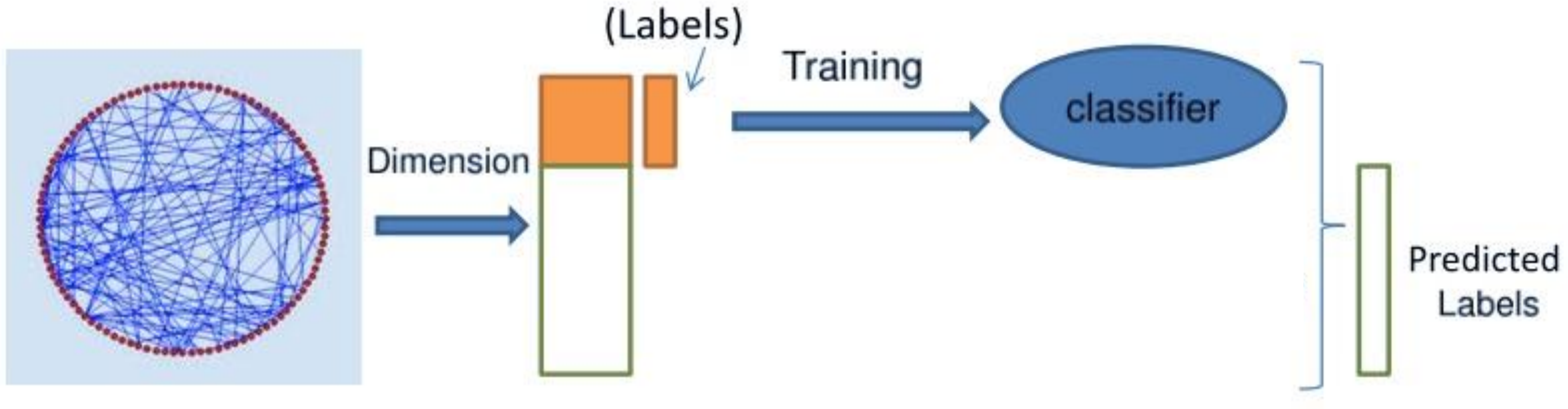


anomaly detection

Limitations of Classical ML Tasks

expensive computation

(high dimension computations)



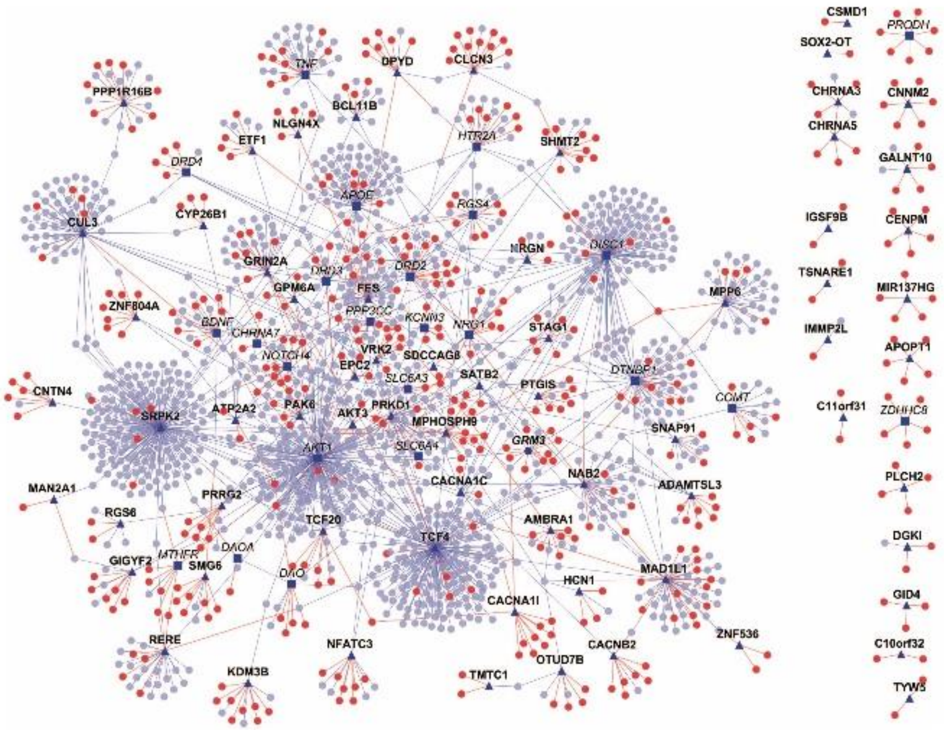
Matrix Factorization

Tang, Lei, and Huan Liu. "Relational learning via latent social dimensions." *Proceedings of the 15th ACM SIGKDD inte*

Limitations of Classical ML Tasks

extensive domain knowledge

(task specific)



Ganapathiraju, Madhavi K., et al. "Schizophrenia interactome with 504 novel protein–protein interactions." *npj Schizo*

Network Representation Learning (NRL)

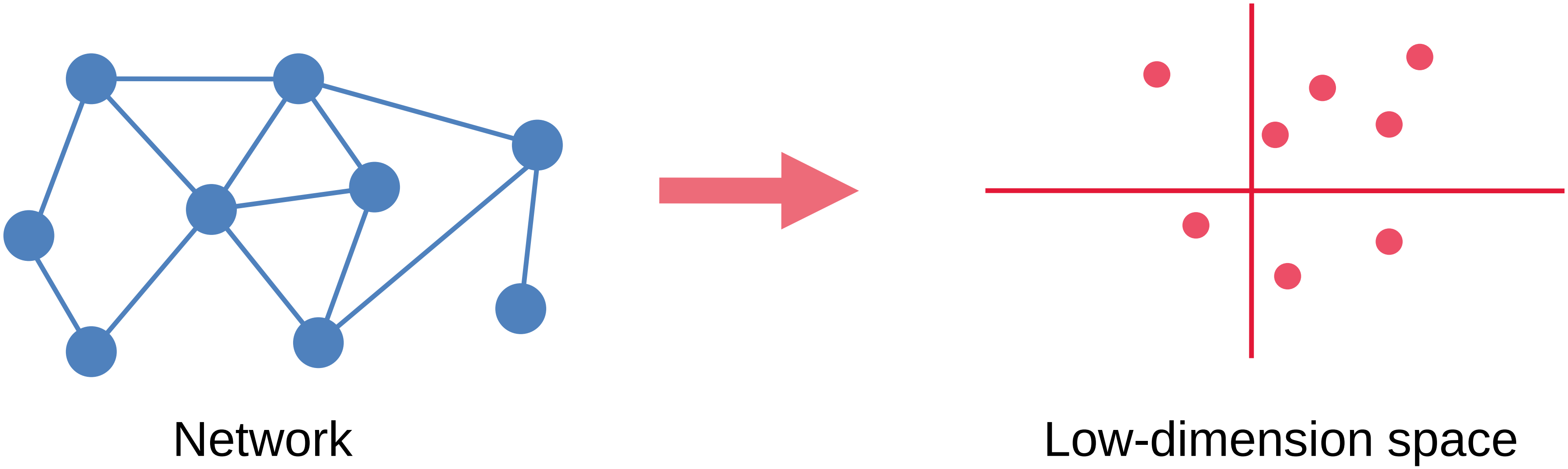
faster computations

(low dimension computations)

agnostic domain knowledge

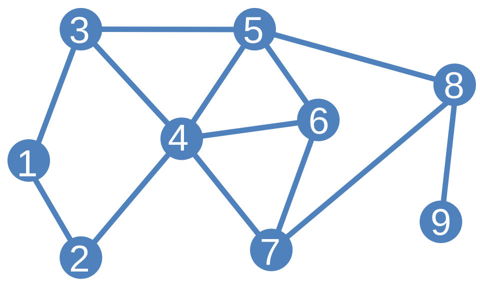
(task independent)

Network Representation Learning (NRL)

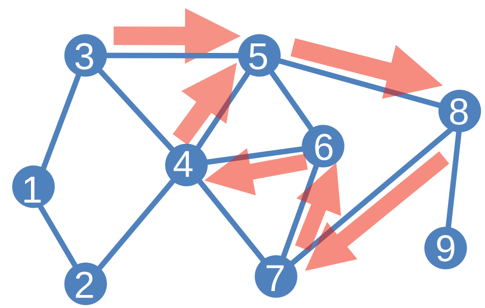


several network structural properties can be learned/embedded
(nodes, edges, subgraphs, graphs, …)

Random Walk-based NRL



Input network

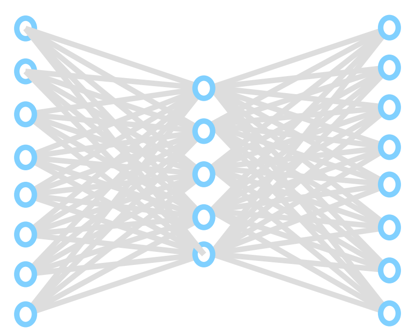


Obtain a set of random walks

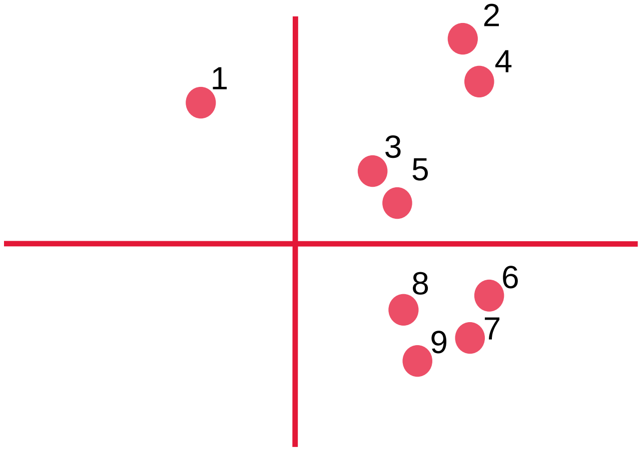
3 5 8 7 6 4 5

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

Treat the set of random walks as sentences



Feed sentences to Skip-gram NN model



Learn a vector representation for each node

Random Walk-based NRL

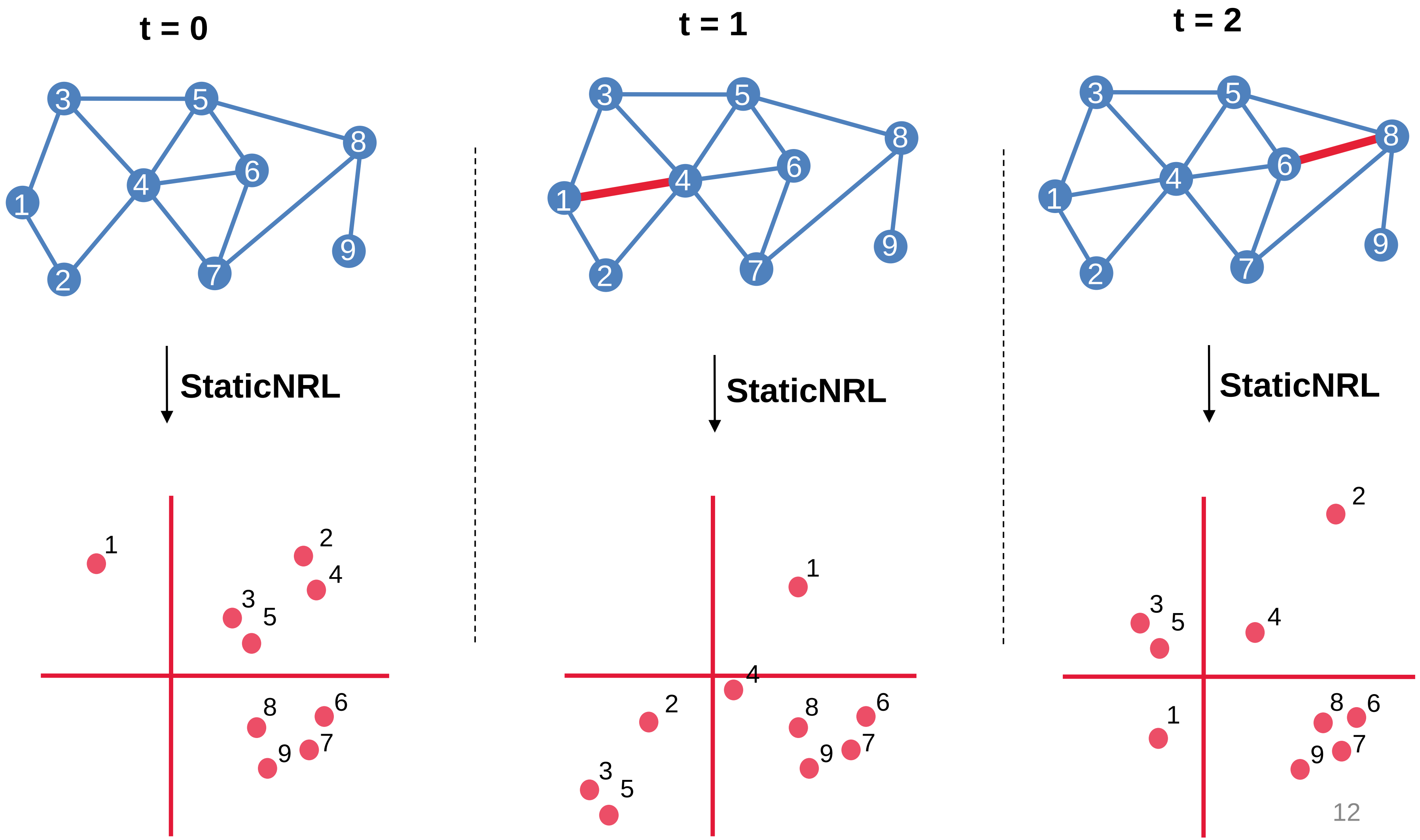
StaticNRL

DeepWalk
node2vec
…

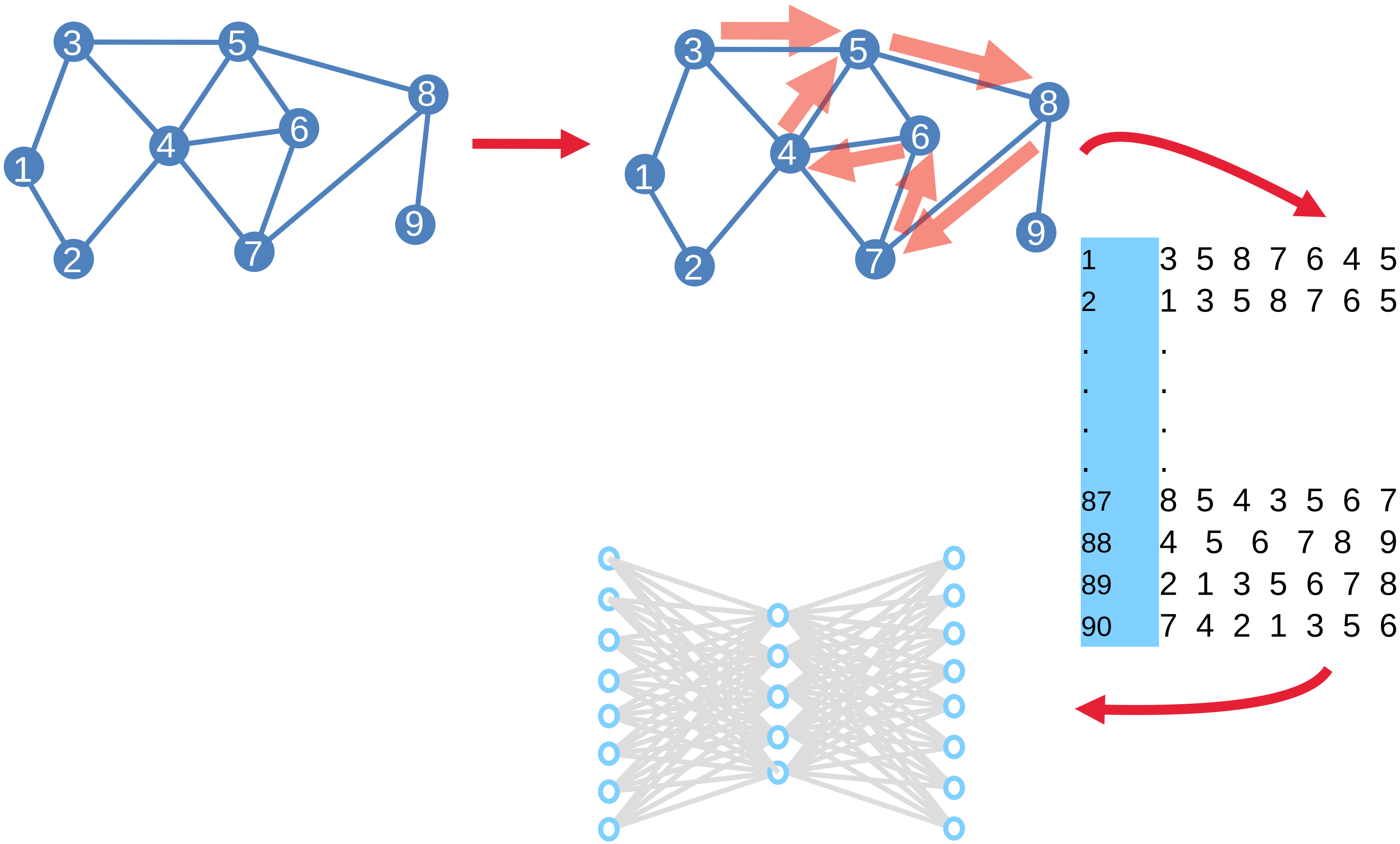
But…

real-world networks are constantly changing how can we learn representations of an evolving network?

Naive Approach

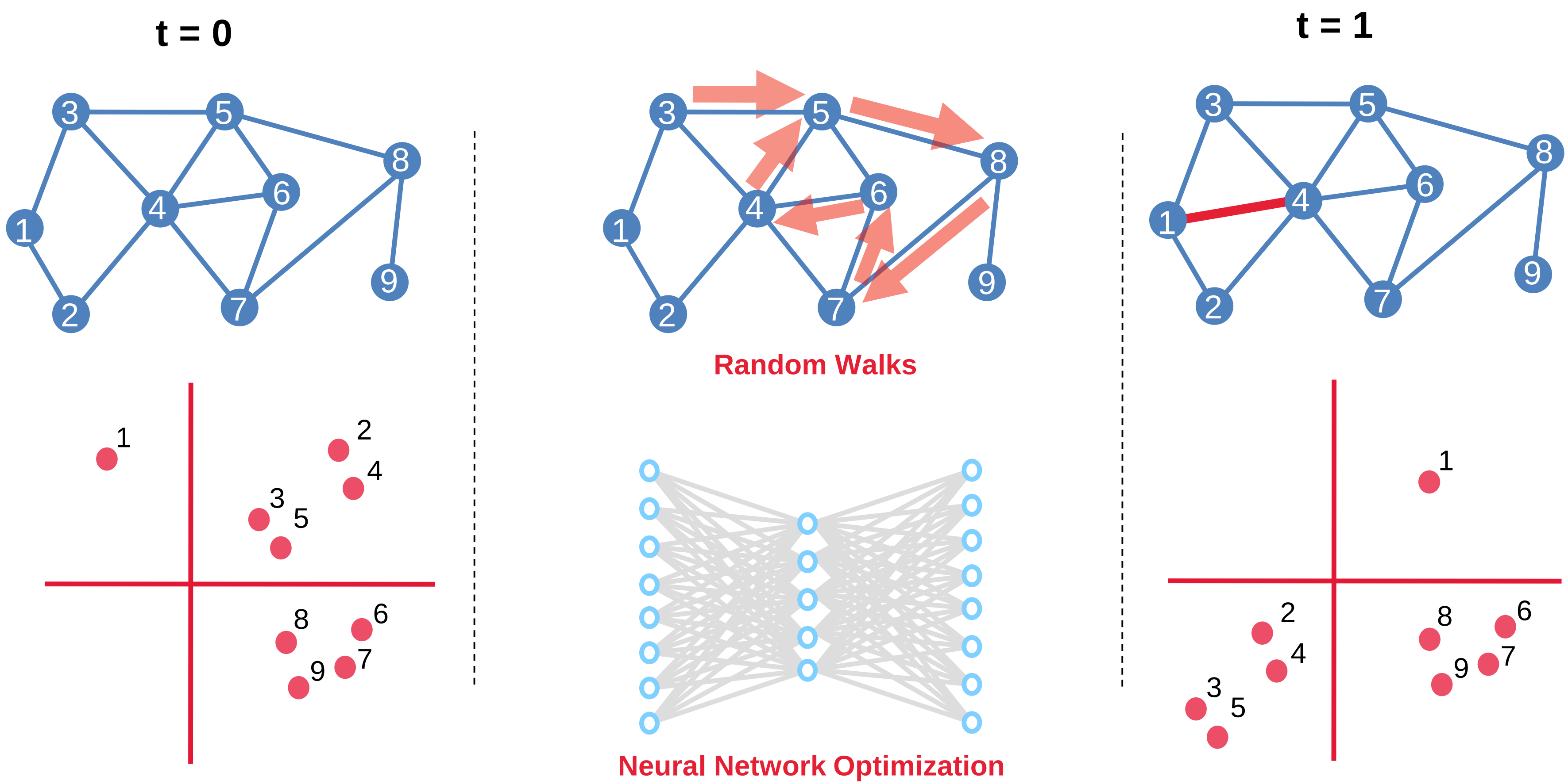


Limitation #1



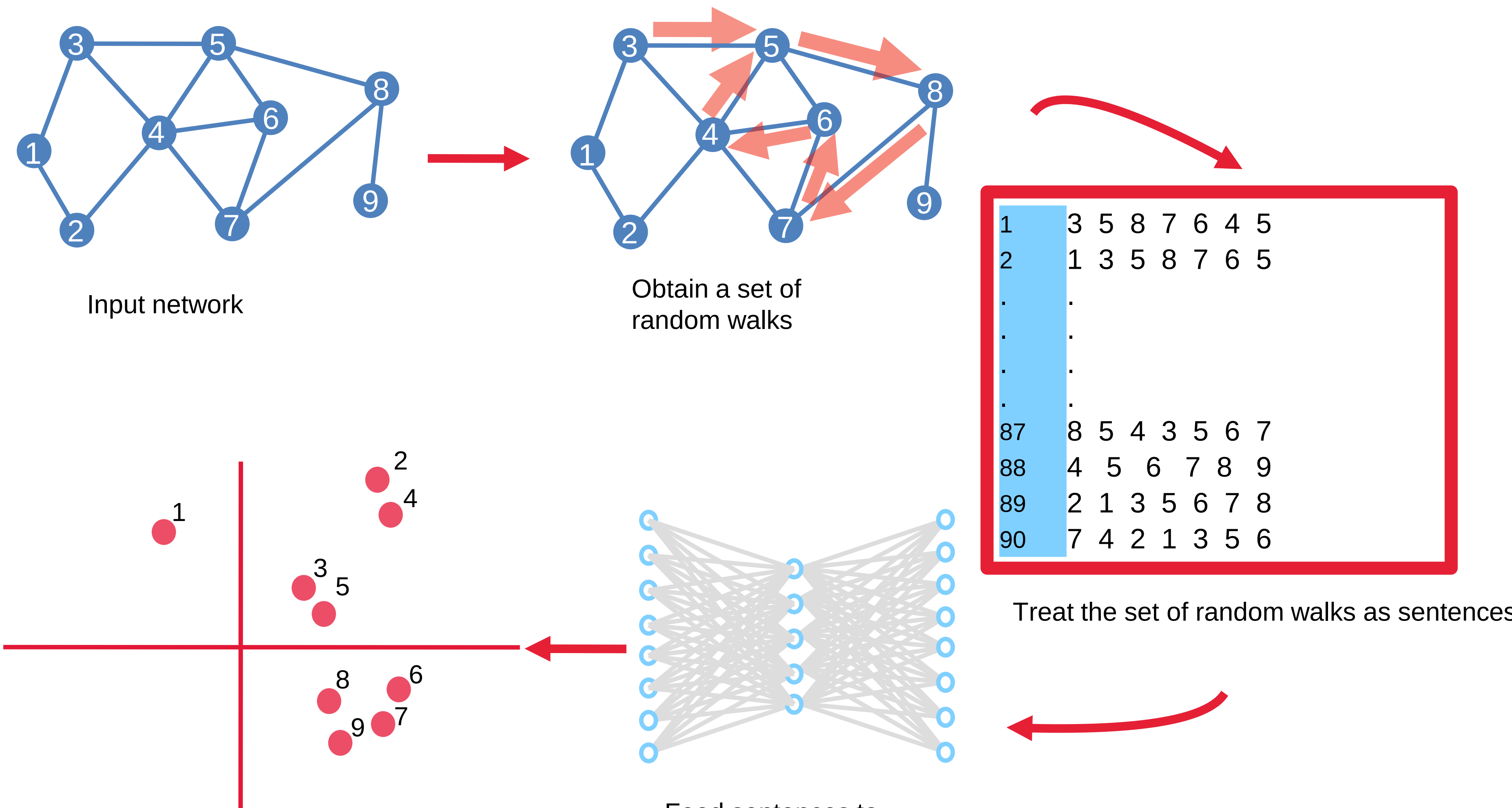
time expensive

Limitation #2



incomparable representations

EvoNRL Key Idea



Input network

Obtain a set of
random walks

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

Treat the set of random walks as sentences

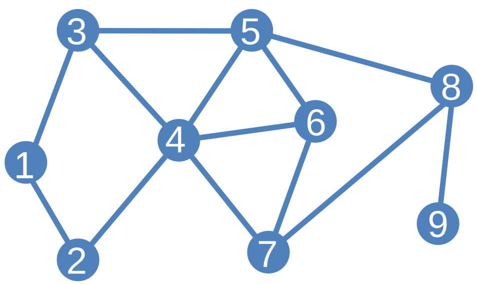
Feed sentences to
Skip-gram NN model

Learn a vector representation
for each node

dynamically maintain a set of random walks for every change in the network

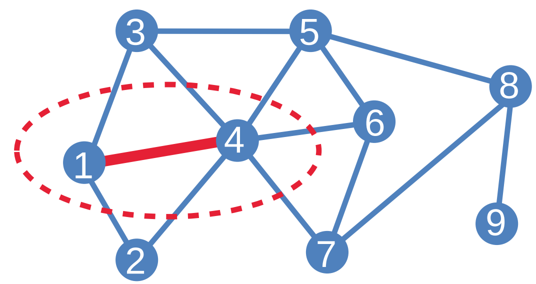
Example

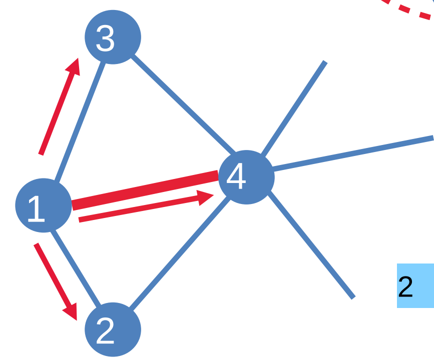
t = 0



addition of edge (1, 4)

t = 1





simulate the rest of the RW

2	1	4	3	5	6	7	8

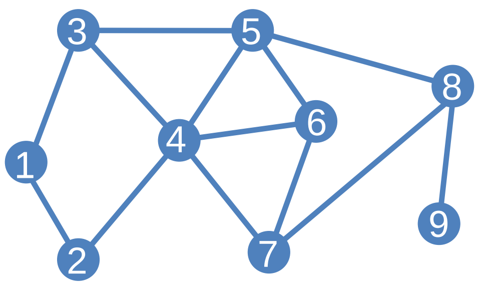
1	3	5	8	7	6	4	5
2	1	3	5	8	7	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	9	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

need to update the RW set

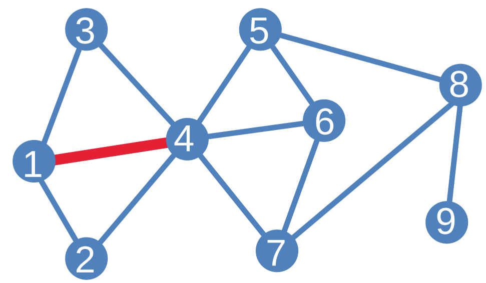
1	3	5	8	7	6	4	5
2	1	3	5	8	7	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	9	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

how can we efficiently maintain a set of random walks?

EvoNRL Operations: edge addition



+ edge(n_1, n_2)



2 | 1 **4** $\boxed{3\ 5\ 6\ 7\ 8}$

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

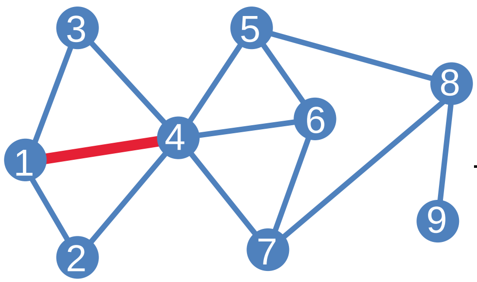
Operations on RW

Search a node

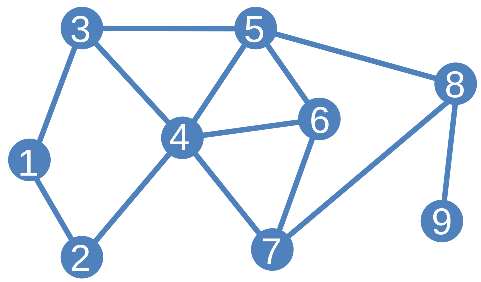
Delete a RW

Insert a new RW

EvoNRL Operations: edge deletion



- edge(n_1, n_2)



1	3 5 8 7 6 4 5
2	1 4 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 4 1 5 6 7 8
90	7 4 2 1 3 5 6

2 **1** $\boxed{3\ 4\ 5\ 6\ 7\ 8}$

1	3 5 8 7 6 4 5
2	1 4 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 4 1 5 6 7 8
90	7 4 2 1 3 5 6

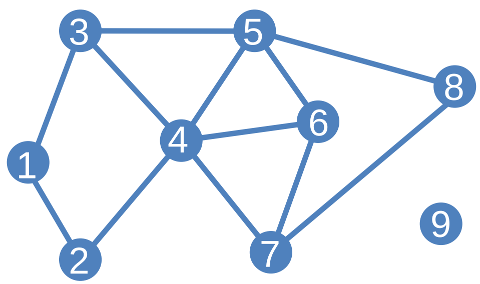
Operations on RW

Search two nodes

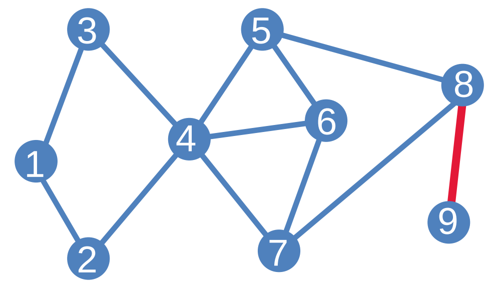
Delete a RW

Insert a new RW

EvoNRL Operations: node addition



+ node(n_1)



2	1 4 8 **9** $\boxed{8\ 7\ 6}$

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 7
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 7 4 3 5 6 7
88	4 5 6 7 8 5
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6
91	9 8 7 4 7 6 5
.	9 . . .
.	9 . . .
100	9 8 5 6 4 3 1

Operations on RW

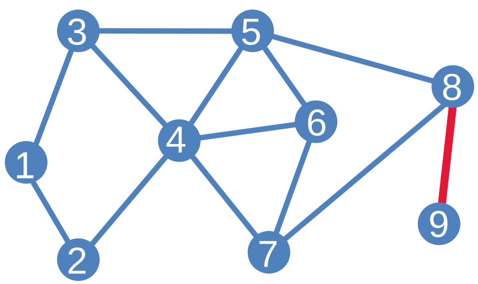
Search a node (node #8)

Delete a RW

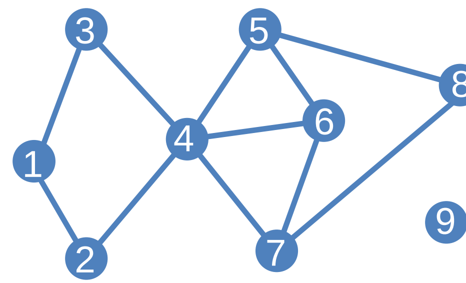
Insert a new RW

Append the RW list

EvoNRL Operations: node deletion



- node(n_1)



2	1	4	8	5	8	7	6

1	3	5	8	7	6	4	5
2	1	3	5	8	9	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	7	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

1	3	5	8	7	6	4	5
2	1	3	5	8	9	6	5
.	.						
.	.						
.	.						
.	.						
87	8	7	4	3	5	6	7
88	4	5	6	7	8	5	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6
91	9	8	7	4	7	6	5
.	9	.	.	.			
.	9	.	.	.			
100	9	8	5	6	4	3	1

Operations on RW

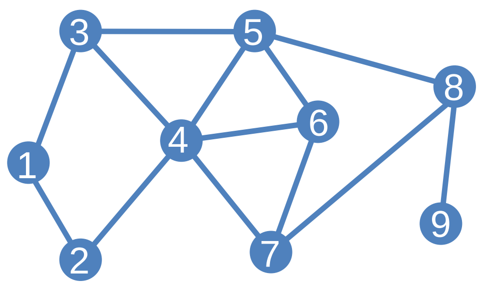
Search two nodes

Delete a RW

Insert a new RW

Deduct the RW list

EvoNRL Indexing



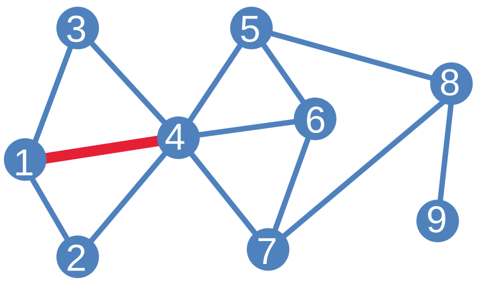
1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

each node is **a keyword**
each RW is **a document**
a set of RWs is **a collection of documents**

Term	Frequency	Postings and Positions
1	3	< 2, 1 >, < 89, 2 >, < 90, 4 >
2	2	<89, 1>, <90, 3>
3	5	<1, 1>, <2, 1>, <87, 3>, <89, 3>, <90, 5>
4	4	<1, 6>, <87, 3>, <90, 2>
5	9	<1, 2>, <1, 7>, <2, 3>, <2, 7>, <87, 5>, <88, 2>, <89, 4>, <90, 6>
6	6	<1, 5>, <2, 6>, <87, 6>, <88, 3>, <89, 3>, <90, 5>
7	5	<1, 4>, <2, 5>, <87, 7>, <88, 4>, <89, 6>, 90, 7>
8	5	<1, 3>, <2, 4>, <87, 1>, <88, 6>, <89, 7>
9	1	<88, 7>

when is the good time to obtain a new representation of the network?

Arriving Edge Importance

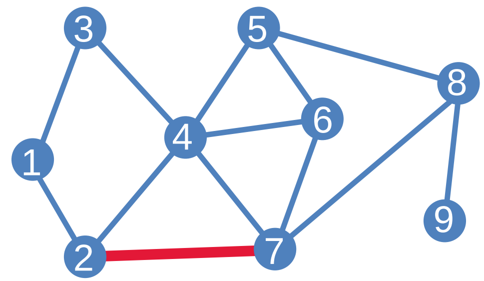


edge addition

1	3	5	8	7	6	4	5
2	1	3	5	8	7	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	7	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

count #RW changed

2



edge addition

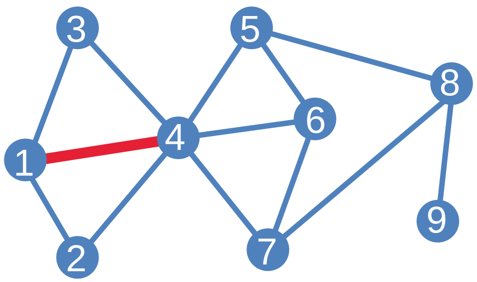
1	3	5	8	7	6	4	5
2	1	3	5	8	7	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	7	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

count #RW changed

4

count the #RW
each edge changes

Adaptive Algorithm

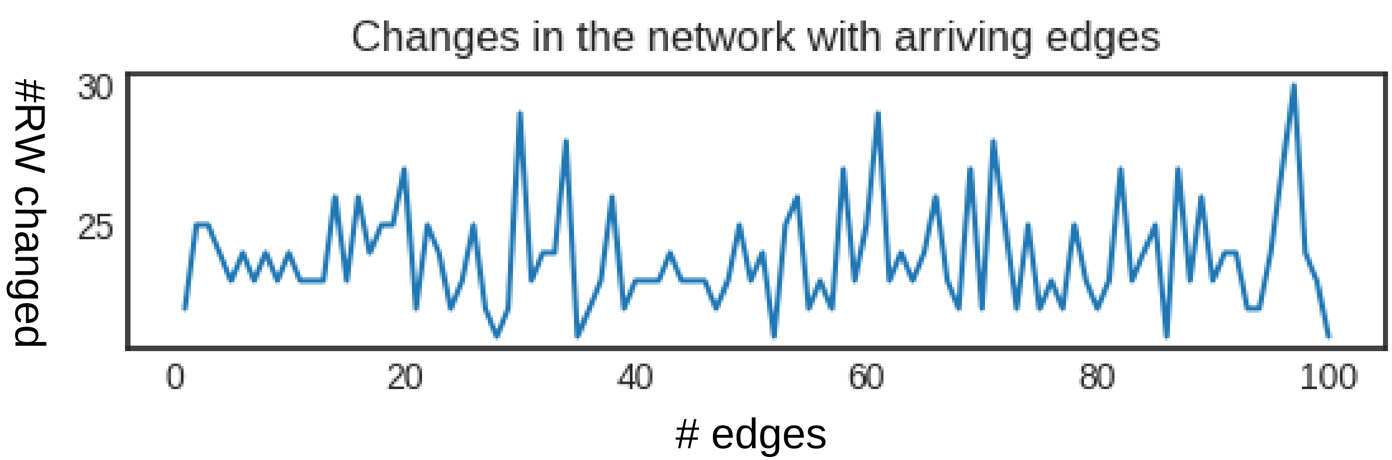


edge addition

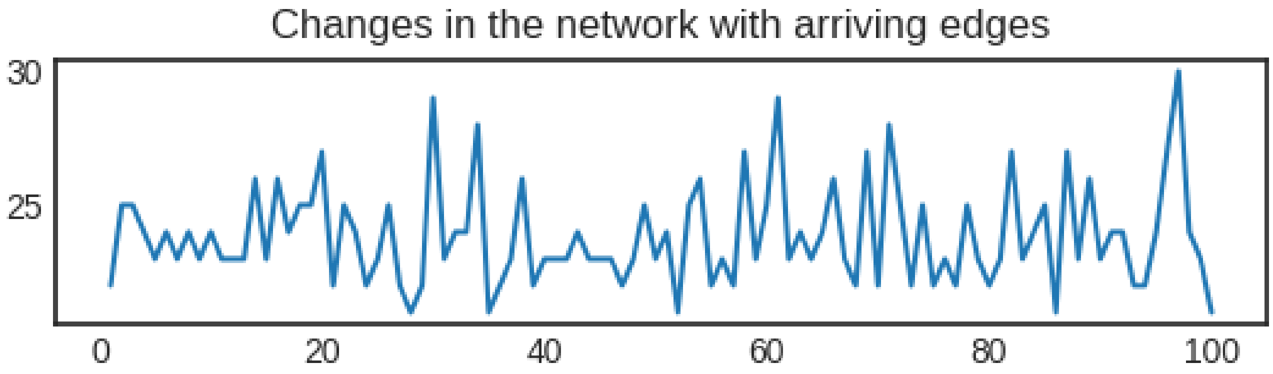
1	3	5	8	7	6	4	5
2	1	3	5	8	7	6	5
.	.						
.	.						
.	.						
.	.						
87	8	5	4	3	5	6	7
88	4	5	6	7	8	7	
89	2	1	3	5	6	7	8
90	7	4	2	1	3	5	6

count #RW changed

repeat for upcoming edges



Peak Detection



When to re-embed?

1	3 5 8 7 6 4 5
2	1 3 5 8 7 6 5
.	.
.	.
.	.
.	.
87	8 5 4 3 5 6 7
88	4 5 6 7 8 9
89	2 1 3 5 6 7 8
90	7 4 2 1 3 5 6

Informed decision based on edge importance (monitor #rw updated so far)

stream of edges

Periodic

(n_1, n_2)
(n_3, n_4)
(n_5, n_6)
(n_7, n_8)
Embed
. . .
Embed
(n_k, n_{k+1})
(n_{k+2}, n_{k+3})
(n_{k+4}, n_{k+5})
(n_{k+5}, n_{k+6})

Adaptive

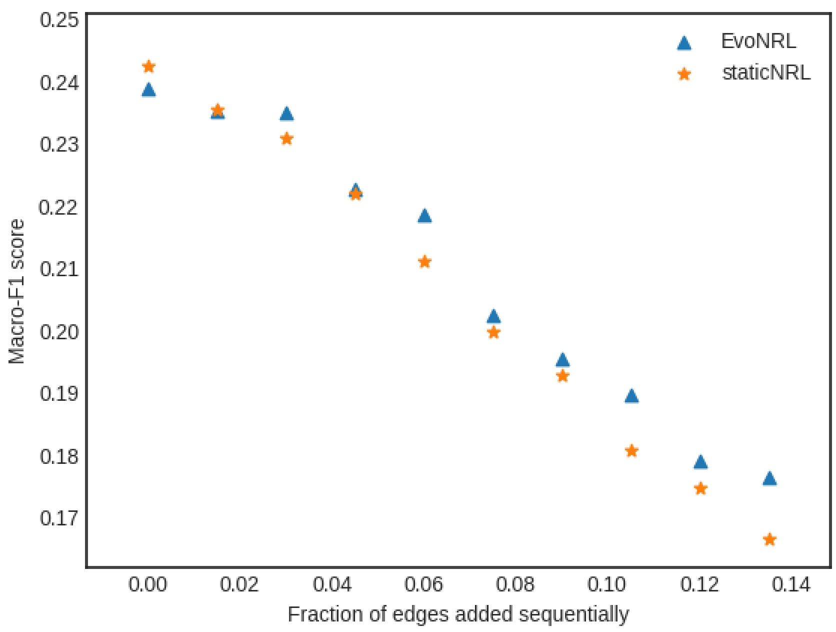
(n_1, n_2)
Embed
(n_3, n_4)
(n_5, n_6)
(n_7, n_8)
. . .
(n_k, n_{k+1})
Embed
(n_{k+2}, n_{k+3})
(n_{k+4}, n_{k+5})
(n_{k+5}, n_{k+6})

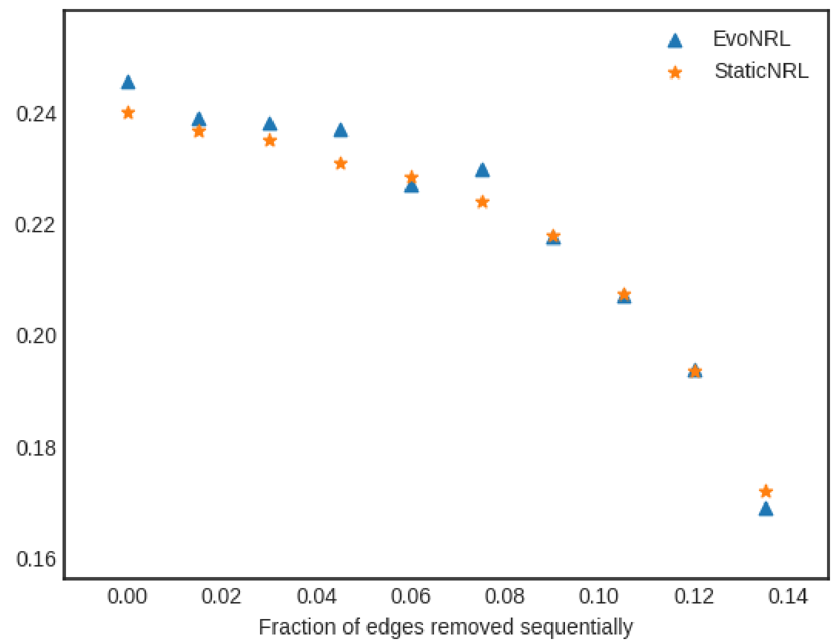
Evaluation: EvoNRL vs StaticNRL

Running Time

- EvoNRL **<<** StaticNRL

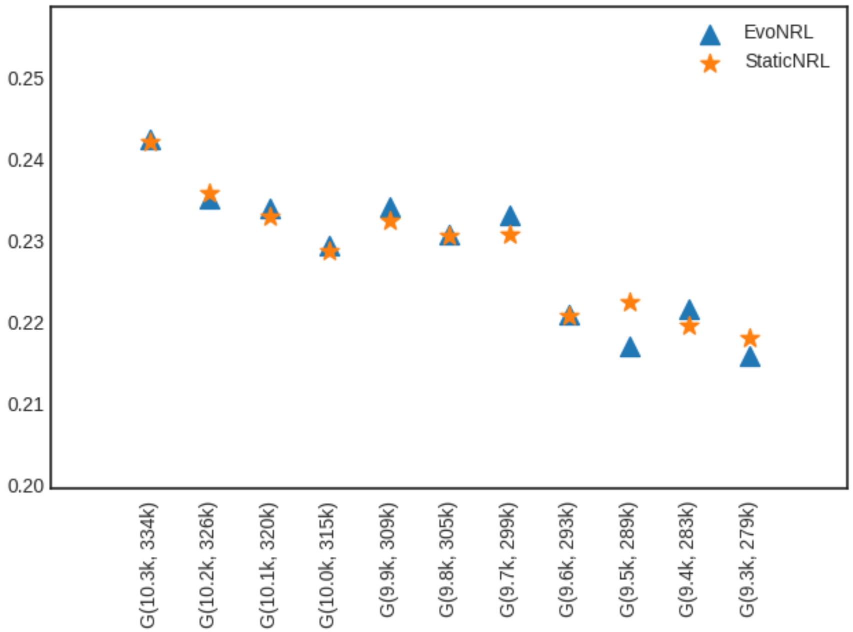
Accuracy: edge addition & edge deletion

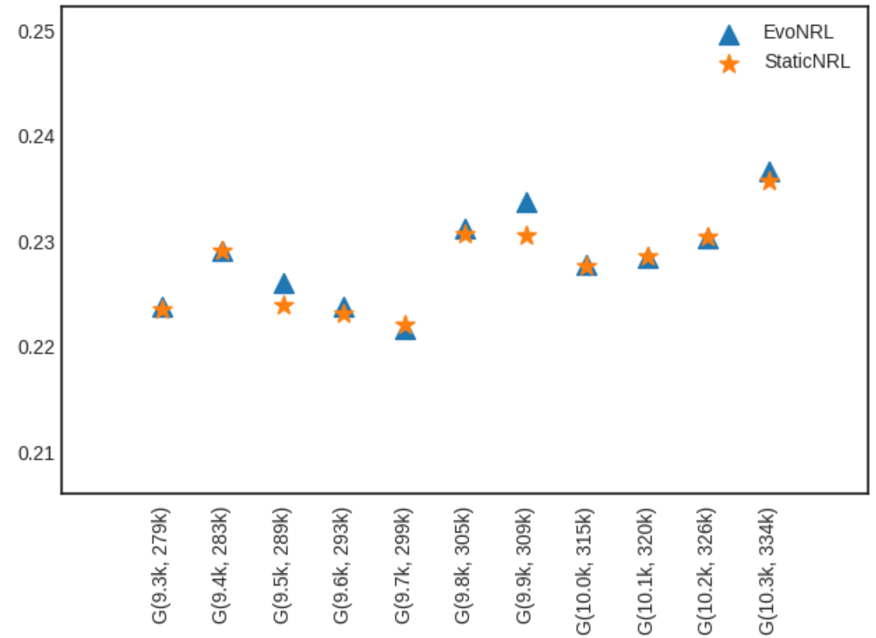




EvoNRL has the similar accuracy as StaticNRL

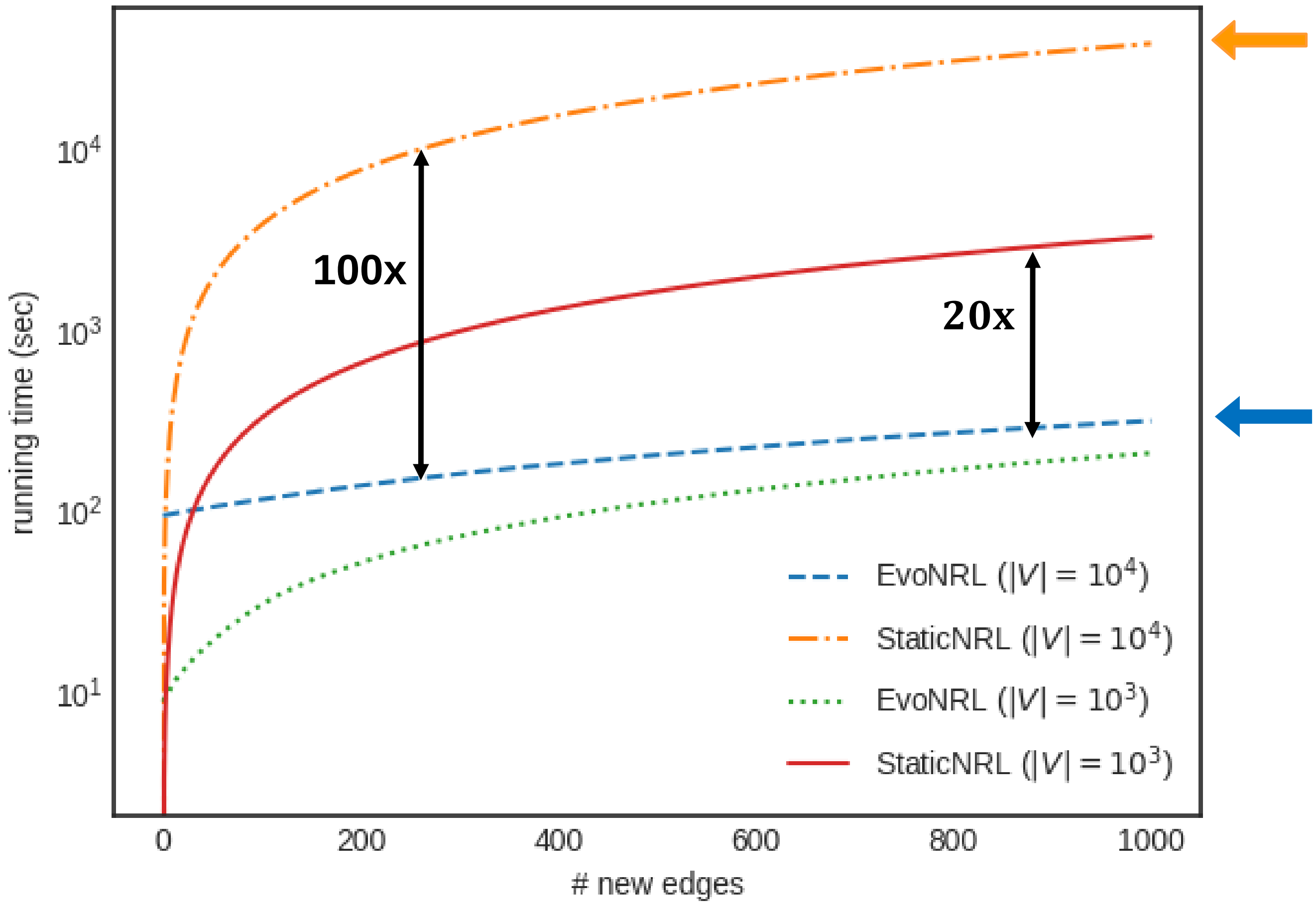
Accuracy: node addition & node deletion





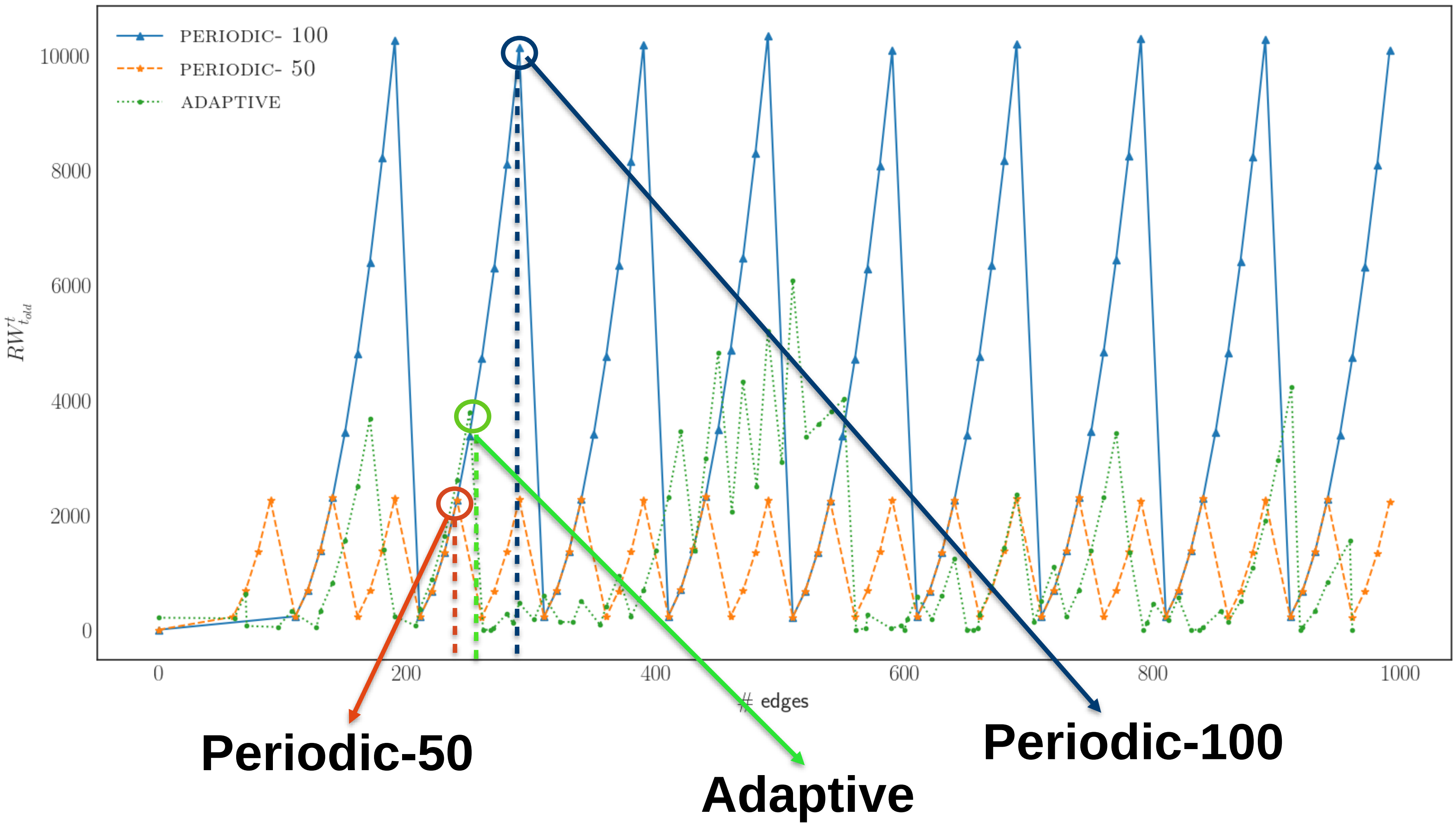
EvoNRL has the similar accuracy as StaticNRL

Time Performance

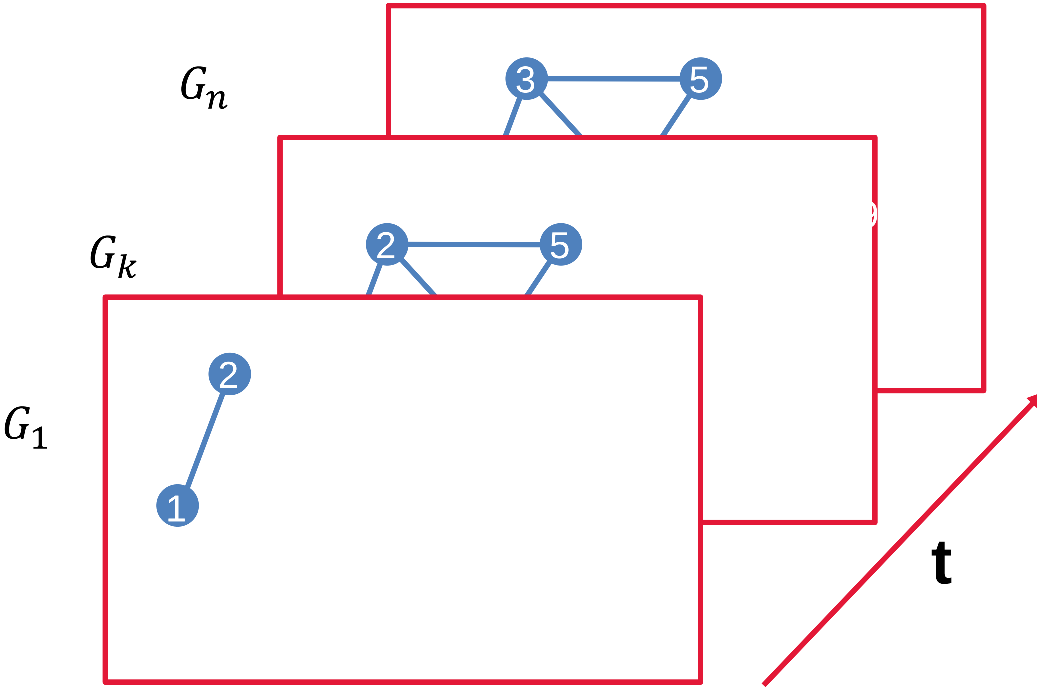


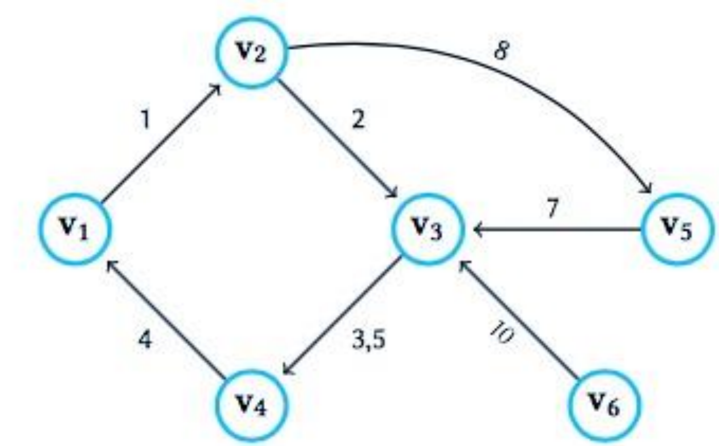
EvoNRL performs orders of time faster than StaticNRL

Decision-making Performance



Related Work





Uses outdated temporal information

Nguyen, Giang Hoang, et al. "Continuous-time dynamic network embeddings." *Companion Proceedings of the The Web C*

Summary

how can we learn representations of an evolving network?

EvoNRL

time efficient

accurate

generic method

Thank you!

Questions?

References

[Complex Networks 2018] **EvoNRL: Evolving Network Representation Learning Based on Random Walks**. Farzaneh Heidari, Manos Papagelis. Proceedings of the 7th International Conference on Complex Networks and Their Applications.

[KDD 2009] **Relational learning via latent social dimensions**. Tang Lei, Huan Liu. Proceedings of the 20th ACM SIGKDD international conference on Knowledge Discovery and Data Mining.

[ACM SIGKDD 2014] **Deepwalk: Online Learning of Social Representations**. Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. Proceedings of the 20th ACM SIGKDD international conference on Knowledge Discovery and Data Mining.

[ACM SIGKDD 2016] **node2vec: Scalable Feature Learning for Networks**. Aditya Grover and Jure Leskovec. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[ACM SIGKDD 2015] **PTE: Predictive Text Embedding Through Large-scale Heterogeneous Text Networks**. Jian Tang, Meng Qu, and Qiaozhu Mei. Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[WWW 2015] **LINE: Large-scale Information Network Embedding**. Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Proceedings of the 24th International World Wide Web Conference.

[ACL 2016] **Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change**. William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Proceedings of the Annual meeting of the Association for Computational Linguistics.

[WWW 2018] **Continuous-time dynamic network embeddings**. Giang Hoang Nguyen, John Boaz Lee, Ryan A. Rossi, Nesreen K. Ahmed, Eunyee Koh, and Sungchul Kim. Proceedings of the International World Wide Web Conferences Steering Committee, 2018.