UNIVERSITÄT BIELEFELD

Knowledge Graph Completion

Introduction to the Semantic Web

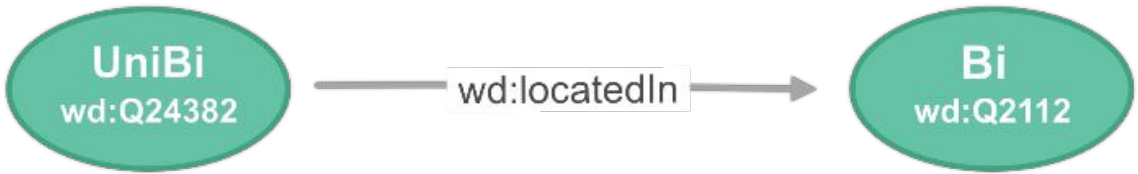
2023-07-17 | Moritz Blum

Recap: KG

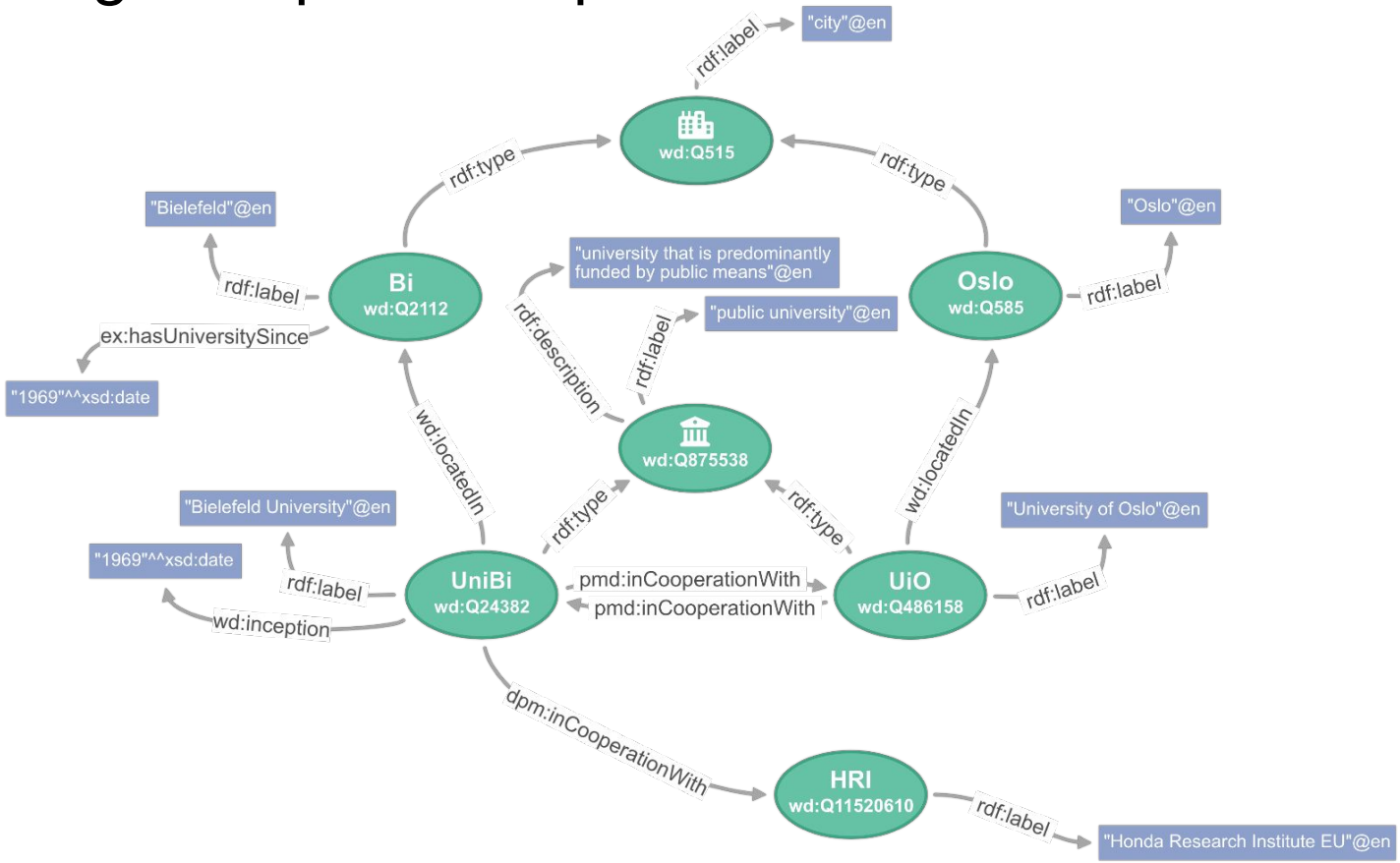
A KG is a graph-structured data model that is used to store and structure information.

Entities (e.g. objects, events, abstract concepts) are connected by relations. Literals are used to store entity property values such as strings, numbers, and dates.

- a set of facts in form of triples *(head entity, relation, tail entity/literal)*, where
  - head from a set of entities
  - relation from a set of relations
  - tail from a set of entities or a literal



Knowledge Graph: Example

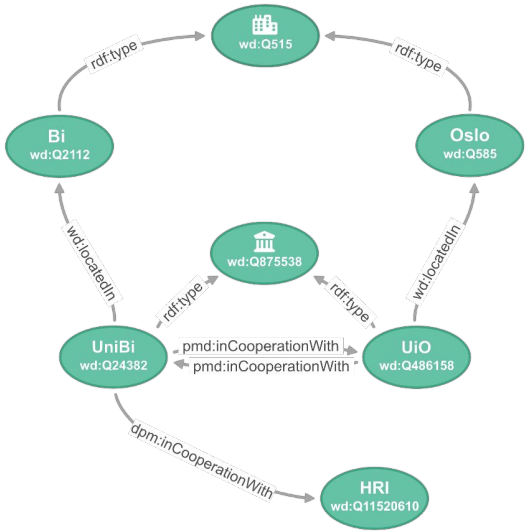


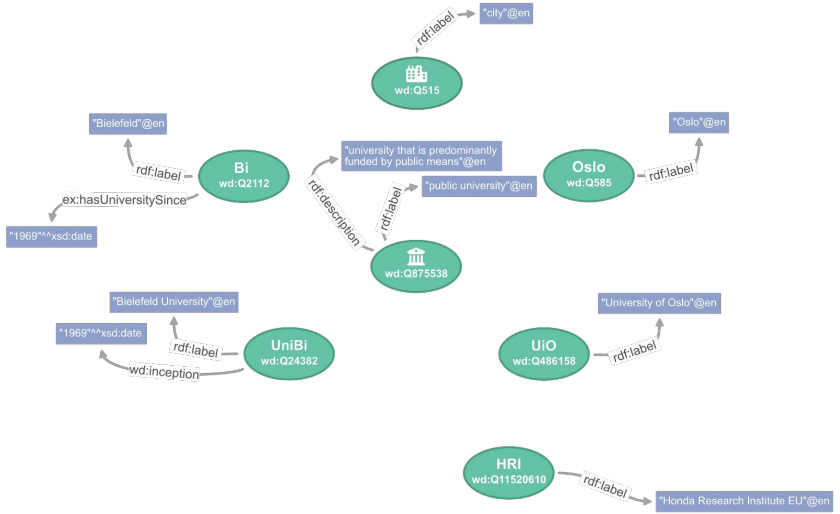
Motivation for Knowledge Graph Completion

- KGs are inherently incomplete (in Freebase e.g. 70% of person entities have no known birthplace, and over 99% have no known ethnicity)
- Task of identifying new facts: Knowledge Graph Completion
  - Graph based methods: Relation Classification, Entity Typing, LP
  - Text based methods: NLP Information Extraction (Entity Linking, Relation Extraction)
- Challenge: scalability to large KGs

Our Data

$$G = \underbrace{\{(s,p,o) \mid (s,p,o) \in G \text{ s.t. } o \in \mathcal{U} \cup \mathcal{B}\}}_{\text{relational triples } G_E} \cup \underbrace{\{(s,p,o) \mid (s,p,o) \in G \text{ s.t. } o \in \mathcal{L}\}}_{\text{attributive triples } G_L}$$





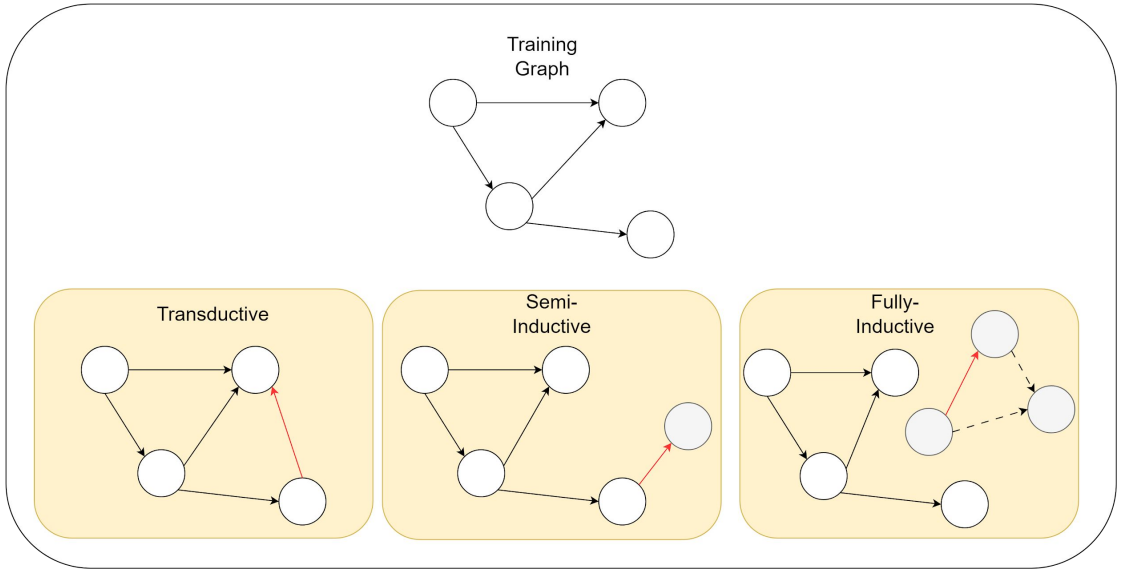
Settings

Link prediction comes in multiple settings:

- transductive: relations only between known entities (most prominent and usually meant by Link Prediction)
- inductive: relations between unknown entities, as well
  - semi inductive vs. fully inductive
  - graph based vs. description based

Approaches to solve Link Prediction can be classified into:

- Rule based
- Embedding based
- Hybrid (a combination of both)
- Language model based



Rule-based

association rule mining is well-known in the context of sales databases

for KGs: rules to deduce missing knowledge

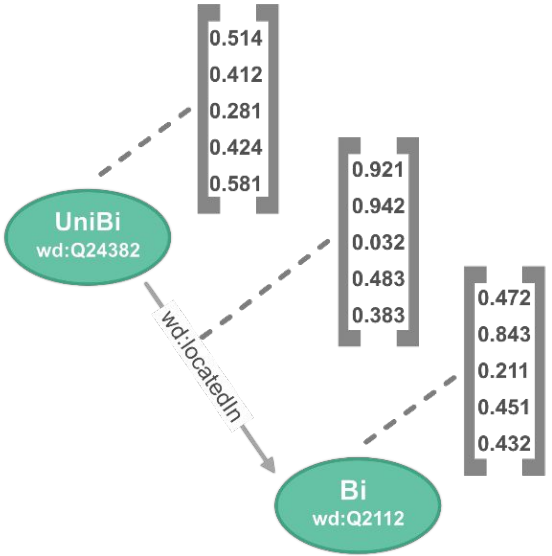
as for any rule, there are usually exceptions, but in we assume that in the vast majority of cases, the rule will hold

Example AMIE - Association rule mining under incomplete evidence in ontological knowledge bases

one of the main problems is to find an **efficient** way to explore the search space as a naive algorithm of enumerating all possible rules is infeasible for large KGs, therefore the search space is iteratively explored by extending rules

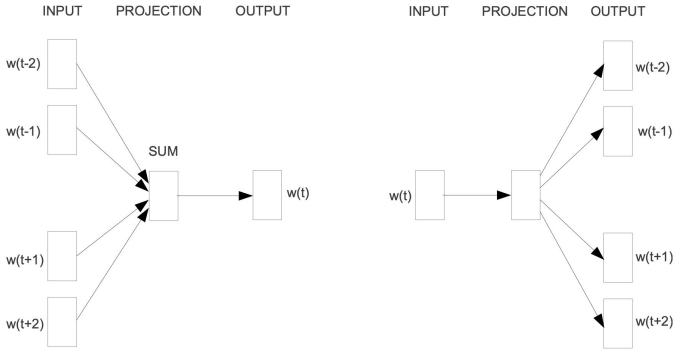
Embedding-based

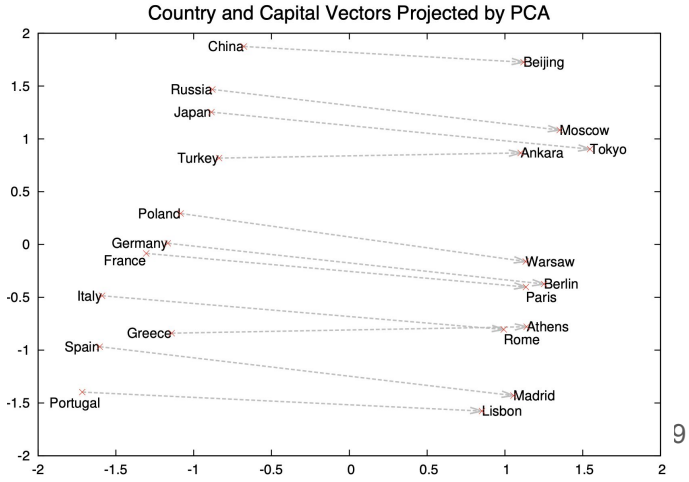
- convert KGs into a low dimensional vector space while preserving the graph properties
- embeddings should model semantics present in the graph as good as possible
- learned automatically, based on how the corresponding elements **occur and interact with each other** in the dataset
- in practice, the embeddings are learned on known facts only, but they should be able to generalize and associate high scores to unseen true facts, as well.



Embedding-based - Word embeddings

- Convert KGs into a low dimensional vector space while preserving the graph properties - similar or related entities should be located close to each other
- Embeddings are not new, widely used in NLP: Word2Vec (Skip-gram & CBOW)
- For KGs: RDF2Vec



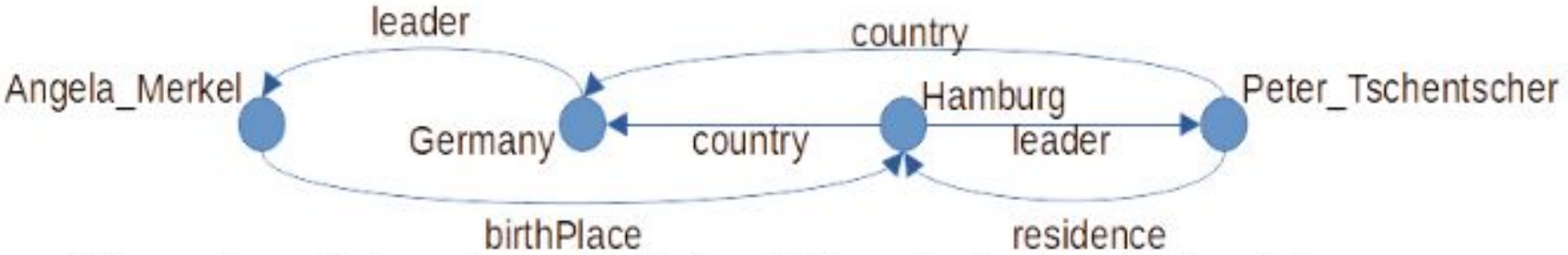


Mikolov, Tomas; et al. (2013). "Efficient Estimation of Word Representations in Vector Space". arXiv:1301.3781

Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S.; Dean, Jeff (2013). *Distributed representations of words and phrases and their compositionality*. Advances in Neural Information Processing Systems. arXiv:1310.4546

Petar Ristoski, Heiko Paulheim: RDF2Vec: RDF Graph Embeddings for Data Mining. International Semantic Web Conference, 2016

Embedding-based - RDF2Vec Walk Generation

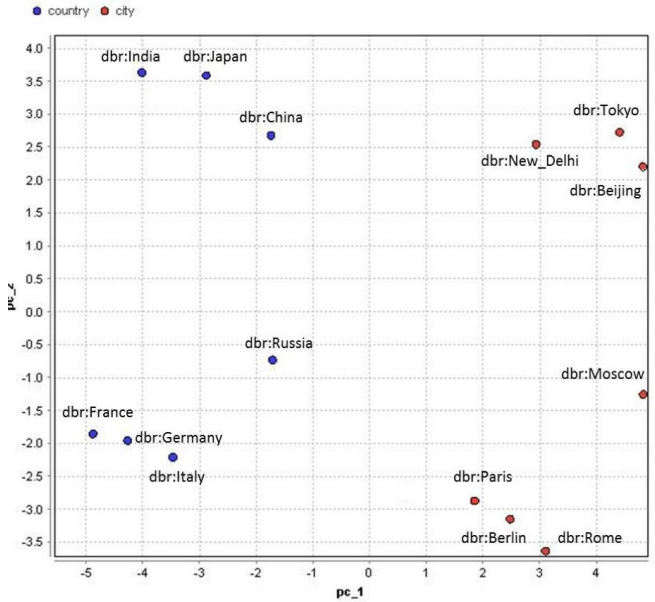


From this graph, a set of random walks that could be extracted may look as follows:

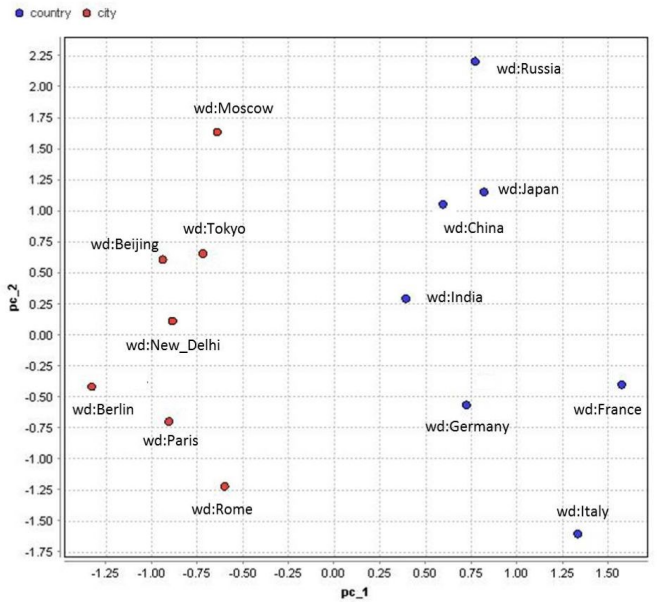
```
Hamburg -> country -> Germany           -> leader     -> Angela_Merkel
Germany -> leader  -> Angela_Merkel     -> birthPlace -> Hamburg
Hamburg -> leader  -> Peter_Tschentscher -> residence  -> Hamburg
```

http://rdf2vec.org/

Embedding-based - RDF2Vec



a) *DBpedia vectors*



b) *Wikidata vectors*

→ Not suitable for LP. Therefore, models are trained with the LP objective directly.

Petar Ristoski, Heiko Paulheim: RDF2Vec: RDF Graph Embeddings for Data Mining. International Semantic Web Conference, 2016

Knowledge Graph Embeddings for Link Prediction vs. Knowledge Graph Embeddings for Data Mining

projections of entities and relations to lower dimensional spaces, have been proposed for two purposes

1) Encoding for Data Mining (RDF2Vec)
2) Link Prediction

Both lines of research have been pursued rather in isolation from each other.

However, both tasks are somehow related, and it was shown that both types of embeddings can be used for the other task somehow.

Data Mining Embeddings = semantic relatedness (Olaf Scholz ~ Angela Merkel) & (Olaf Scholz ~ Germany)

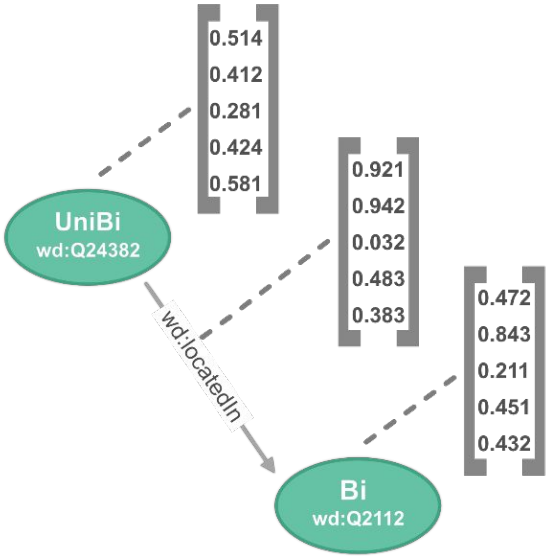
Link Prediction = semantic similarity (Olaf Scholz ~ Angela Merkel) & (Olaf Scholzl !~ Germany)

Semantic similarity is a special case of semantic relatedness! Benefits and drawbacks depending on the application.

Embedding-based LP

General: a link prediction model defines a scoring function $f(\overrightarrow{e_{head}}, \overrightarrow{e_{relation}}, \overrightarrow{e_{tail}})$, that estimate the plausibility of a triple (head, relation, tail) using the embeddings

- geometric
- matrix factorization
- deep learning



Embedding-based

- machine learning task
- training: usually, embeddings are initialized randomly, and then optimized using back-propagation with gradient descent
- loss function: maximize the plausibility of true facts while minimizing it for false facts

Unsupervised Training

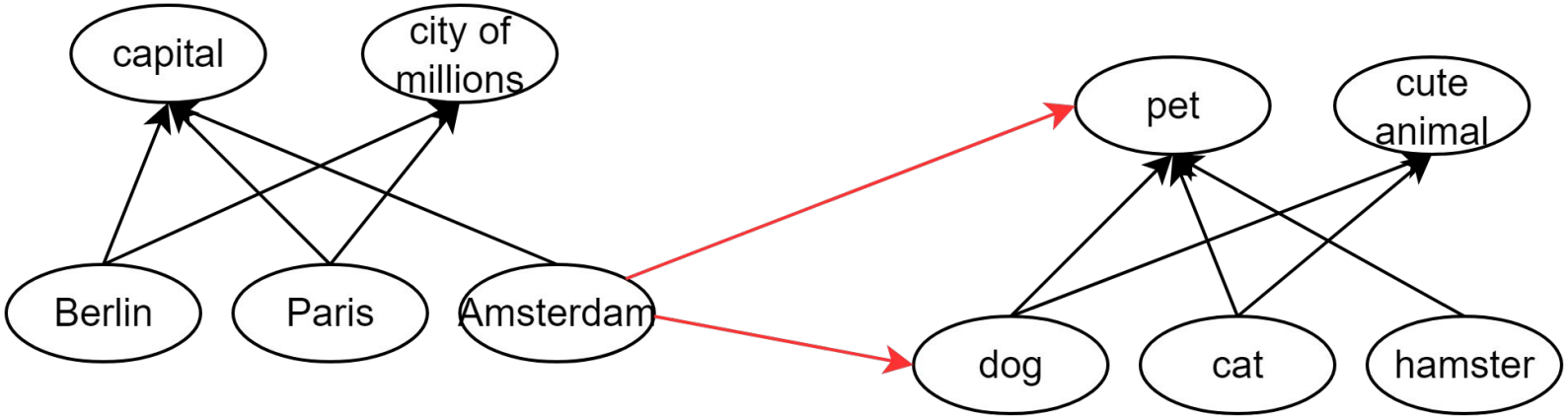
- learning relies on contrasting positive and negative facts (triples)
- positive facts = our KG
- negative facts?

Unsupervised Training

- learning relies on contrasting positive and negative facts (triples)
- positive facts = our KG
- negative facts = randomly samples triples by replacing head or tail with random entities
- problem with generating negative facts?

Unsupervised Training

- learning relies on contrasting positive and negative facts (triples)
- positive facts = our KG
- negative facts = randomly samples triples by replacing head or tail with random entities
- problem with generating negative facts:
  - Open World Assumption → however, it is expected that most of them are false
  - Some negative triples trivial to identify as false, e.g. the “capital of” relation can not hold between a “city” and a type of “animal” → advanced sampling methods were proposed



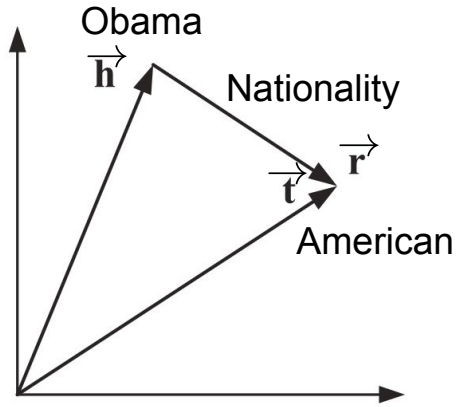
Embedding-based Link Prediction - Geometric

These models interpret relations as **geometric operations** in the latent space (inspired by analogies found in word2vec vectors).

The head embedding undergoes a spatial transformation depending on the value of the relation embedding. The fact score is the distance between the resulting vector and the tail vector.

Geometric Example: TransE (Borders et al. 2013)

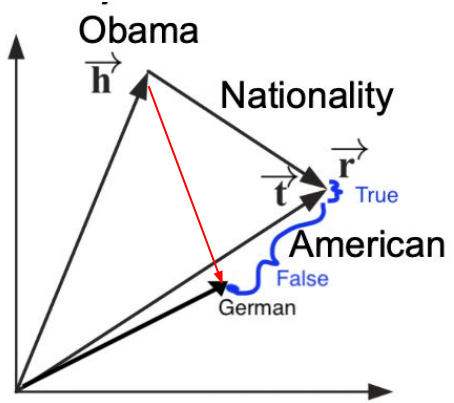
TransE was the first geometric LP model, inspired by the translational properties of Word2Vec embeddings.



The inverse fact score is the distance between the addition of head and relation embedding and the tail vector:

$$f_{TransE}(h, r, t) = d(\overrightarrow{e_h} + \overrightarrow{e_r} - \overrightarrow{e_t})$$

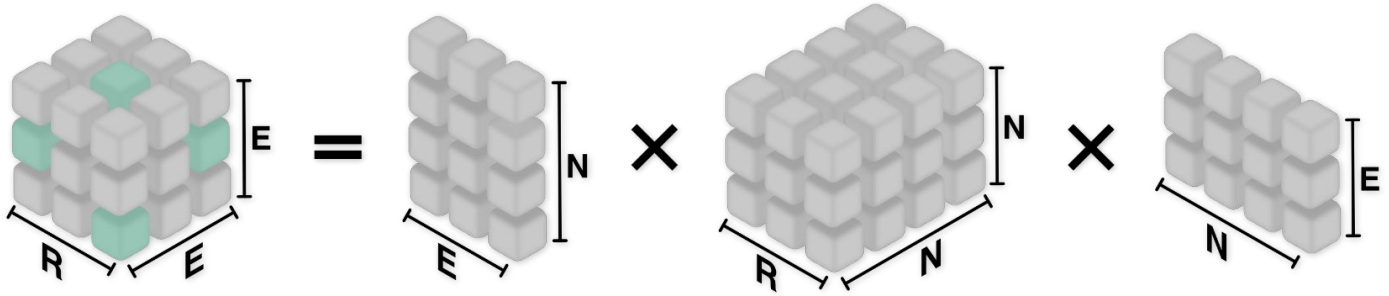
The model can be trained by minimizing this distance for triples in the training set, and maximizing the distance for corrupted triples.



Embedding-based Link Prediction - Tensor Factorization

Link Prediction as a task of tensor decomposition, where the KG is a three-dimensional adjacency matrix $E \times E \times R$.

This tensor can be decomposed into a combination of low-dimensional vectors, i.e., the embeddings of entities and relations.



Tensor Factorization Example: DistMult (Yang et al. 2014)

A triple is scored by the dot product of the learned embedding vectors, where the relation embedding matrices are restricted to diagonal matrices:

$$f_{DistMult}(h,r,t) = \overrightarrow{e_h} \cdot diag(\overrightarrow{e_r}) \cdot \overrightarrow{e_t}$$

The model is trained by minimizing the Pairwise Ranking Loss

$$\mathcal{L}_{DistMult}(h,r,t) = \sum_{(h,r,t)\in G} \sum_{(h',r,t')\in G_{corr}} |\gamma - f(h,r,t) + f(h',r,t')$$

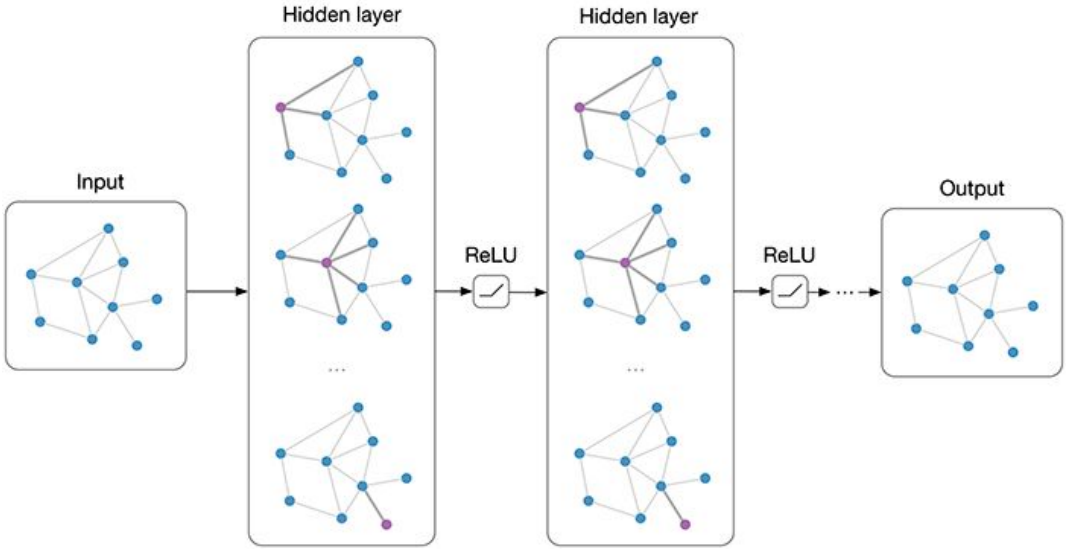
where γ is a margin hyperparameter. The loss enforces negative triples to have smaller scores than positive ones.

DistMult is able to achieve state of the art comparable performance, even though the scoring function is commutative (treating all relations as symmetric).

Embedding-based Link Prediction - Deep Learning

Deep Neural Networks have multiple neural layers, generally interspersed with non-linear activation functions. There are different types of neural layers, e.g.:

- linear dense layers: input data X is combined with weights W and a bias b
- graph convolution layer: graph message passing protocol with convolutional filters

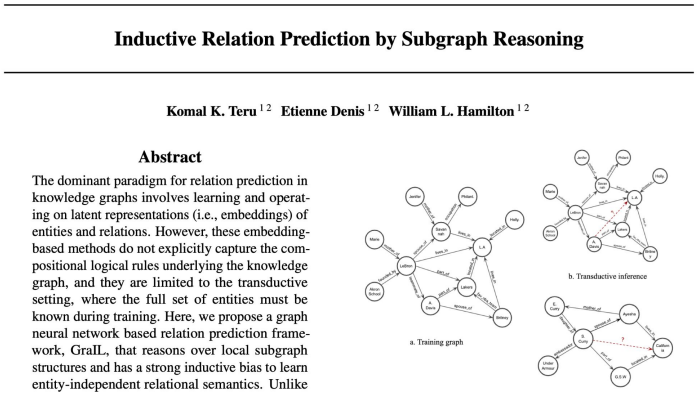


Deep Learning Example: R-GCNs (Relational Graph Convolutional Networks - Schlichtkrull 2017)

- R-GCNs serve as an effective encoder for relational data.
- Encoder-Decoder architecture. R-GCN Encoder + LP Decoder
- Drawback: scalability

Hybrid LP models: Embeddings + Rules

- models that use mined rules as node features
- models that use rules to weight the importance of certain relations/neighboring nodes
- models that are trained to pay attention to logical rules



Inductive Relation Prediction by Subgraph Reasoning

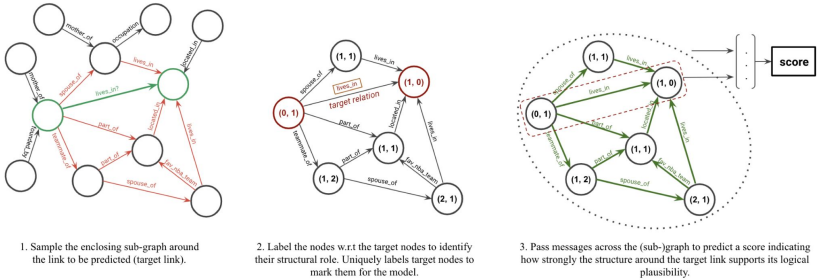
Komal K. Teru [1 2] Etienne Denis [1 2] William L. Hamilton [1 2]

Abstract

The dominant paradigm for relation prediction in knowledge graphs involves learning and operating on latent representations (i.e., embeddings) of entities and relations. However, these embedding-based methods do not explicitly capture the compositional logical rules underlying the knowledge graph, and they are limited to the transductive setting, where the full set of entities must be known during training. Here, we propose a graph neural network based relation prediction framework, GraIL, that reasons over local subgraph structures and has a strong inductive bias to learn entity-independent relational semantics. Unlike

a. Training graph

b. Transductive inference



1. Sample the enclosing sub-graph around the link to be predicted (target link).

2. Label the nodes w.r.t the target nodes to identify their structural role. Uniquely labels target nodes to mark them for the model.

3. Pass messages across the (sub-)graph to predict a score indicating how strongly the structure around the target link supports its logical plausibility.

neighborhood sampling and relabeling these nodes with respect to head and subject → allows to model to generalize from concrete neighboring nodes more to the structure and relations in the neighborhood

Language Models for LP

KG-BERT = BERT trained on LP completely based on entity descriptions

Drawback: Strukture is learned only implicitly, depends on the quality of the descriptions

Benefit: generalizes to new entities

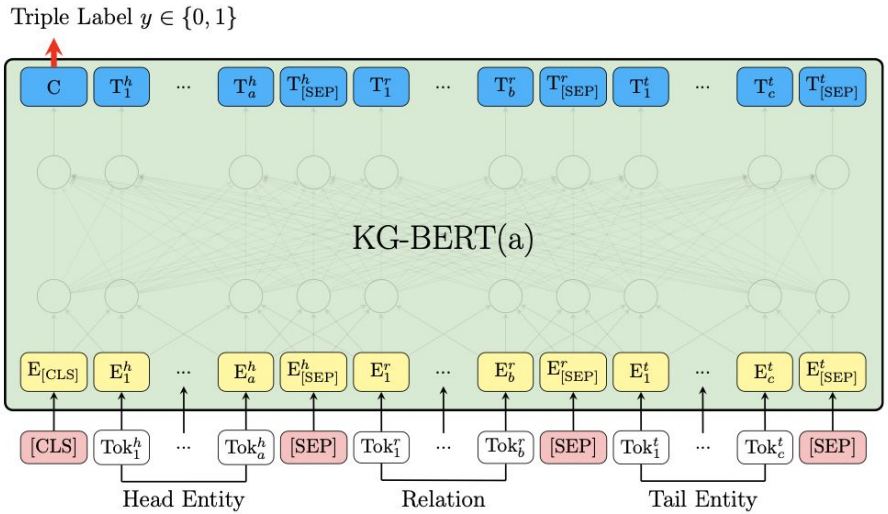
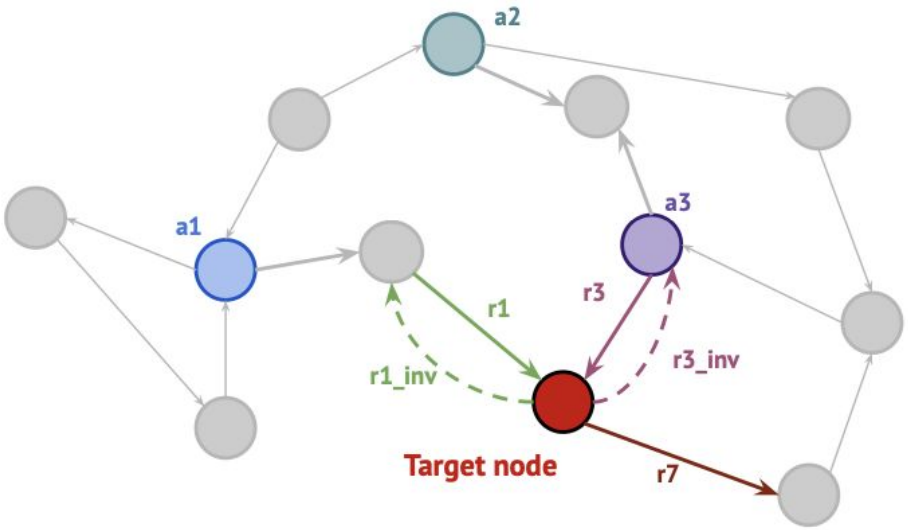


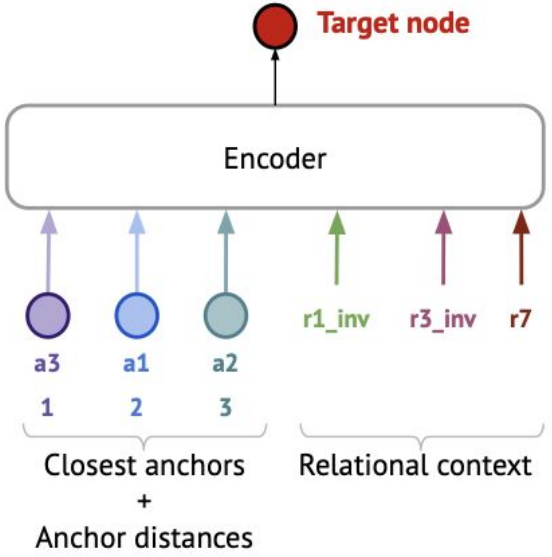
Figure 1: Illustrations of fine-tuning KG-BERT for predicting the plausibility of a triple.

Yao, Liang, Chengsheng Mao, and Yuan Luo. "KG-BERT: BERT for knowledge graph completion." *arXiv preprint arXiv:1909.03193* (2019).

Inductive LP based on Graph Structure: NodePiece

In order to generalize to new entities, NodePiece extracts features from the graph which are used by the LP model instead of one-hot-encoding.





Galkin, Mikhail, et al. "Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs." *arXiv preprint arXiv:2106.12144* (2021).

Evaluation Metrics

For a triple (head, relation, tail), the rank is defined as

$$r_{(h,r,t)} = \underbrace{|f(h,r,t') > f(h,r,t)|t' \in E \backslash \{t\}|}_{\text{tail corruption}} + \underbrace{|f(h',r,t) > f(h,r,t)|h' \in E \backslash \{h\}|}_{\text{head corruption}} + 1$$

Ranking of a valid triple across corrupted head and corrupted tail entities. A good model leads to low ranks, a bad model to high ranks. The minimum rank is 1, the maximum rank is $2 \cdot |E|$.

Q denotes the set of ranks for a set of triples, e.g., for G_{test} .

Evaluation Metrics

Mean rank
average of the obtained ranks

$$MR = \frac{1}{|Q|} \sum_{q \in Q} q$$

Mean reciprocal rank
average of the reciprocal ranks

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{q}$$

Hits@K
number of true triples ranking in the K most likely predicted triples
(usually reported: Hits@1, Hits@3, Hitls@10)

$$Hits@K = \frac{|q \in Q : q \leq K|}{|Q|}$$

Datasets

- usually obtained by sampling real-world KGs
- split in G_train, G_validation, G_test

Most prominent benchmark datasets for transductive LP:

- FB15k - derived from freebase
- FB15k-237 - variant of FB15k where inverse relations are removed
- YAGO3-10 - derived form YAGO (contains entities associated with at least ten different relations)

Most prominent benchmark datasets for inductive LP:

- Wikidata5m
- ILPC

Comparison

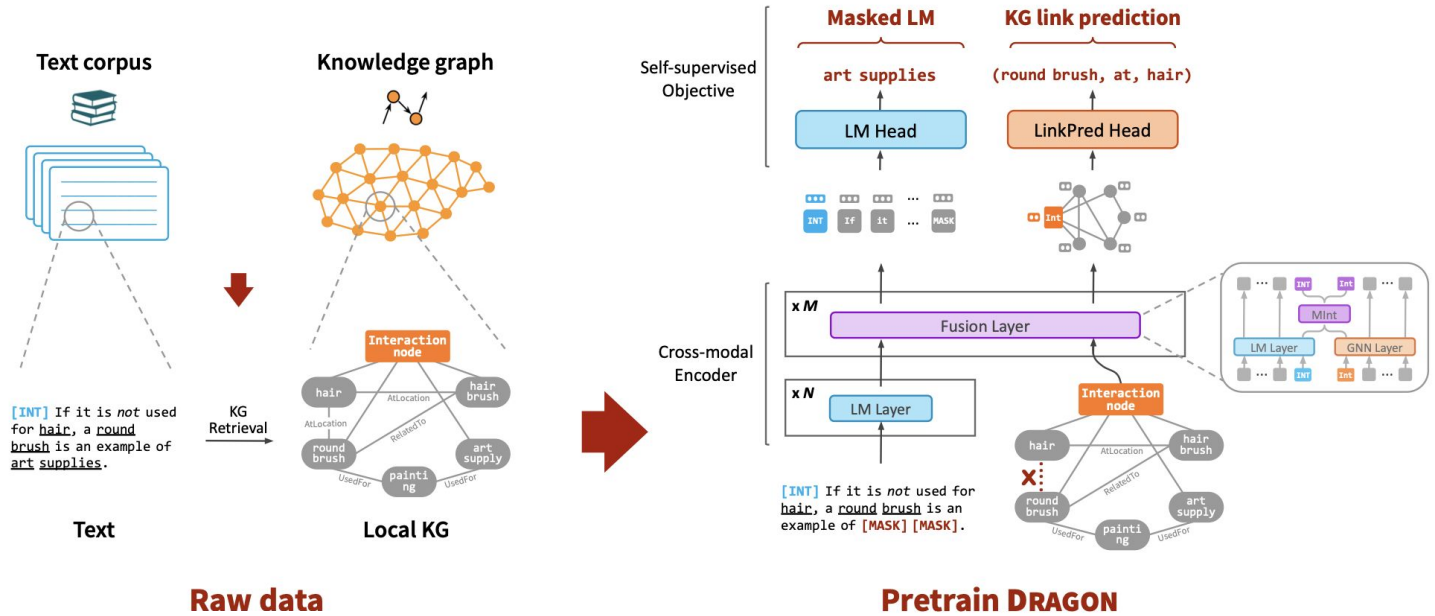
embedding based approaches have shown the best scalability and generalizability on large KGs

scores are quite similar (choose model depending on your task)

	FB15k					WN18				
	MRR		Hits @			MRR		Hits @		
Model	Raw	Filtered	1	3	10	Raw	Filtered	1	3	10
LinkFeat		0.779			0.804		0.938			0.939
DistMult	0.248	0.634	0.522	0.718	0.814	0.526	0.813	0.701	0.921	0.943
R-GCN	0.251	0.651	0.541	0.736	0.825	0.553	0.814	0.686	0.928	0.955
R-GCN+	**0.262**	**0.696**	**0.601**	**0.760**	**0.842**	0.561	0.819	0.697	0.929	**0.964**
CP*	0.152	0.326	0.219	0.376	0.532	0.075	0.058	0.049	0.080	0.125
TransE*	0.221	0.380	0.231	0.472	0.641	0.335	0.454	0.089	0.823	0.934
HolE**	0.232	0.524	0.402	0.613	0.739	**0.616**	0.938	0.930	**0.945**	0.949
ComplEx*	0.242	0.692	0.599	0.759	0.840	0.587	**0.941**	**0.936**	**0.945**	0.947

LP as additional Objective for Language Models

KGs provide valuable knowledge for various NLP applications
→ research about fusion of KGs and Text



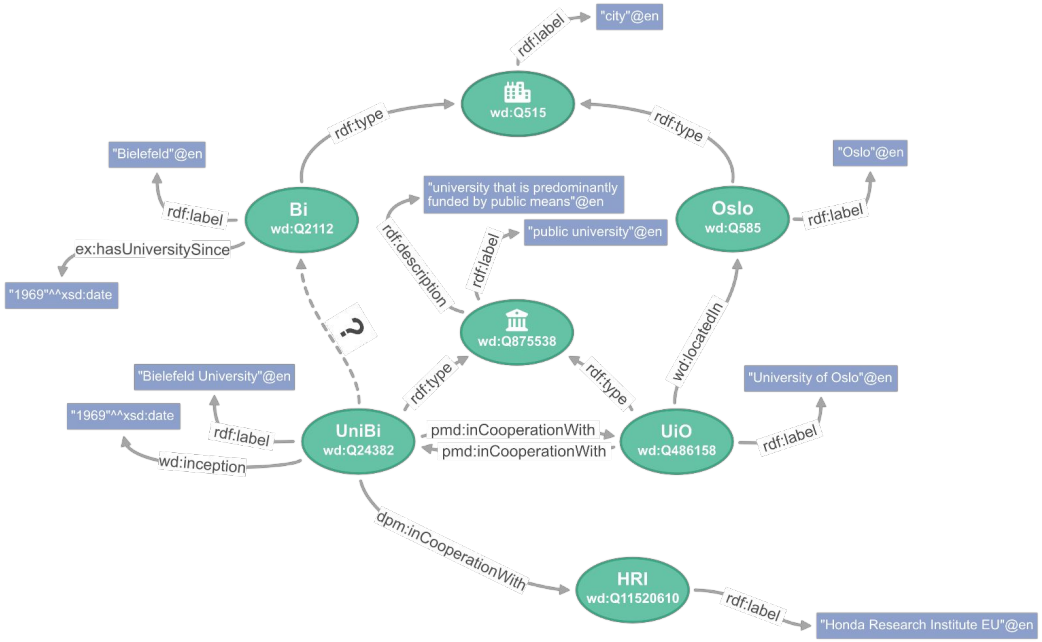
Yasunaga, Michihiro, et al. "Deep bidirectional language-knowledge graph pretraining." *Advances in Neural Information Processing Systems* 35 (2022): 37309-37323.

Problem with current approaches

- combination of embedding based link prediction approaches with different sources of information, e.g. textual descriptions
- struggling with entities that appear in a low number of relations

Link Prediction with literals

- literals are used in many KGs very frequently, e.g.,
  ~40% of DBpedia,
  ~57% of Freebase, and
  ~41% Wikidata
  triples are attributive triples
- literals can hold relevant information for Link Prediction
- most Link Prediction models and benchmark datasets do not take literal information into account as they add additional complexity to the task
- types of literals: text, numeric, units of measurement (numeric), images, other (e.g. external URI links to e.g. audio, video, or PDF files)



Related Work

- **DistMult** (Yang et al. in 2014) - interaction between two entity and one relation embedding

$$f_{DistMult}(h,r,t) = \overrightarrow{e_h} \cdot diag(\overrightarrow{e_r}) \cdot \overrightarrow{e_t}$$

- **LiteralE** (Kristiadi et al. 2018) - adds a function g that takes an entity's embedding and a literal vector and maps them to a vector of the same dim as the entity embedding

$$f^{LiteralE}_{DistMult}(h,r,t) = g(\overrightarrow{e_h}, \overrightarrow{l_h}) \cdot diag(\overrightarrow{e_r}) \cdot g(\overrightarrow{e_t}, \overrightarrow{l_t})$$

- **MKBE** (Pezeshkpour et al. 2018) - adds datatype specific encoders g_i , where g_i embeds t of type i

$$f^{MKBE}_{DistMult}(h,r,t) = \overrightarrow{e_h} \cdot diag(\overrightarrow{e_r}) \cdot g_i(\vec{t})$$

Method

- Modifications of existing models or loss functions or completely new architectures → requires a new implementation or re-implementation of core components and is only applicable to certain models

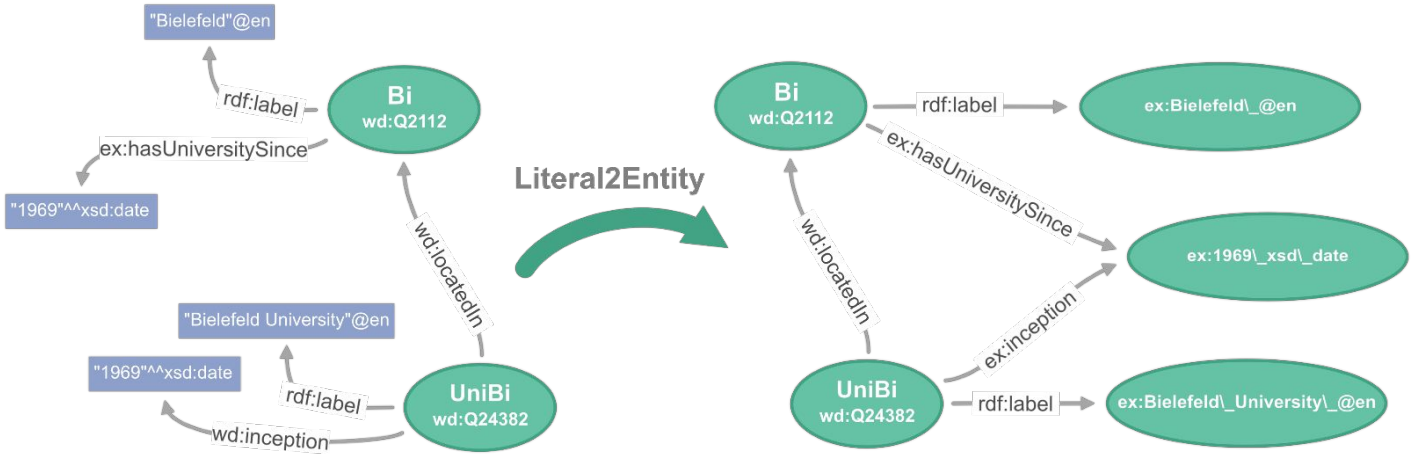
$$f \times G \rightarrow P + Literals \Rightarrow f' \times G \rightarrow P$$

- **Our approach:** represent literal triples as relational triples → all state-of-the-art Link Prediction models can be used without modification

$$f \times G \rightarrow P + Literals \Rightarrow f \times G' \rightarrow P$$

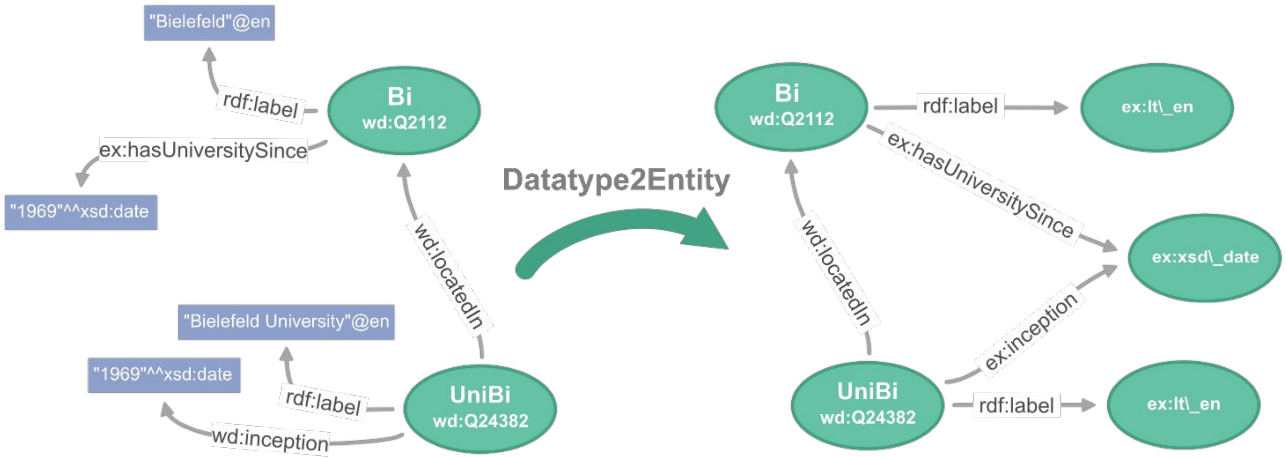
Literal2Entity

- transforms every literal into an entity, creating a new URI
- number of new entities $O(|G_L|)$
- number of new triples $O(|G_L|)$



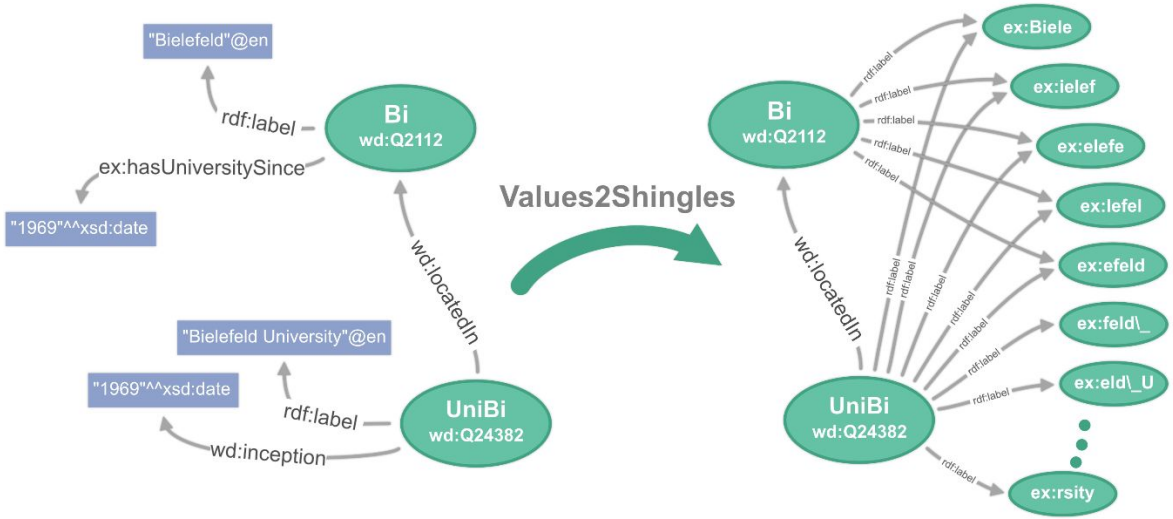
Datatype2Entity

- represent the literal's data type as an entity and set it into relation to the subject entity according to the attributive triple
- number of new entities $O(|D_G \cup T_G|)$
- number of new triples $O(|G_L|)$



Values2Shingles

- relies on the computation of k-shingles occurring in any textual literal, introducing a URI for each shingle and linking the corresponding subject entity to each of these shingle entities
- number of new entities $O((m-k)\cdot|G_L|)$
- number of new triples $O((m-k)\cdot|G_L|)$



Datasets - with literals

- FB15k, FB15k-237, and YAGO3-10 entities are enriched by literals present in the source graphs
- LitWD48K - specifically designed for the evaluation of Link Prediction with literals

Further hot research direction on KGC

- Temporal Link Prediction
- Explainability of Link Prediction Models
- Graph curation - e.g. which graph structure is relevant
- Global Knowledge Graph features
- Hybrid approaches

Conclusion

- A KG is a graph-structured data model that is used to store and structure information.
- Link Prediction: inferring new facts from those already in the KG
  - rule based: AMIE
  - embedding based: Geometric (TransE), Tensor Factorization (DistMult), Graph Neural Networks (R-GCN)
  - hybrid: combination of both
- most approaches do not take literal data into account → emerging field of research
- our approach: literal transformations