UNIVERSITÄT MANNHEIM

New Adventures in RDF2vec

also includes the latest adventures

Heiko Paulheim
University of Mannheim

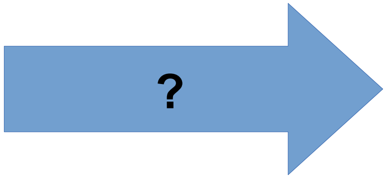
Alert:
may contains spoilers on future publications

Graphs vs. Vectors

- Data Science tools for prediction etc.
  - Python, Weka, R, RapidMiner, …
  - Algorithms that work on vectors, not graphs
- Bridges built over the past years:
  - FeGeLOD (Weka, 2012), RapidMiner LOD Extension (2015), Python KG Extension (2021)

FeGeLOD
Feature Generation from Linked Open Data

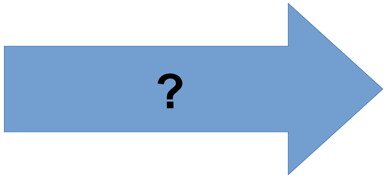
kgextension



Row No.	attribute_1	attribute_2	attribute_3	attribute_4	attribute_5	attribute_6	attribute_7	attribute_8	attribute_9	attribute_10	attribute_11
1	0.020	0.037	0.043	0.021	0.095	0.099	0.154	0.160	0.311	0.211	0.161
2	0.045	0.052	0.084	0.069	0.118	0.258	0.216	0.348	0.334	0.287	0.492
3	0.026	0.058	0.110	0.108	0.097	0.228	0.243	0.377	0.560	0.619	0.633
4	0.010	0.017	0.062	0.021	0.021	0.037	0.110	0.128	0.060	0.126	0.088
5	0.076	0.067	0.048	0.039	0.059	0.065	0.121	0.247	0.356	0.446	0.415
6	0.029	0.045	0.028	0.017	0.038	0.099	0.120	0.183	0.210	0.304	0.299
7	0.032	0.096	0.132	0.141	0.167	0.171	0.073	0.140	0.208	0.351	0.179
8	0.052	0.055	0.084	0.032	0.116	0.092	0.103	0.061	0.146	0.284	0.280
9	0.022	0.037	0.048	0.048	0.065	0.059	0.075	0.010	0.068	0.149	0.116
10	0.016	0.017	0.035	0.007	0.019	0.067	0.106	0.070	0.096	0.025	0.080
11	0.004	0.006	0.015	0.034	0.031	0.028	0.040	0.027	0.032	0.045	0.049
12	0.012	0.031	0.017	0.031	0.036	0.010	0.018	0.058	0.112	0.084	0.055
13	0.008	0.009	0.005	0.025	0.034	0.055	0.053	0.096	0.101	0.124	0.110
14	0.009	0.006	0.025	0.049	0.120	0.159	0.139	0.099	0.096	0.190	0.190
15	0.012	0.043	0.060	0.045	0.060	0.035	0.053	0.034	0.105	0.212	0.164
16	0.030	0.061	0.065	0.092	0.162	0.229	0.218	0.203	0.146	0.085	0.248
17	0.035	0.012	0.019	0.047	0.074	0.118	0.168	0.154	0.147	0.291	0.233
18	0.019	0.061	0.038	0.077	0.139	0.081	0.057	0.022	0.104	0.119	0.124
19	0.027	0.009	0.015	0.028	0.041	0.076	0.103	0.114	0.079	0.152	0.168
20	0.013	0.015	0.064	0.173	0.257	0.256	0.295	0.411	0.498	0.592	0.583
21	0.047	0.051	0.082	0.125	0.178	0.307	0.301	0.236	0.383	0.376	0.302
22	0.066	0.058	0.084	0.037	0.046	0.077	0.077	0.113	0.235	0.184	0.287
23	0.010	0.048	0.030	0.030	0.065	0.108	0.236	0.238	0.007	0.188	0.146
24	0.011	0.015	0.014	0.008	0.021	0.106	0.102	0.044	0.093	0.073	0.074

Graphs vs. Vectors

- Transformation strategies (aka *propositionalization*)
  - e.g., types: type_horror_movie=true
  - e.g., data values: year=2011
  - e.g., aggregates: nominations=7



Row No.	attribute_1	attribute_2	attribute_3	attribute_4	attribute_5	attribute_6	attribute_7	attribute_8	attribute_9	attribute_10	attribute_11
1	0.020	0.037	0.043	0.021	0.095	0.099	0.154	0.160	0.311	0.211	0.161
2	0.045	0.052	0.084	0.069	0.118	0.258	0.216	0.348	0.334	0.287	0.492
3	0.026	0.058	0.110	0.108	0.097	0.228	0.243	0.377	0.560	0.619	0.633
4	0.010	0.017	0.062	0.021	0.021	0.037	0.110	0.128	0.060	0.126	0.088
5	0.076	0.067	0.048	0.039	0.059	0.065	0.121	0.247	0.356	0.446	0.415
6	0.029	0.045	0.028	0.017	0.038	0.099	0.120	0.183	0.210	0.304	0.299
7	0.032	0.096	0.132	0.141	0.167	0.171	0.073	0.140	0.208	0.351	0.179
8	0.052	0.055	0.084	0.032	0.116	0.092	0.103	0.061	0.146	0.284	0.280
9	0.022	0.037	0.048	0.048	0.065	0.059	0.075	0.010	0.068	0.149	0.116
10	0.016	0.017	0.035	0.007	0.019	0.067	0.106	0.070	0.096	0.025	0.080
11	0.004	0.006	0.015	0.034	0.031	0.028	0.040	0.027	0.032	0.045	0.049
12	0.012	0.031	0.017	0.031	0.036	0.010	0.018	0.058	0.112	0.084	0.055
13	0.008	0.009	0.005	0.025	0.034	0.055	0.053	0.096	0.101	0.124	0.110
14	0.009	0.006	0.025	0.049	0.120	0.159	0.139	0.099	0.096	0.190	0.190
15	0.012	0.043	0.060	0.045	0.060	0.035	0.053	0.034	0.105	0.212	0.164
16	0.030	0.061	0.065	0.092	0.162	0.229	0.218	0.203	0.146	0.085	0.248
17	0.035	0.012	0.019	0.047	0.074	0.118	0.168	0.154	0.147	0.291	0.233
18	0.019	0.061	0.038	0.077	0.139	0.081	0.057	0.022	0.104	0.119	0.124
19	0.027	0.009	0.015	0.028	0.041	0.076	0.103	0.114	0.079	0.152	0.168
20	0.013	0.015	0.064	0.173	0.257	0.256	0.295	0.411	0.498	0.592	0.583
21	0.047	0.051	0.082	0.125	0.178	0.307	0.301	0.236	0.383	0.376	0.302
22	0.066	0.058	0.084	0.037	0.046	0.077	0.077	0.113	0.235	0.184	0.287
23	0.010	0.048	0.030	0.030	0.065	0.108	0.236	0.238	0.007	0.188	0.146
24	0.011	0.015	0.014	0.008	0.021	0.106	0.102	0.044	0.093	0.073	0.074

Graphs vs. Vectors

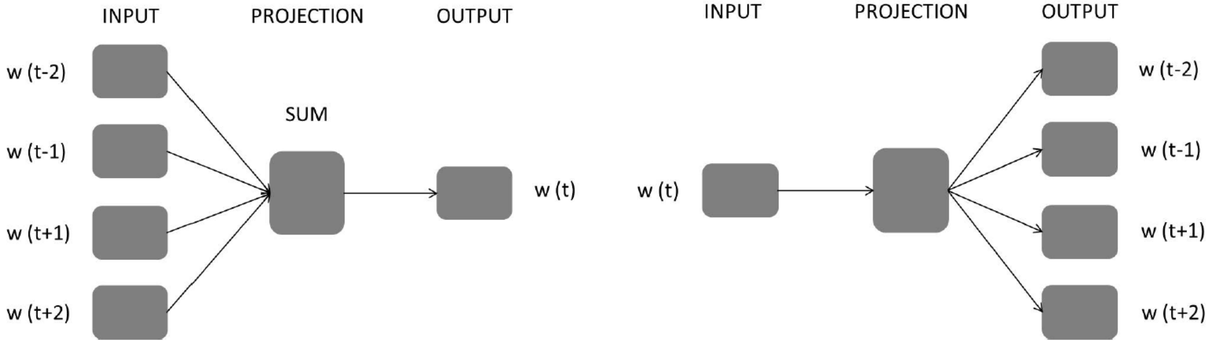
- Observations with simple propositionalization strategies
  - Even simple features (e.g., add all numbers and types) can help on many problems
  - More sophisticated features often bring additional improvements
    - Combinations of relations and individuals
      - e.g., movies directed by Steven Spielberg
    - Combinations of relations and types
      - e.g., movies directed by Oscar-winning directors
    - …
  - But
    - The search space is enormous!
    - *Generate first, filter later* does not scale well

FeGeLOD
Feature Generation from Linked Open Data

kgextension

Towards RDF2vec

- Excursion: word embeddings
  - word2vec proposed by Mikolov et al. (2013)
  - predict a word from its context or vice versa
    - Idea: similar words appear in similar contexts, like
      - Jobs, Wozniak, and Wayne founded **Apple** Computer Company in April 1976
      - **Google** was officially founded as a company in January 2006
  - usually trained on large text corpora
    - projection layer: embedding vectors



From Word Embeddings to Graph Embeddings

- Basic idea:
  - extract random walks from an RDF graph:

    Mulholland Dr. $\xrightarrow{\text{director}}$ David Lynch $\xrightarrow{\text{nationality}}$ US
  - feed walks into word2vec algorithm
- Order of magnitude (e.g., DBpedia)
  - ~6M entities ("words")
  - start up to 500 random walks per entity, length up to 8

    → corpus of >20B tokens
- Result:
  - entity embeddings
  - most often outperform other propositionalization techniques
  - fixed number of features

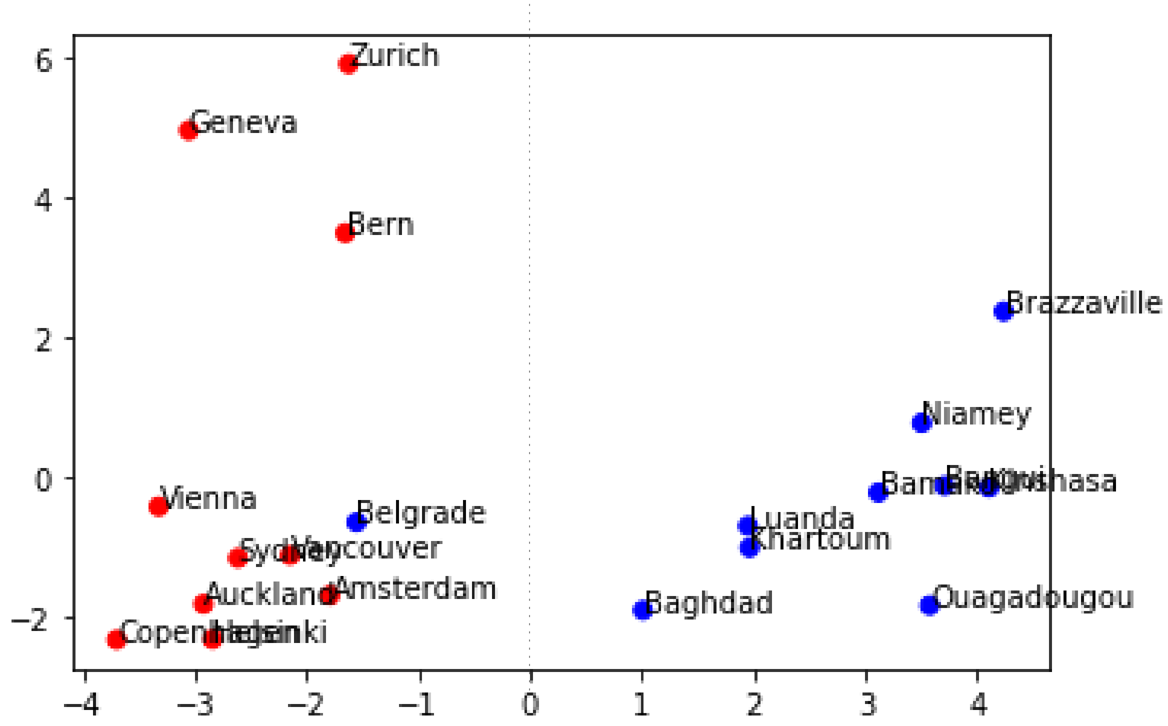
Ristoski and Paulheim (2016): RDF2vec: RDF graph embeddings for data mining

The End of Petar's PhD Journey…

- ...and the beginning of the RDF2vec adventure

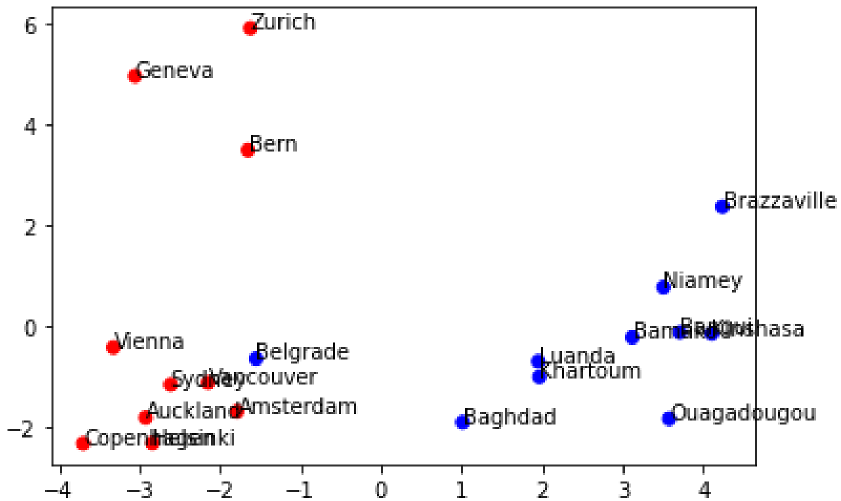
Why does RDF2vec Work?

- Example: PCA plot of an excerpt of a cities classification problem
  - From cities classification task in the embedding evaluation framework by Pellegrino et al.



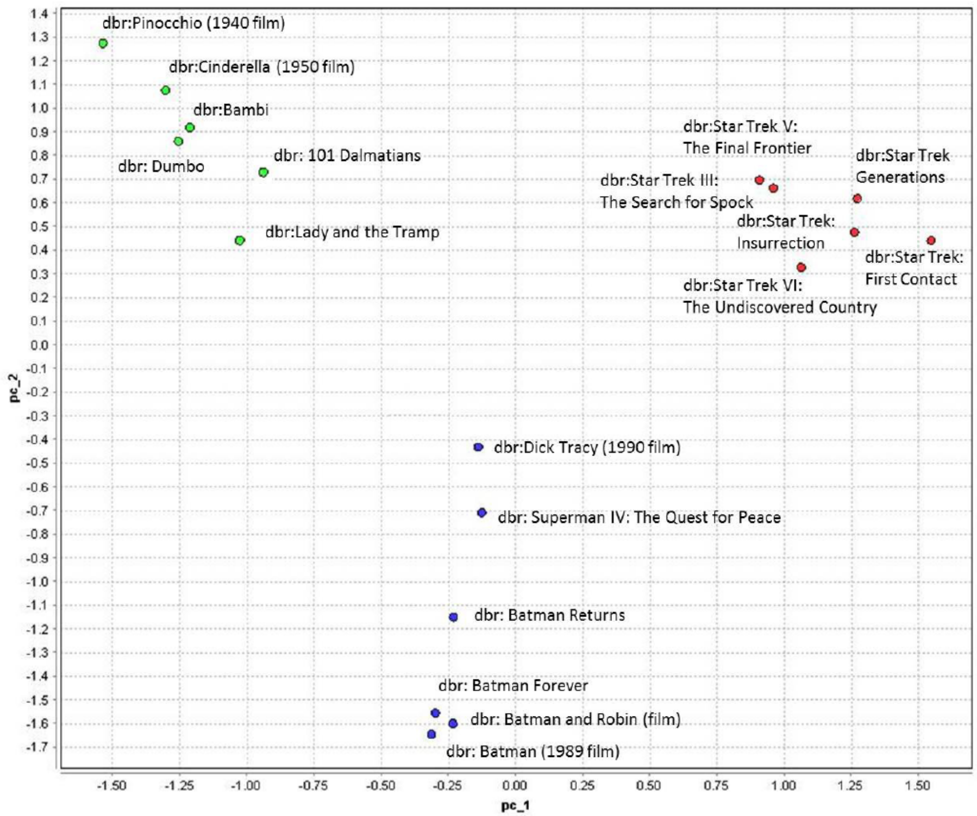
Why does RDF2vec Work?

- In downstream machine learning, we usually want *class separation*
  – to make the life of the classifier as easy as possible
- Class separation means
  – Similar entities (i.e., same class) are projected *closely* to each other
  – Dissimilar entities (i.e., different classes) are projected *far away* from each other



Why does RDF2vec Work?

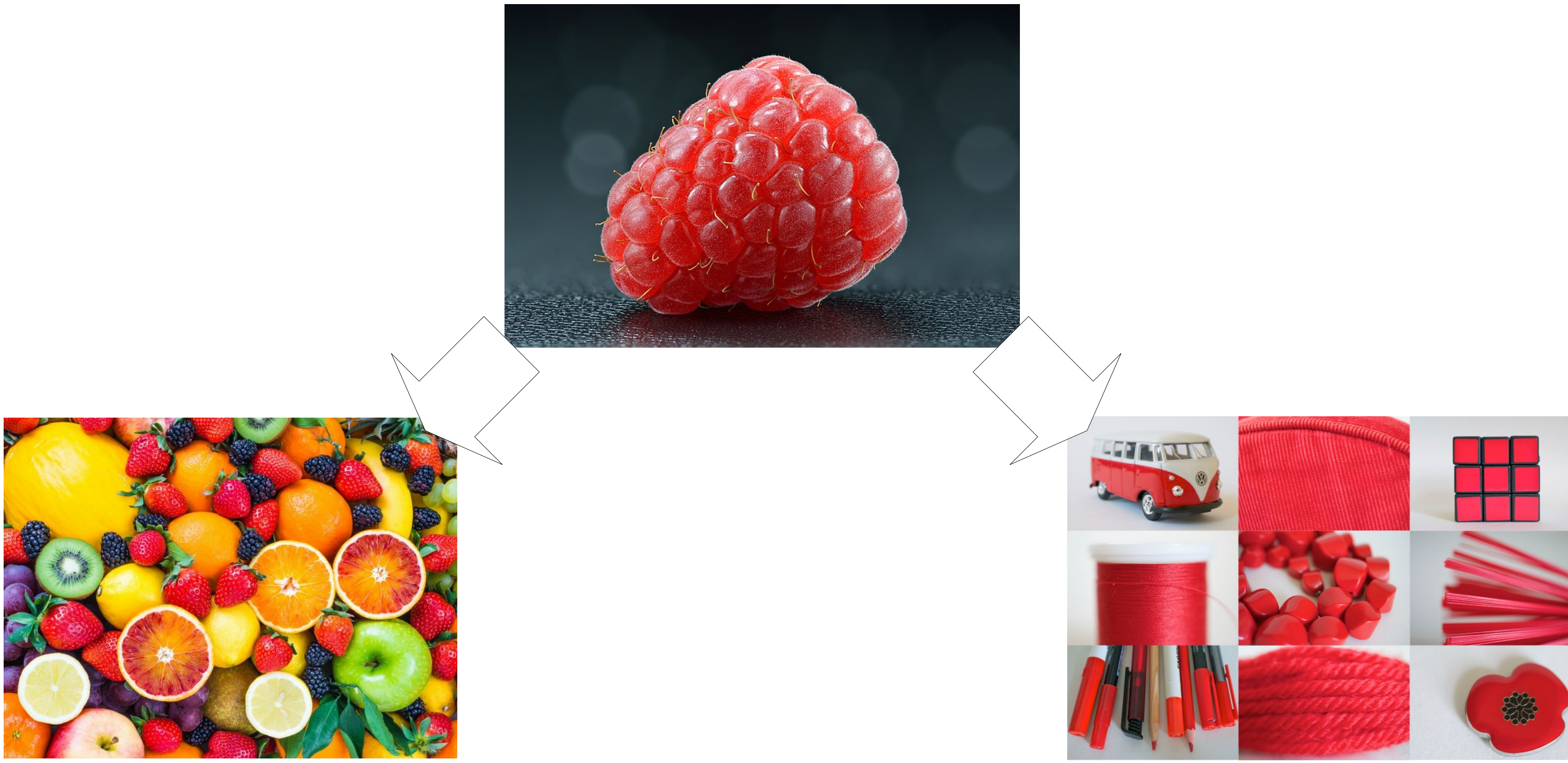
- Observation: close projection of similar entities
  – Usage example: content-based recommender system based on k-NN



Ristoski and Paulheim (2016): RDF2vec: RDF graph embeddings for data mining

Close Projection of Similar Entities

- What does *similar* mean?



Similarity vs. Relatedness

- Closest 10 entities to *Angela Merkel* in different vector spaces

RDF2vec	TransE-L1	TransE-L2	TransR
Joachim Gauck	Gerhard Schröder	Gerhard Schröder	Sigmar Gabriel
Norbert Lammert	James Buchanan	Helmut Kohl	Frank-Walter Steinmeier
Stanislaw Tillich	Neil Kinnock	Konrad Adenauer	Philipp Rösler
Andreas Voßkuhle	Nicolas Sarkozy	Helmut Schmidt	Gerhard Schröder
Berlin	Joachim Gauck	Werner Faymann	Joachim Gauck
German language	Jacques Chirac	Alfred Gusenbauer	Christian Wulff
Germany	Jürgen Trittin	Kurt Georg Kiesinger	Guido Westerwelle
federalState	Sigmar Gabriel	Philipp Scheidemann	Helmut Kohl
Social Democratic Party	Guido Westerwelle	Ludwig Erhard	Jürgen Trittin
deputy	Christian Wulff	Wilhelm Marx	Jens Böhrnsen
RotatE	**DistMult**	**RESCAL**	**ComplEx**
Pontine raphe nucleus	Gerhard Schröder	Gerhard Schröder	Gerhard Schröder
Jonathan W. Bailey	Milan Truban	Kurt Georg Kiesinger	Diána Mészáros
Zokwang Trading	Maud Cuney Hare	Helmut Kohl	Francis M. Bator
Steven Hill	Tristan Matthiae	Annemarie Huber-Hotz	William B. Bridges
Chad Kreuter	Gerda Hasselfeldt	Wang Zhaoguo	Mette Vestergaard
Fred Hibbard	Faustino Sainz Muñoz	Franz Vranitzky	Ivan Rosenqvist
Mallory Ervin	Joachim Gauck	Bogdan Klich	Edward Clouston
Paulinho Kobayashi	Carsten Linnemann	İrsen Küçük	Antonio Capuzzi
Fullmetal Alchemist and the Broken Angel	Norbert Blüm	Helmut Schmidt	Steven J. McAuliffe
Archbishop Dorotheus of Athens	Neil Hood	Mao Zedong	Jenkin Coles

Portisch et al. (2022): Knowledge Graph Embedding for Data Mining vs. Knowledge Graph Embedding for Link Prediction - Two Sides of the Same Coin?

Back to Class Separation

- What is a class?
  - e.g., cities per se
  - e.g., cities in France
  - e.g., cities in France above 250k inhabitants
- Or something different, such as
  - e.g., everything located in Strasbourg
  - e.g., everything Strasbourg is known for

Back to Class Separation

- Observation: there are different kinds of classes:
  - Classes of objects of the same category (e.g., cities)

    → those are *similar*
  - Classes of objects of different categories (e.g., buildings, dishes, organizations, persons)

    → those are *related*

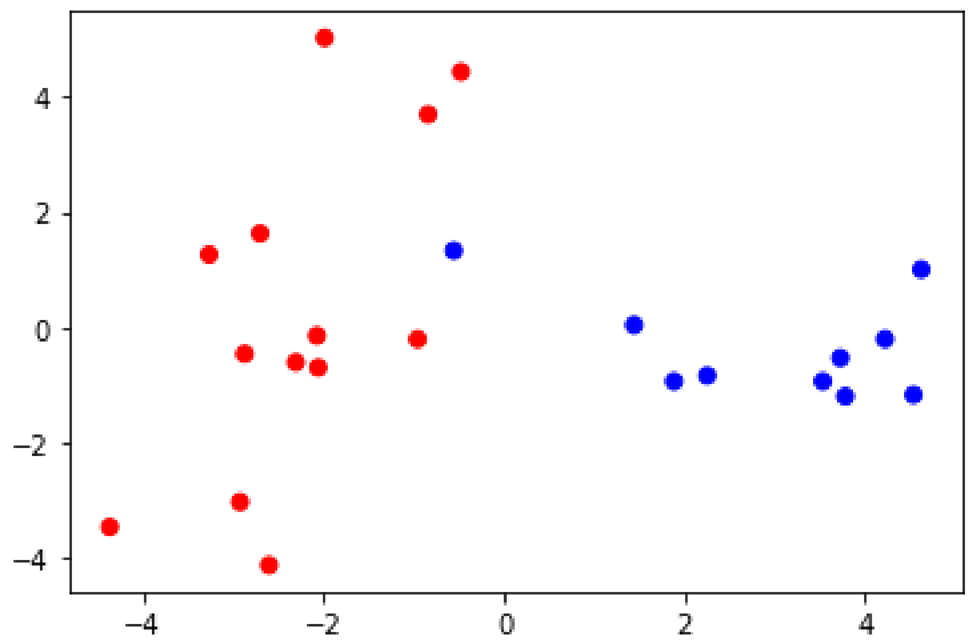
Intermediate Observation

- In most vector spaces of link prediction embeddings (TransE etc.): proximity ~ similarity
- In RDF2vec embedding space:
  proximity ~ a mix of similarity and relatedness

RDF2vec	TransE-L1	TransE-L2	TransR
Joachim Gauck	Gerhard Schröder	Gerhard Schröder	Sigmar Gabriel
Norbert Lammert	James Buchanan	Helmut Kohl	Frank-Walter Steinmeier
Stanislaw Tillich	Neil Kinnock	Konrad Adenauer	Philipp Rösler
Andreas Voßkuhle	Nicolas Sarkozy	Helmut Schmidt	Gerhard Schröder
Berlin	Joachim Gauck	Werner Faymann	Joachim Gauck
German language	Jacques Chirac	Alfred Gusenbauer	Christian Wulff
Germany	Jürgen Trittin	Kurt Georg Kiesinger	Guido Westerwelle
federalState	Sigmar Gabriel	Philipp Scheidemann	Helmut Kohl
Social Democratic Party	Guido Westerwelle	Ludwig Erhard	Jürgen Trittin
deputy	Christian Wulff	Wilhelm Marx	Jens Böhrnsen
RotatE	**DistMult**	**RESCAL**	**ComplEx**
Pontine raphe nucleus	Gerhard Schröder	Gerhard Schröder	Gerhard Schröder
Jonathan W. Bailey	Milan Truban	Kurt Georg Kiesinger	Diána Mészáros
Zokwang Trading	Maud Cuney Hare	Helmut Kohl	Francis M. Bator
Steven Hill	Tristan Matthiae	Annemarie Huber-Hotz	William B. Bridges
Chad Kreuter	Gerda Hasselfeldt	Wang Zhaoguo	Mette Vestergaard
Fred Hibbard	Faustino Sainz Muñoz	Franz Vranitzky	Ivan Rosenqvist
Mallory Ervin	Joachim Gauck	Bogdan Klich	Edward Clouston
Paulinho Kobayashi	Carsten Linnemann	İrsen Küçük	Antonio Capuzzi
Fullmetal Alchemist and the Broken Angel	Norbert Blüm	Helmut Schmidt	Steven J. McAuliffe
Archbishop Dorotheus of Athens	Neil Hood	Mao Zedong	Jenkin Coles

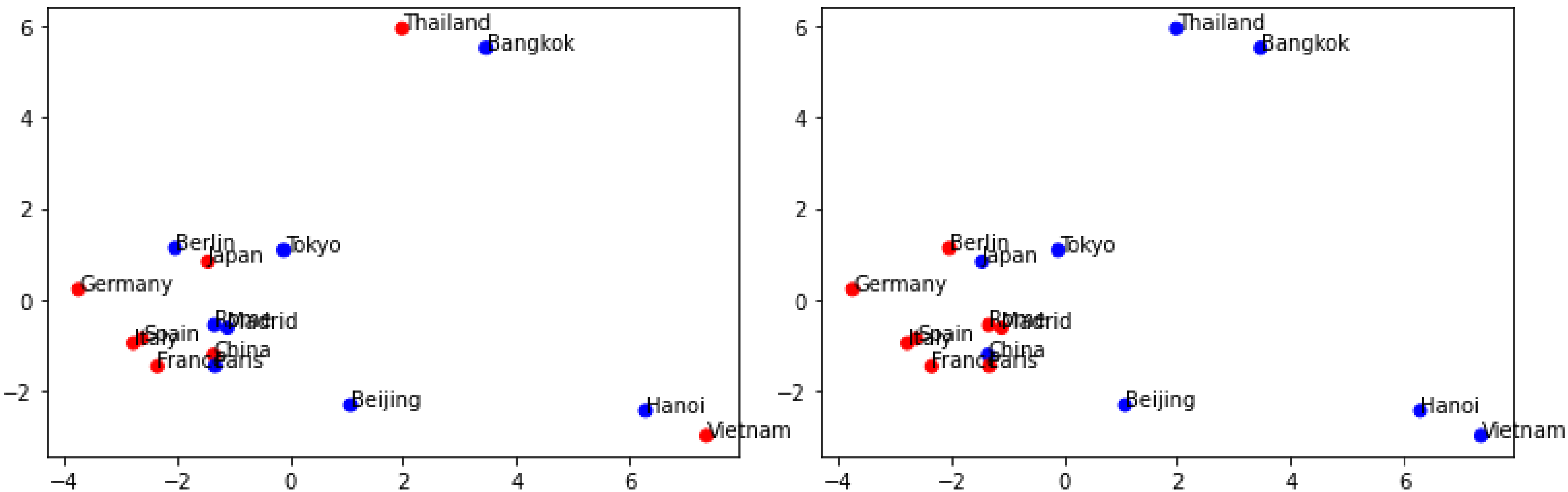
So… why does RDF2vec Work Then?

- Recap: downstream ML algorithms need class separation
  - but RDF2vec groups items by similarity *and relatedness*
- Why is RDF2vec still so good at classification?



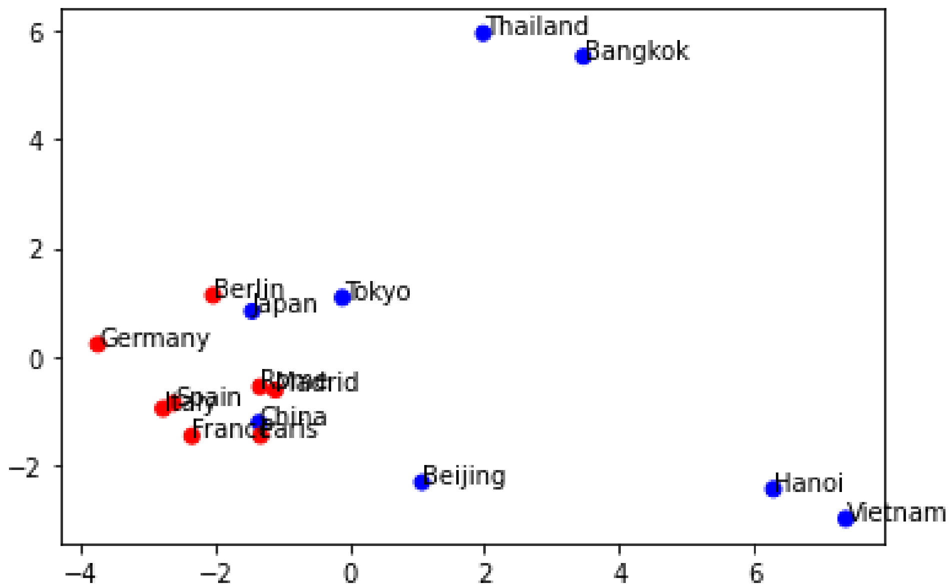
Example

- It depends on the classification problem at hand!
  - Cities vs. countries
  - Places in Europe vs. places in Asia



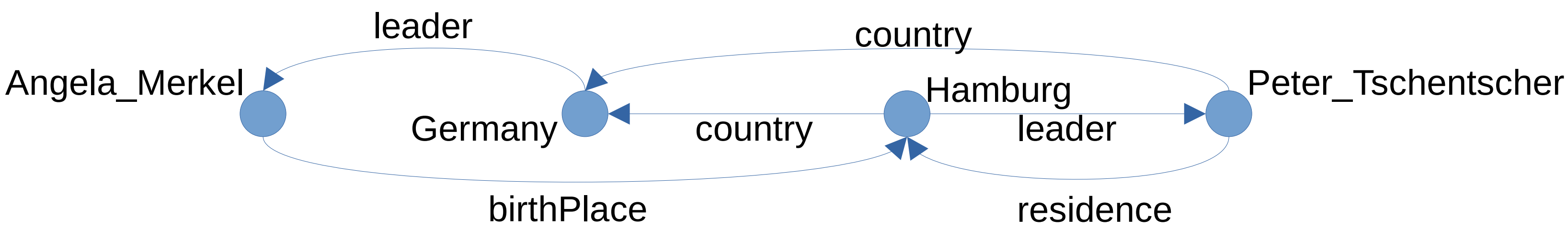
So… why does RDF2vec Work Then?

- Many downstream classification tasks are *homogeneous*
  - e.g., classifying cities in different subclasses
- For homogeneous entities:
  - relatedness provides finer-grained distinctions



Similarity vs. Relatedness

- Recap word embeddings:
  - Jobs, Wozniak, and Wayne founded **Apple** Computer Company in April 1976
  - **Google** was officially founded as a company in January 2006
- Graph walks:
  - Hamburg → country → **Germany** → leader → Angela_Merkel
  - Germany → leader → **Angela_Merkel** → birthPlace → Hamburg
  - Hamburg → leader → **Peter_Tschentscher** → residence → Hamburg



Order-Aware RDF2vec

- Using an order-aware variant of word2vec
- Experimental results:
  - order-aware RDF2vec most often outperforms classic RDF2vec
  - a bit more computation heavy, but still scales to DBpedia etc.

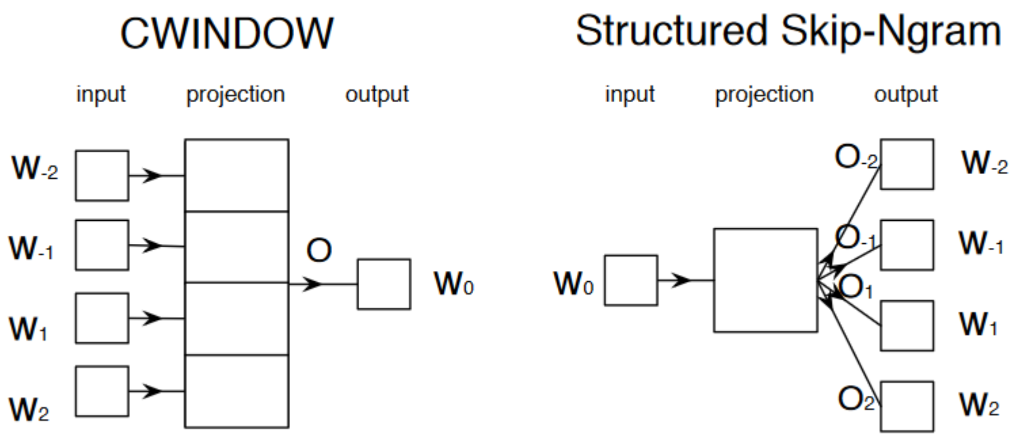


Figure 2: Illustration of the Structured Skip-gram and Continuous Window (CWindow) models.

Ling et al. (2015): Two/Too Simple Adaptations of Word2Vec for Syntax Problems.

Similarity vs. Relatedness

- Exploiting different notions of proximity
  - Use case: table interpretation (a special case of entity disambiguation)

related

similar

Country Name	Capital	Currency	Official Language	Head of Government
Afghanistan	Kabul	Afghani	Dari Persian; Pashto	… Ashraf Ghani
Albania	irane	Lek	Albanian	… Minister – Edi Rama
Algeria	lgiers	Dinar	Arabic; Tamazight; French	Prime Minister – Abdelaziz Djerad
Andorra	ndorra la Vella	Euro	Catalan	Prime Minister - Xavier Espot Zamora
Angola	uanda	New Kwanza	Portuguese	President – João Lourenço
Antigua and Barbuda	aint John's	East Caribbean dollar	English	Prime Minister – Gaston Browne
Argentina	uenos Aires	Peso	Spanish	President – Alberto Fernández
Armenia	revan	Dram	Armenian	President – Armen Sarksyan
Australia	Canberra	Australian dollar	English	Prime Minister – Scott Morrison

Similarity vs. Relatedness in Graph Walks

- Which parts of a walk denote what?
  - *Hamburg → country → **Germany** → leader → Angela_Merkel*
  - *Germany → leader → **Angela_Merkel** → birthPlace → Hamburg*
  - *Hamburg → leader → **Peter_Tschentscher** → residence → Hamburg*
  - *California → leader → **Gavin_Newsom** → birthPlace → San_Francisco*

- Common predicates (leader, birthPlace)
  - Similar entities
- Common entities (Hamburg)
  - Related entities
  - For same-class entities: *similar* entities!

Portisch and Paulheim (ESWC 2022): Walk this Way! Entity Walks and Property Walks for RDF2vec.

Similarity vs. Relatedness in Graph Walks

- Given that observation:
  - Common predicates (leader, birthPlace)
    - Similar classes
  - Common entities (Hamburg)
    - Related entities
    - For same-class entities: *similar* entities!
- ...we should be able to learn tailored embeddings
  - using walks of predicates → embedding space encodes similarity
  - using walks of entities → embedding space encodes relatedness

Portisch and Paulheim (ESWC 2022): Walk this Way! Entity Walks and Property Walks for RDF2vec.

Similarity vs. Relatedness in Graph Walks

- Classic RDF2vec walks:
  - *Germany* → *leader* → ***Angela_Merkel*** → *birthPlace* → *Hamburg*
- p-walk (predicates only except for focus entity)
  - *country* → *leader* → ***Angela_Merkel*** → *birthPlace* → *mayor*
- e-walk (entities only)
  - *Berlin* → *Germany* → ***Angela_Merkel*** → *Hamburg* → *Elbphilharmonie*

Portisch and Paulheim (ESWC 2022): Walk this Way! Entity Walks and Property Walks for RDF2vec.

The RDF2vec Zoo

- We now have an entire zoo of RDF2vec variants
  - SG vs. CBOW
  - Order-aware vs. unordered (“classic”)
  - Classic walks vs. e-walks vs. p-walks

The RDF2vec Zoo – Preliminary Evaluation

Table 1. Result of the 12 RDF2vec variants on 20 tasks. The best score for each task is printed in bold. The first four columns use classic RDF2vec walks, while variants using e-walks and p-walks are marked with e and p, respectively. The suffix $_{oa}$ marks the ordered variant of RDF2vec.

Task	Metric	Dataset	sg	sg_{oa}	cbow	$cbow_{oa}$	e sg	e sg_{oa}	e cbow	e $cbow_{oa}$	p sg	p sg_{oa}	p cbow	p $cbow_{oa}$
[illegible]	[illegible]	[illegible]	0.706	0.713	[illegible]	0.690	0.696	**0.717**	0.703	0.690	0.564	0.623	0.551	0.612
		[illegible]	**0.818**	0.803	0.725	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.677	0.501	0.707
		[illegible]	**0.623**	0.605	0.575	0.600	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.610	0.560	0.578
		Metacritic Albums	0.586	0.585	0.536	0.532	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.632	0.569	**0.667**
		Metacritic Movies	0.726	0.716	0.549	0.626	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.660	0.535	0.663
Clustering	ACC	Cities and Countries (2k)	0.789	0.900	0.520	**0.917**	0.726	0.726	0.668	0.660	0.605	0.520	0.637	0.733
		Cities and Countries	0.587	0.760	0.783	0.720	0.749	0.766	**0.820**	0.745	0.687	0.782	0.787	0.728
		Cities, Albums Movies, AAUP, Forbes	0.829	**0.854**	0.547	0.652	0.759	0.828	0.557	0.719	0.598	0.798	0.663	0.748
		Teams	0.909	0.931	0.940	0.925	0.889	0.926	0.916	0.931	**0.941**	0.938	0.940	0.580
Regression	RMSE	AAUP	65.985	**63.814**	77.250	66.473	67.337	65.429	70.482	69.292	80.318	72.610	96.248	77.895
		Cities	15.375	**12.782**	18.963	19.287	17.017	16.913	17.290	20.798	20.322	17.214	24.743	20.334
		Forbes	36.545	**36.050**	39.204	37.067	38.589	38.558	39.867	36.313	37.146	36.374	37.947	38.952
		Metacritic Albums	15.288	15.903	15.812	15.705	15.573	15.785	15.574	**14.640**	15.178	14.869	15.000	16.679
		Metacritic Movies	**20.215**	20.420	24.238	23.362	20.436	20.258	23.348	22.518	23.235	22.402	23.979	22.071
Semantic Analogies	ACC	capital country entities	**0.957**	0.864	0.810	0.789	0.794	0.747	0.660	0.397	0.008	0.091	0.000	0.036
		all capital country entities	**0.905**	0.857	0.594	0.758	0.657	0.591	0.359	0.592	0.014	0.073	0.002	0.052
		currency entities	**0.574**	0.535	0.338	0.447	0.309	0.193	0.198	0.297	0.006	0.076	0.002	0.085
		city state entities	**0.609**	0.578	0.507	0.442	0.459	0.484	0.250	0.361	0.009	0.048	0.000	0.036
Entity Relatedness	Kendall Tau		0.747	0.716	0.611	0.547	**0.832**	0.800	0.726	0.779	0.432	0.768	0.568	0.737
Document Similarity	Harmonic Mean		0.237	0.230	0.283	0.209	0.275	0.250	0.170	0.111	0.193	**0.382**	0.296	0.256

Classic is usually quite good

oa variants are often superior

Portisch & Paulheim(2023): The RDF2vec Family of Knowledge Graph Embedding Methods

Which Classes can be Learned with RDF2vec?

- We already saw that there are different notions of *classes*
- Idea: compile a list of class definitions as a benchmark
  - Classes are expressed as DL formulae, e.g.
  - $\exists r.\top$, e.g. Class *person with children*
  - $\exists r.\{e\}$, e.g.: Class *person born in New York City*
  - $\exists R.\{e\}$, e.g., Class *person with any relation to New York City*
  - $\exists r.C$, e.g., Class *person playing in a basketball team*
  - *…*

Which Classes can be Learned with RDF2vec?

- Formulating hypotheses
  - e.g., $\exists$r.T, cannot be learned when using e walks
- Testing hypotheses
  - using queries against DBpedia
- ...

The DLCC DBpedia Gold Standard

- Six classes (person, book, city, movie, album, species)
- Twelve test cases
  - Sometimes also with “harder” negatives
- Three sizes per test case (50, 500, 5,000 examples)
  - Each is a balanced binary classification problem

- >200 hand written SPARQL queries
- Dataset and code available online

Hypotheses (Overview)

- Different patterns require different signals, e.g.,
  - Specific relations (visible to classic and p-walks)
  - Specific entities (visible to classic and e-walks)
  - Distinguishing subject and object (only possible for oa variants)
  - ...and mixes of those

	Test Case	DL Expression	$RDF2vec$	$RDF2vec_{oa}$	$p-RDF2vec$	$p-RDF2vec_{oa}$	$e-RDF2vec$	$e-RDF2vec_{oa}$
H1a	tc01	$r.\top$		✓		✓		
H1a'	tc02	$r^{-1}.\top$		✓		✓		
H1b	tc03	$\exists r.\top \sqcup \exists r^{-1}.\top$	✓	✓	✓	✓		
H2a	tc04	$\exists R.\{e\} \sqcup \exists R^{-1}.\{e\}$	✓	✓			✓	✓
H2b	tc05	$\exists R_1.(\exists R_2.\{e\}) \sqcup \exists R_1^{-1}.(\exists R_2^{-1}.\{e\})$	✓	✓			✓	✓
H3	tc06	$r.\{e\}$		✓				
H4a	tc07	$\exists r.T$		✓		(✓)		
H4b	tc08	$\exists r^{-1}.T$						
H5	tc09	$\geq 2r.\top$		(✓)		(✓)		
H5'	tc10	$\geq 2r^{-1}.\top$		(✓)		(✓)		
H6a	tc11	$\geq 2r.T$		(✓)				
H6b	tc12	$\geq 2r^{-1}.T$						

Portisch & Paulheim (2023): The RDF2vec Family of Knowledge Graph Embedding Methods

Which Classes can be Learned with RDF2vec?

- Formulating hypotheses
  - e.g., $\exists r.T$, cannot be learned when using e-walks
- Testing hypotheses
  - using queries against DBpedia
- Seeing surprises
  - e.g., models trained on e-walks can reach ~90% accuracy in that case

Experiments on DBpedia Gold Standard

- Most hypotheses could **not** be confirmed
- All problems are learnable with an accuracy >75%
  - i.e., significantly better than guessing
- Also LP embeddings such as TransE work surprisingly well

TC	SG	SG_{oa}	CBOW	$CBOW_{oa}$	p-SG	$p\text{-}SG_{oa}$	p-CBOW	$p\text{-}CBOW_{oa}$	e-SG	$e\text{-}SG_{oa}$	e-CBOW	$e\text{-}CBOW_{oa}$	TransE-L1	TransE-L2	TransR	DistMult	ComplEx	RESCAL	RotatE
tc01	0.915	0.937	0.778	0.870	0.907	0.933	0.780	0.924	0.845	0.860	0.840	0.840	0.842	0.947	0.858	0.874	0.862	**0.966**	0.768
tc01 hard	0.681	0.891	0.637	0.891	0.627	0.903	0.576	0.894	0.644	0.651	0.659	0.659	0.799	**0.916**	0.744	0.646	0.651	0.830	0.618
tc02	0.953	0.961	0.865	0.956	0.930	0.972	0.901	**0.974**	0.883	0.895	0.906	0.906	0.852	0.970	0.832	0.859	0.853	0.908	0.737
tc02 hard	0.637	0.780	0.618	0.774	0.628	0.828	0.583	0.838	0.623	0.628	0.607	0.607	0.780	**0.849**	0.693	0.622	0.608	0.729	0.649
tc03	0.949	**0.958**	0.846	0.905	0.913	0.956	0.800	0.938	0.883	0.900	0.886	0.886	0.821	0.933	0.856	0.894	0.874	0.943	0.780
tc04	0.960	0.968	0.705	0.872	0.877	0.908	0.659	0.873	0.965	0.969	0.915	0.915	0.934	0.986	0.973	0.984	**0.990**	0.990	0.862
tc04 hard	0.963	0.984	0.674	**0.992**	0.725	0.828	0.583	0.782	0.938	0.990	0.983	0.983	0.814	0.912	0.855	0.917	0.935	0.918	0.789
tc05	0.986	0.992	0.772	0.906	0.869	0.899	0.719	0.870	0.990	**0.995**	0.931	0.931	0.867	0.948	0.881	0.907	0.905	0.908	0.802
tc06	0.957	0.963	0.698	0.850	0.876	0.903	0.641	0.857	0.960	0.969	0.928	0.928	0.929	0.965	0.976	0.985	**0.998**	0.990	0.866
tc06 hard	0.863	0.936	0.604	0.908	0.708	0.770	0.559	0.745	0.699	0.708	0.650	0.650	0.823	0.779	**0.964**	0.882	0.933	0.964	0.819
tc07	0.938	0.955	0.742	0.785	0.895	0.924	0.728	0.863	0.946	0.946	0.859	0.859	0.930	**0.987**	0.978	0.929	0.966	0.945	0.847
tc08	0.961	0.966	0.891	0.896	0.911	**0.968**	0.841	0.951	0.904	0.914	0.925	0.925	0.898	0.964	0.870	0.856	0.888	0.875	0.831
tc09	0.902	0.901	0.773	0.858	0.819	0.858	0.728	0.832	0.874	0.884	0.840	0.840	0.884	**0.938**	0.879	0.877	0.883	0.929	0.780
tc09 hard	0.785	0.793	0.659	0.751	0.698	0.741	0.600	0.712	0.777	0.782	0.744	0.744	0.749	**0.848**	0.758	0.774	0.776	0.820	0.676
tc10	0.947	0.958	0.918	0.905	0.924	0.975	0.852	0.969	0.911	0.912	0.925	0.925	0.957	**0.984**	0.898	0.918	0.931	0.927	0.878
tc10 hard	0.740	0.737	0.716	0.711	0.610	0.679	0.569	0.652	0.715	0.718	0.729	0.729	**0.775**	0.774	0.656	0.743	0.739	0.713	0.665
tc11	0.932	0.897	0.865	0.780	0.884	**0.991**	0.808	0.954	0.928	0.972	0.921	0.921	0.917	0.960	0.930	0.889	0.946	0.954	0.838
tc11 hard	0.725	0.737	0.687	0.676	0.684	0.707	0.631	0.707	0.763	0.734	0.641	0.641	0.712	**0.806**	0.753	0.666	0.723	0.726	0.638
tc12	0.955	0.938	0.888	0.909	0.900	0.971	0.830	0.965	0.893	0.905	0.904	0.904	0.961	**0.984**	0.879	0.912	0.894	0.927	0.834
tc12 hard	0.714	0.717	0.712	0.699	0.628	0.637	0.545	0.628	0.690	0.713	0.715	0.715	0.762	**0.765**	0.659	0.714	0.710	0.701	0.652

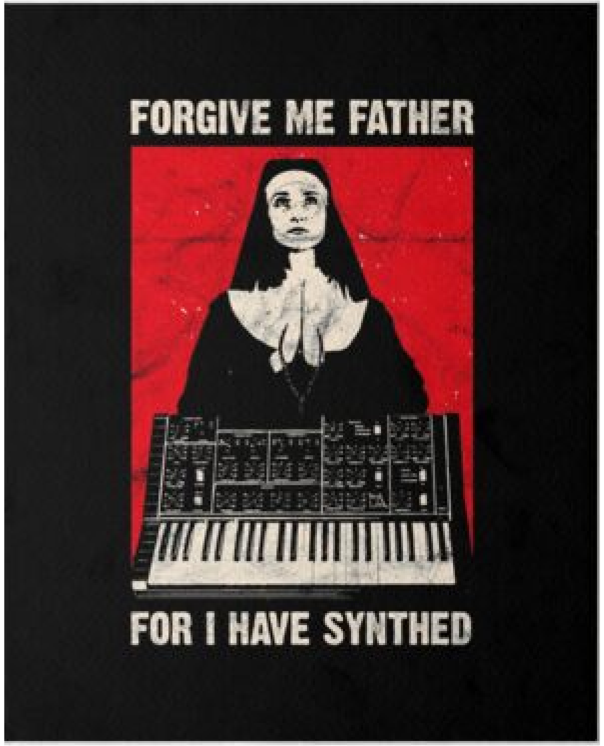
Portisch & Paulheim (2023): The RDF2vec Family of Knowledge Graph Embedding Methods

Experiments on DBpedia Gold Standard

- Challenge: isolating effects
  - Let's consider, ∃r.T: e.g. ∃almaMater.T
  - In theory, we should not be able to learn this with e-walks
  - Frequent entities in the neighborhoods of positive examples:
    - Politician (3k examples)
    - Bachelor of Arts (3k examples)
    - Harvard Law School (2k examples)
    - Lawyer (2k examples)
    - Northwestern University (2k examples)
    - Harvard University (2k examples)
    - Doctor of Philosophy (2k examples)
    - …
  - Those signals are visible to e-walks!

Which Classes can be Learned with RDF2vec?

- Maybe, DBpedia is not such a great testbed
  - Hidden patterns, e.g., for relation cooccurence
  - Many inter-pattern dependencies
  - Information not missing at random
- Possible solution:
  - Synthetic knowledge graphs!
  - First experiments show better visibility of expected effects



The DLCC Synthetic Gold Standard

- Same twelve test cases as before
- Synthesize a knowledge graph for each test case
  - Create an ontology
  - Create positive examples
  - Create negative examples (double check for accidental positives)
- Test bed
  - 12 different classification problems, 1k positives/negatives each
  - Ontology and graph structure are similar to DBpedia

Experiments on DLCC Synthetic Gold Standard

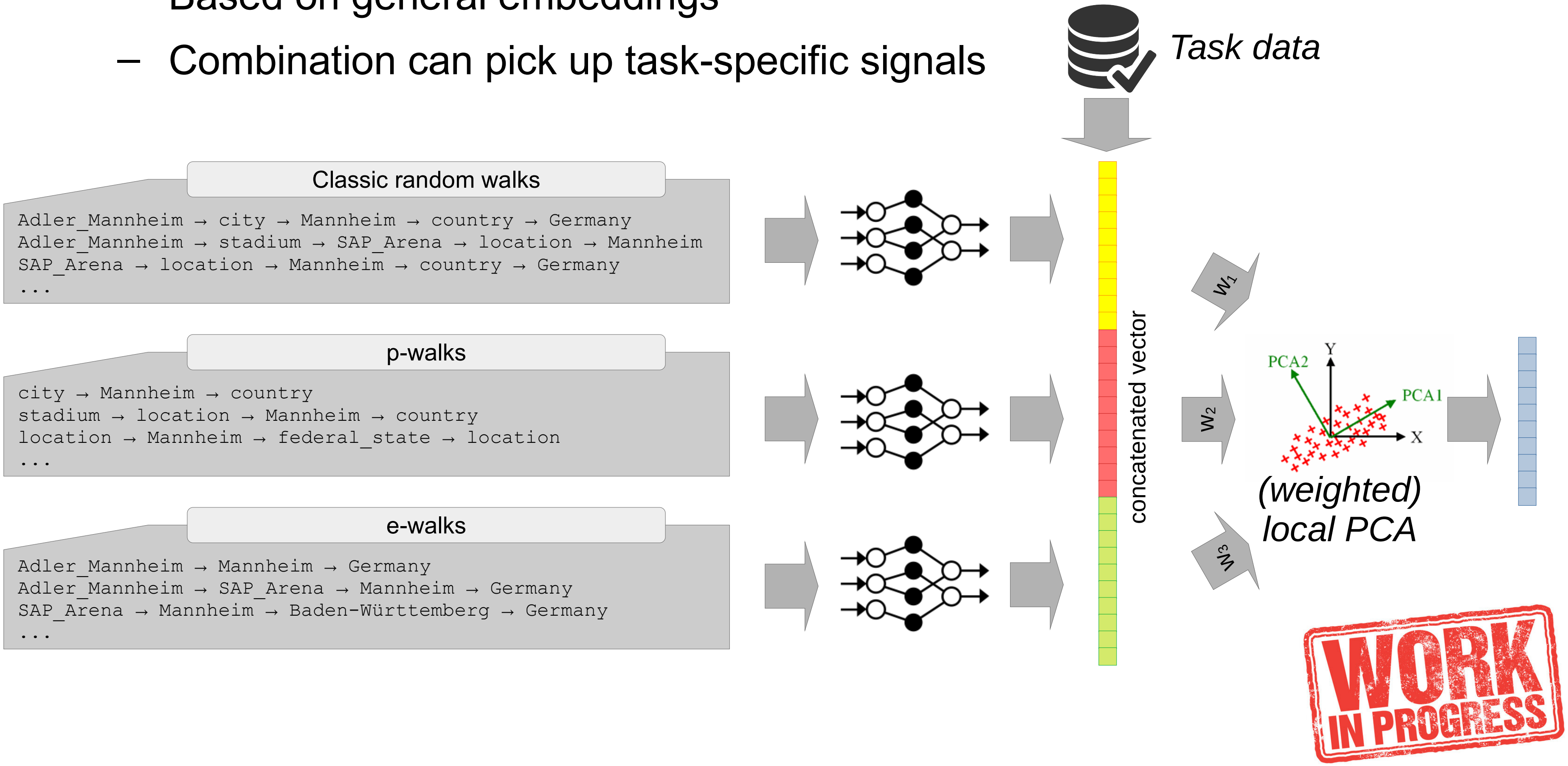
- Hypotheses can be mostly confirmed
  - Quantified restrictions (tc09-tc12) are very badly learned by all approaches (as expected)
  - tc06 is extremely well learned by LP embeddings
    - Classifying r.{e} is, in fact, classic link prediction/triple scoring
  - RDF2vec$_{oa}$ variant is not superior on synthetic data

TC	SG	SG$_{oa}$	CBOW	CBOW$_{oa}$	s-SG	s-SG$_{oa}$	s-CBOW	s-CBOW$_{oa}$	e-SG	e-SG$_{oa}$	e-CBOW	e-CBOW$_{oa}$	TransE-L1	TransE-L2	TransR	DistMult	ComplEx	RESCAL	RotatE
tc01	0.882	0.867	0.566	0.877	0.870	0.842	0.802	0.847	0.774	0.757	0.752	0.727	0.767	0.752	0.712	0.837	0.789	**0.895**	0.769
tc02	0.742	0.737	0.769	0.732	**0.822**	0.734	0.769	0.754	0.536	0.529	0.536	0.529	0.677	0.677	0.531	0.584	0.549	0.689	0.546
tc03	0.797	0.812	**0.927**	0.774	0.794	0.709	0.784	0.742	0.526	0.526	0.561	0.519	0.531	0.561	0.554	0.556	0.536	0.634	0.541
tc04	**1.000**	0.998	0.990	0.998	0.548	0.588	0.608	0.628	**1.000**	0.995	**1.000**	0.998	0.790	0.898	0.685	0.588	0.553	0.528	0.728
tc05	**0.892**	0.819	0.889	0.819	0.631	0.648	0.681	0.648	0.832	0.819	0.882	0.791	0.691	0.774	0.631	0.658	0.726	0.608	0.646
tc06	0.978	0.963	0.898	0.965	0.800	0.828	0.748	0.820	0.970	0.968	0.905	0.965	0.898	0.978	0.888	**1.000**	**1.000**	**1.000**	0.955
tc07	0.583	0.583	0.575	0.555	0.553	0.553	0.535	0.540	0.543	0.525	0.498	0.518	0.540	0.615	**0.673**	0.565	0.518	0.550	0.508
tc08	0.563	0.585	0.555	0.583	0.635	**0.628**	0.568	0.618	0.525	0.533	0.553	0.540	0.585	0.613	0.540	0.535	0.523	0.533	0.535
tc09	0.610	0.628	**0.648**	0.605	0.563	0.550	0.605	0.590	0.550	0.535	0.508	0.528	0.588	0.543	0.525	0.525	0.545	0.638	0.538
tc10	0.638	0.623	**0.665**	0.600	0.548	0.560	0.633	0.565	0.593	0.565	0.548	0.515	0.588	0.573	0.518	0.525	0.510	0.580	0.533
tc11	0.633	0.580	**0.668**	0.575	0.573	0.555	0.580	0.553	0.550	0.545	0.540	0.545	0.583	0.590	0.573	0.518	0.590	0.625	0.538
tc12	0.644	0.614	**0.657**	0.638	0.563	0.565	0.590	0.640	0.541	0.568	0.560	0.524	0.618	0.550	0.513	0.553	0.540	0.578	0.533

Portisch & Paulheim (2022): The DLCC Node Classification Benchmark for Analyzing Knowledge Graph Embeddings

The RDF2vec Zoo – Breeding New Embeddings

- Combinations can be task specific
  - Based on general embeddings
  - Combination can pick up task-specific signals

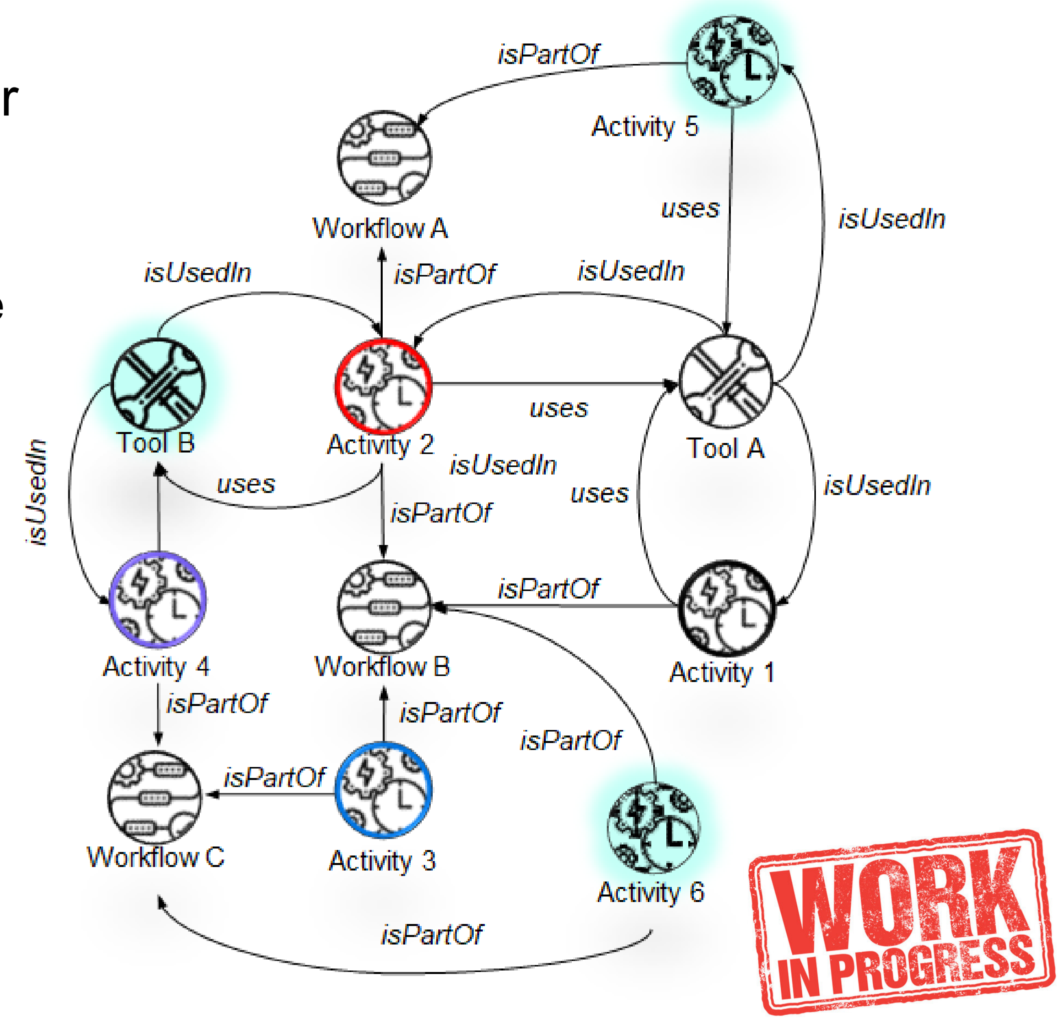


Current Challenges with RDF2vec

Dynamic Knowledge Graphs

- In theory, RDF2vec can also produce embeddings for *dynamic* knowledge graphs to a certain extent
  - given that the neighbors are all known
  - Experiments are still under way

teaming.ai

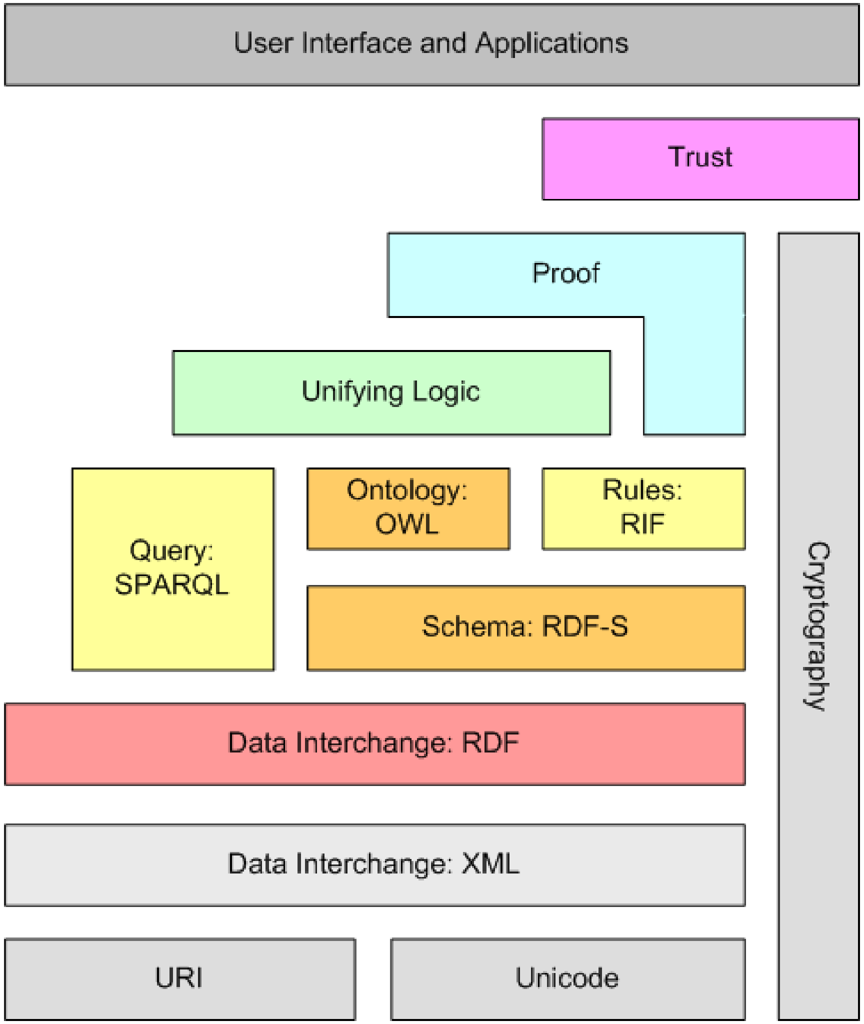


Embeddings and Interpretability

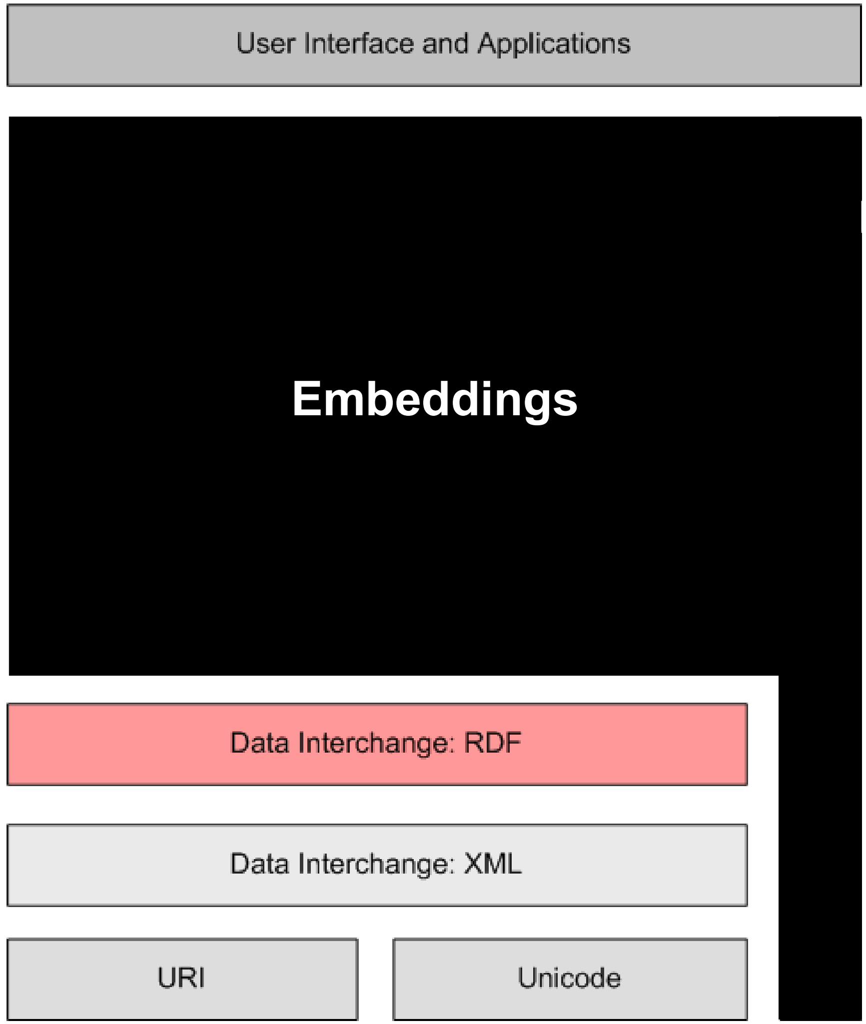
- Hot topic: Explainable AI
  - Knowledge Graphs are a favorable ingredient
  - Human/machine interpretable knowledge → explainable systems
- However:
  - Embeddings replace *interpretable axioms* with *numeric vectors over non-interpretable dimensions*
  - *Where did the semantics go?*

Paulheim (2018): Make Embeddings Semantic Again!

The 2009 Semantic Web Layer Cake



The 2018 Semantic Web Layer Cake



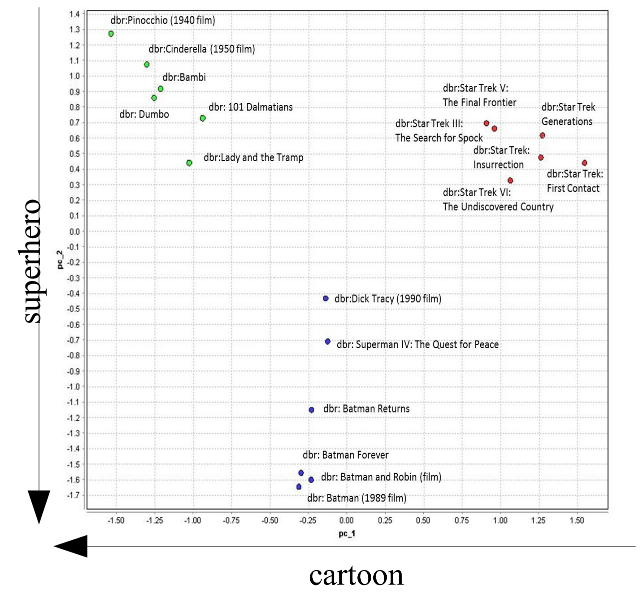
Alternatives to Understand KG Embeddings

- Approach 1: learn symbolic interpretation function for dimensions

- Each dimension of the embedding model is a target for a separate *learning problem*
- Learn a function to explain the dimension

- E.g.: $y \approx -|\exists character.Superhero|$

- Just an approximation used for explanations and justifications

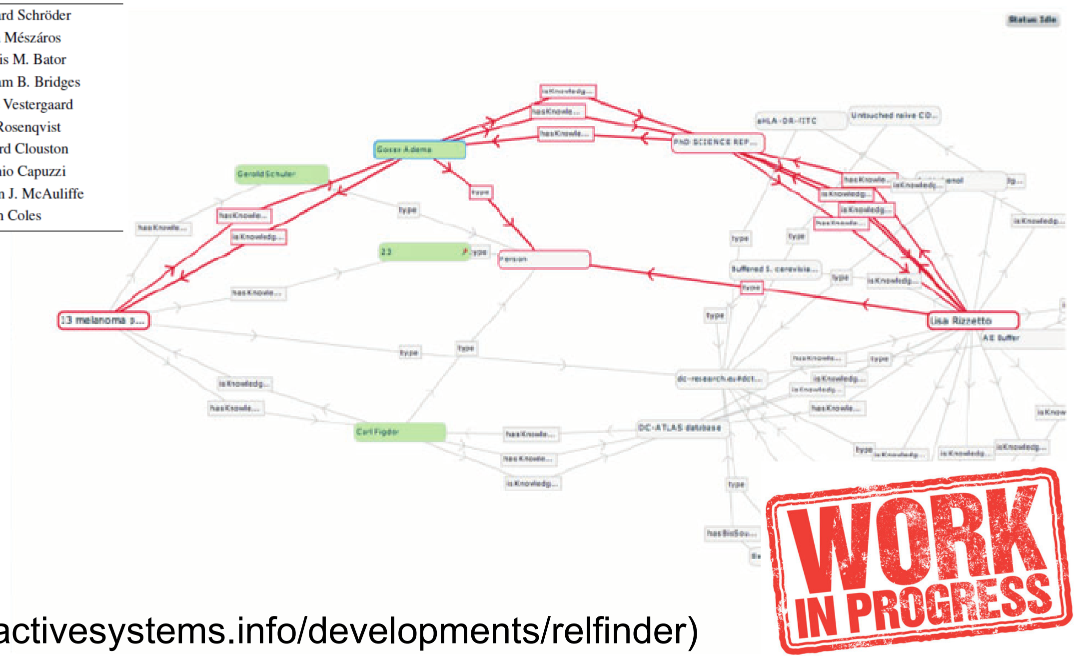


Alternatives to Understand KG Embeddings

- Approach 2: learn symbolic substitute function for similarity function

RDF2vec	TransE-L1	TransE-L2	TransR
Joachim Gauck	Gerhard Schröder	Gerhard Schröder	Sigmar Gabriel
Norbert Lammert	James Buchanan	Helmut Kohl	Frank-Walter Steinmeier
Stanislaw Tillich	Neil Kinnock	Konrad Adenauer	Philipp Rösler
Andreas Voßkuhle	Nicolas Sarkozy	Helmut Schmidt	Gerhard Schröder
Berlin	Joachim Gauck	Werner Faymann	Joachim Gauck
German language	Jacques Chirac	Alfred Gusenbauer	Christian Wulff
Germany	Jürgen Trittin	Kurt Georg Kiesinger	Guido Westerwelle
federalState	Sigmar Gabriel	Philipp Scheidemann	Helmut Kohl
Social Democratic Party	Guido Westerwelle	Ludwig Erhard	Jürgen Trittin
deputy	Christian Wulff	Wilhelm Marx	Jens Böhrnsen

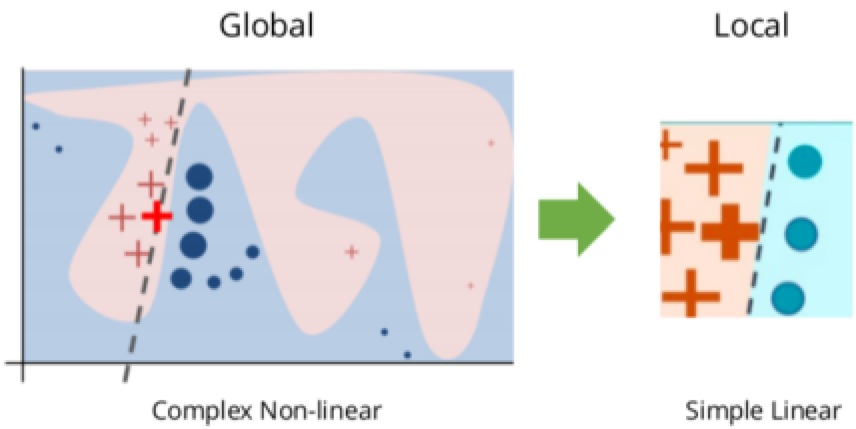
RotatE	DistMult	RESCAL	ComplEx
Pontine raphe nucleus	Gerhard Schröder	Gerhard Schröder	Gerhard Schröder
Jonathan W. Bailey	Milan Truban	Kurt Georg Kiesinger	Diána Mészáros
Zokwang Trading	Maud Cuney Hare	Helmut Kohl	Francis M. Bator
Steven Hill	Tristan Matthiae	Annemarie Huber-Hotz	William B. Bridges
Chad Kreuter	Gerda Hasselfeldt	Wang Zhaoguo	Mette Vestergaard
Fred Hibbard	Faustino Sainz Muñoz	Franz Vranitzky	Ivan Rosenqvist
Mallory Ervin	Joachim Gauck	Bogdan Klich	Edward Clouston
Paulinho Kobayashi	Carsten Linnemann	İrsen Küçük	Antonio Capuzzi
Fullmetal Alchemist and the Broken Angel	Norbert Blüm	Helmut Schmidt	Steven J. McAuliffe
Archbishop Dorotheus of Athens	Neil Hood	Mao Zedong	Jenkin Coles



Right hand side picture: RelFinder (https://interactivesystems.info/developments/relfinder)

Alternatives to Understand KG Embeddings

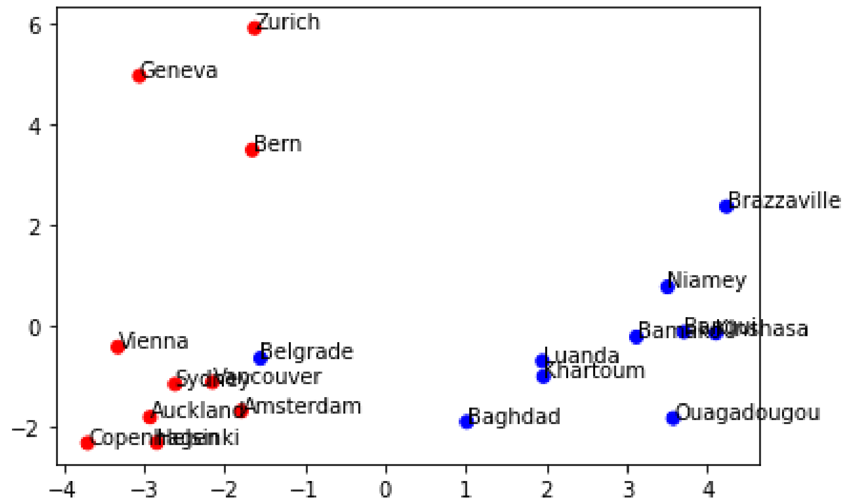
- Approach 3: generate symbolic interpretations for individual predictions
  - Inspired by LIME:
    - Generate perturbed examples
    - Label them using embedding+downstream classifier
    - Learn symbolic model on this labeled set
  - Good news:
    - RDF2vec can, in principle, create embeddings for unseen entities
    - Those can be used to classify perturbed examples
  - Bad news:
    - First experiments have been discouraging



https://c3.ai/glossary/data-science/lime-local-interpretable-model-agnostic-explanations/

Dealing with non-Relational Information

- Graph walks do not include literal (e.g., numeric) information
  - e.g., population
- Two cities are closer
  - If they share relations and entities with others
  - not: if they have a similar population



Summary

- Knowledge Graph Embeddings with RDF2vec
  - Encode similarity and relatedness
    - Explicit trade-off is possible!
  - Variations visited: walk extraction, order-awareness, materialization, ...
  - Additional insights that are not explicit in the graph
    - aka *latent semantics*
  - Challenges include, but are not limited to
    - Dynamic knowledge graphs
    - Interpretability

More on RDF2vec

- Collection of
  - Implementations
  - Pre-trained models
  - >50 documented use cases in various domains

About Implementations Models and Services Extensions Other Resources Applications References Acknowledgements Contact

RDF2vec.org

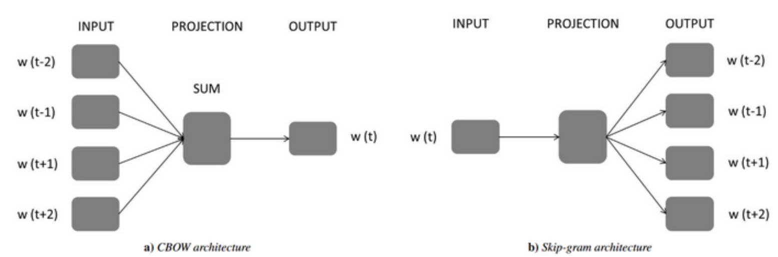
The hitchhiker's guide to RDF2vec.

About RDF2vec

RDF2vec is a tool for creating vector representations of RDF graphs. In essence, RDF2vec creates a numeric vector for each node in an RDF graph.

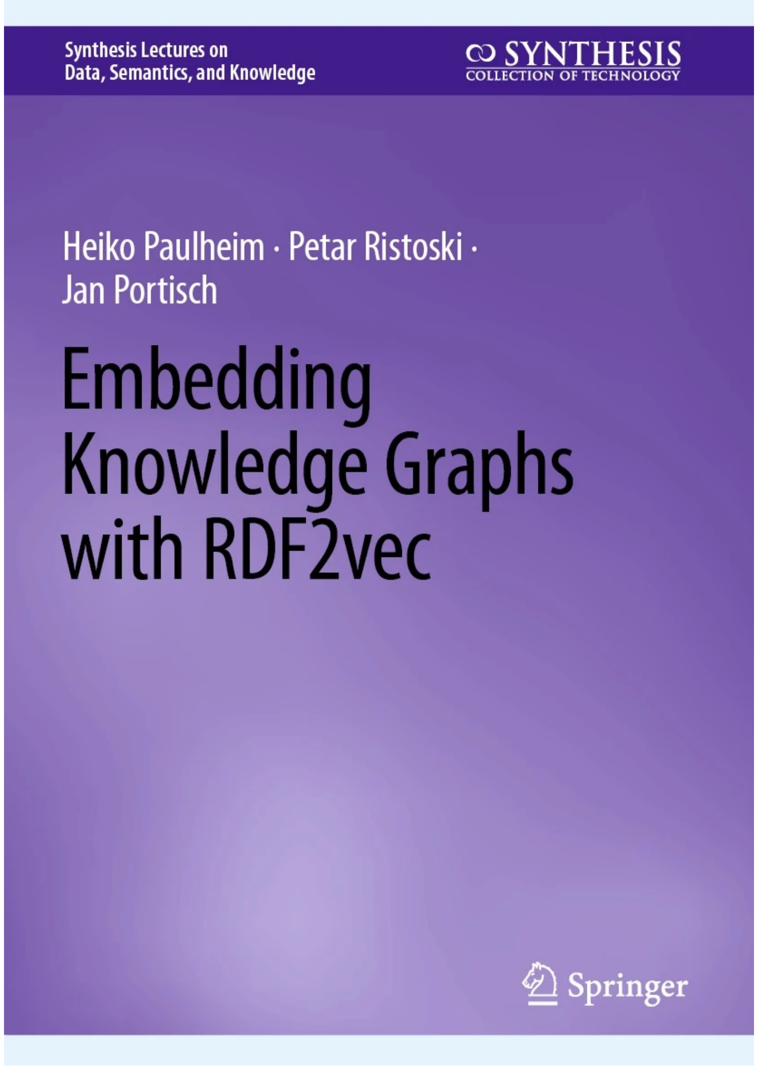
RDF2vec was developed by Petar Ristoski as a key contribution of his PhD thesis Exploiting Semantic Web Knowledge Graphs in Data Mining [Ristoski, 2019], which he defended in January 2018 at the Data and Web Science Group at the University of Mannheim, supervised by Heiko Paulheim. In 2019, he was awarded the SWSA Distinguished Dissertation Award for this outstanding contribution to the field.

RDF2vec was inspired by the word2vec approach [Mikolov et al., 2013] for representing words in a numeric vector space. word2vec takes as input a set of sentences, and trains a neural network using one of the two following variants: predict a word given its context words (continuous bag of words, or CBOW), or to predict the context words given a word (skip gram, or SG):



More on RDF2vec

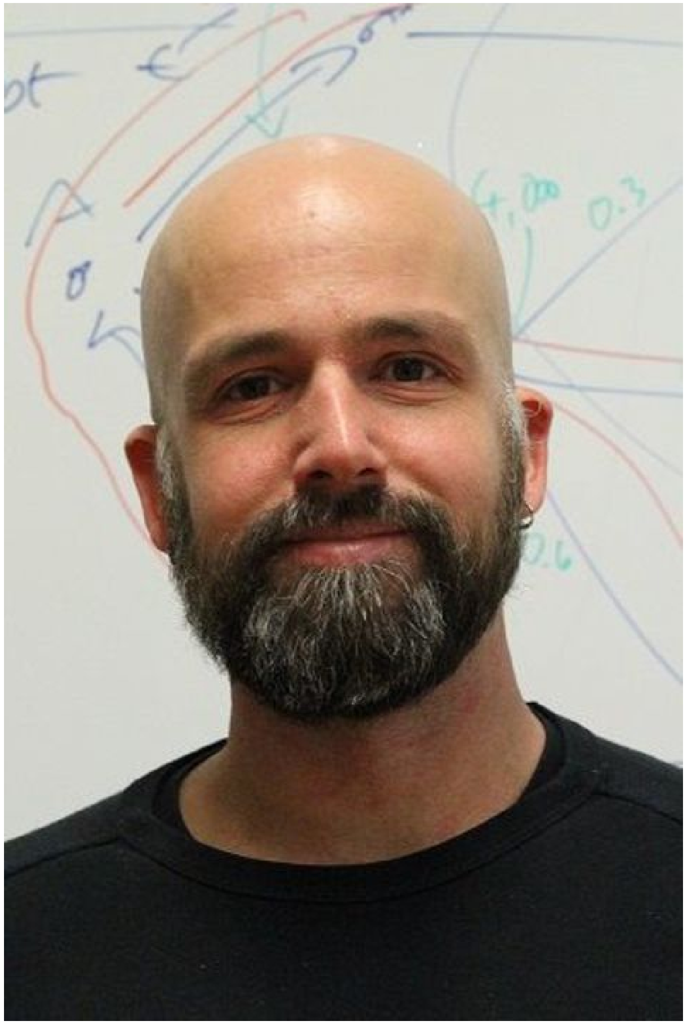
- Text book on RDF2vec
  - Includes all variants discussed in this talk
  - Python cookbooks for common tasks, e.g., node classification, recommender systems, …



Thank you!

http://www.heikopaulheim.com

@heikopaulheim



UNIVERSITÄT MANNHEIM

New Adventures in RDF2vec

Heiko Paulheim
University of Mannheim