Statistical Inference on KGs

VU VRIJE UNIVERSITEIT AMSTERDAM

Knowledge bases on the Web

Motivation

- There are some serious challenges related to Knowledge Graphs
  - **Growth:** knowledge graphs are highly incomplete!
    - *Knowledge extraction:* extract new entities and relations
    - *Link prediction:* add (or guess) new relations between entities
  - **Validation:** knowledge graphs are not always correct!
    - *Entity resolution:* merge duplicate entities, split wrongly merged ones
    - *Error detection:* remove false assertions
  - **Interface:** how to make it easier to access knowledge?
    - *Semantic parsing:* interpret the meaning of queries
    - *Question answering:* computing answers using knowledge graphs
  - **Intelligence:** can AI emerge from knowledge graphs?
    - *Automatic reasoning* and planning
    - Generalization and abstraction

Motivation

- There are some serious challenges related to Knowledge Graphs
  - **Growth:** knowledge graphs are highly incomplete!
    - *Knowledge extraction:* extract new entities and relations
    - *Link prediction:* add (or guess) new relations between entities
  - **Validation:** knowledge graphs are not always correct!
    - *Entity resolution:* merge duplicate entities, split wrongly merged ones
    - *Error detection:* remove false assertions
  - **Interface:** how to make it easier to access knowledge?
    - *Semantic parsing:* interpret the meaning of queries
    - *Question answering:* computing answers using knowledge graphs
  - **Intelligence:** can AI emerge from knowledge graphs?
    - *Automatic reasoning* and planning
    - Generalization and abstraction

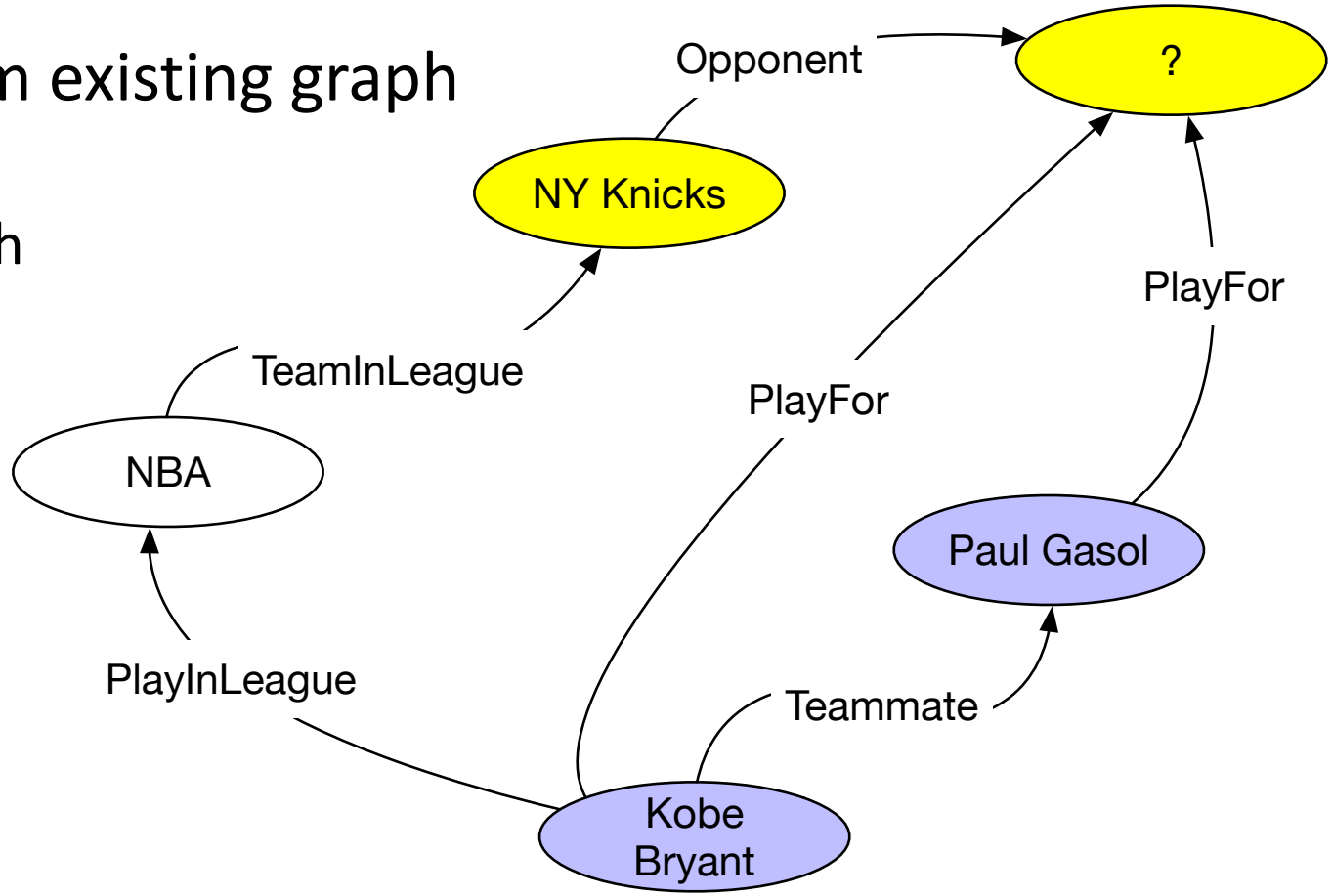
We talked about it in the previous lectures

Motivation

- There are some serious challenges related to Knowledge Graphs
  - **Growth:** knowledge graphs are highly incomplete!
    - *Knowledge extraction:* extract new entities and relations
    - *Link prediction:* add (or guess) new relations between entities
  - **Validation:** knowledge graphs are not always correct!
    - *Entity resolution:* merge duplicate entities, split wrongly merged ones
    - *Error detection:* remove false assertions
  - **Interface:** how to make it easier to access knowledge?
    - *Semantic parsing:* interpret the meaning of queries
    - *Question answering:* computing answers using knowledge graphs
  - **Intelligence:** can AI emerge from knowledge graphs?
    - *Automatic reasoning* and planning
    - Generalization and abstraction

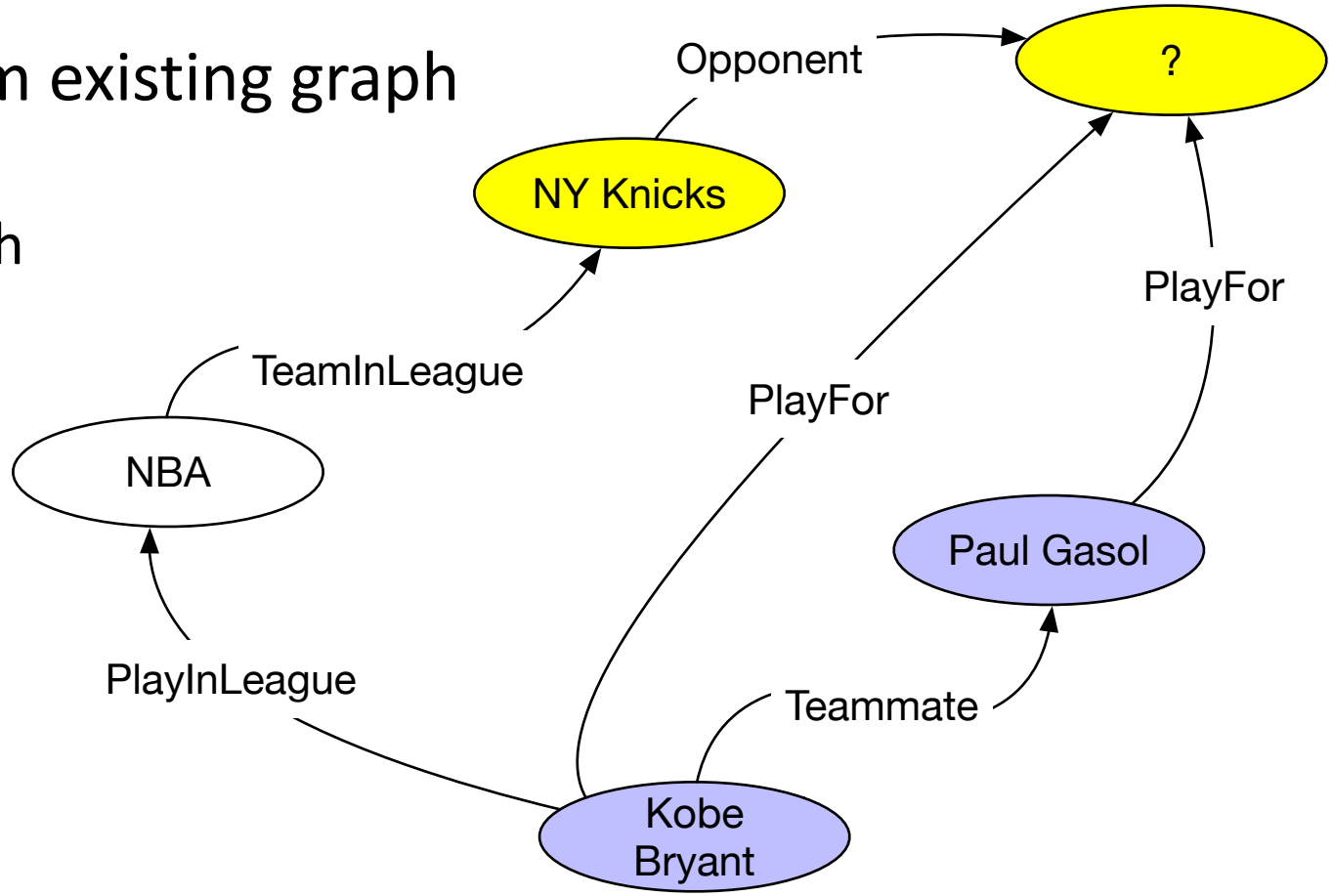
Link Prediction

- **Goal:** Add knowledge from existing graph
  - No external source
  - Reasoning within the graph
- **Some solutions**
  - Rule-based methods
  - Probabilistic models
  - Factorization models
  - Embedding models



Link Prediction

- **Goal:** Add knowledge from existing graph
  - No external source
  - Reasoning within the graph
- **Some solutions**
  - *Rule-based methods*
  - Probabilistic models
  - Factorization models
  - Embedding models



Link Prediction with Rules

- Goal: We want to make the predictions by applying rules that we extract from the graph => *rule mining*
- Rule mining is an important research topic in several communities
- **Association rule mining:** process of discovering rules that associate items together

$I = \{i_1, i_2, \ldots, i_m\}$ is a set of items
Transaction $t \subseteq I$ is a subset of items
Transaction database $T = \{ t_1, t_2, t_3, \ldots, t_n \}$. We assume *T* is large
An *association rule* is defined as $X \rightarrow Y \; where \; X, Y \subseteq I \; and \; X \cap Y = \emptyset$

- A famous algorithm for association mining is the *Apriori* algorithm [1]

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2014. Chapter 6.

Link Prediction with Rules

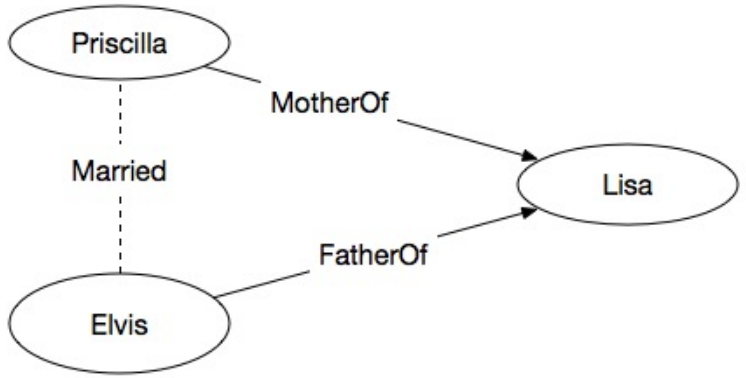
- **Association rule mining:** process of discovering rules that associate items together

In our case, we can assume that a transaction is a connected subgraph

$$t_i = \{mother(Priscilla, Lisa), father(Elvis, Lisa), married(Priscilla, Elvis)\}$$

If we observe that this transaction occurs *often,* then we can infer the rule

$$mother(x, y), father(z, y) \rightarrow married(x, z)$$



Link Prediction with Rules

- **Inductive Logic Programming (ILP):** field of machine learning that deduces logical rules from ground facts

Input:
- A background knowledge B (mostly a set of atoms, e.g. *married(obama,michelle)*)
- A set of positive examples E^+ (some facts that hold)
- A set of negative examples E^- (some facts that do not hold)

Goal:
Calculate an hypothesis *h* s.t. $B \wedge h$ entails E^+ and does not entail E^-

- Several ILP systems have been designed to extract rules from the Web, e.g. SHERLOCK [1] and DL-Learner [2]

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] S. Schoenmackers, O. Etzioni, D. S. Weld, and J. Davis, "Learning First-order Horn Clauses from Web Text," in EMNLP 2010, pp. 1088–1098.
[2] J. Lehmann, "DL-Learner: learning concepts in description logics," *Journal of Machine Learning Research*, vol. 10, no. Nov, pp. 2639–2642, 2009.

Link Prediction with Rules: AMIE

One big problem of rule association mining and ILP methods is that they either assume CWA or require negative samples

Knowledge Graphs operate under OWA: what is not known might still be true!

Link Prediction with Rules: AMIE

- **AMIE [1]** is a system where rules are extracted with a "rule-association" mining mindset
- Horn rules are constructed incrementally and in parallel
  - They start considering a rule with a single atom, e.g. *fatherOf(X,Y)*
  - Then add another atom which shares at least one variable, e.g. *motherOf(Z,Y)*
  - When the rule is closed (e.g. each variable appears in at least two atoms), it is evaluated using confidence: Only rules with high confidence are kept

$$conf(rule) = \frac{count(body \ \wedge \ head)}{count(body)}$$

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] L. A. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek, "AMIE: Association Rule Mining Under Incomplete Evidence in Ontological Knowledge Bases," in *WWW*, 2013, pp. 413–422.

Link Prediction with Rules: AMIE

- An important novelty of the AMIE system is the introduction of the notion of *Partial Completeness Assumption*

Definition: Let KG' be the graph which contain all true facts which are not known in KG. The assumption consists of assuming that if $r(x,y) \in KG$ then $\nexists\, y'.r(x,y') \in KG'$

Standard confidence

$$conf(\overrightarrow{B} \Rightarrow r(x,y)) := \frac{supp(\overrightarrow{B} \Rightarrow r(x,y))}{\#(x,y) : \exists z_1, ..., z_m : \overrightarrow{B}}$$

PCA-based confidence

$$conf_{pca}(\overrightarrow{B} \Rightarrow r(x,y)) := \frac{supp(\overrightarrow{B} \Rightarrow r(x,y))}{\#(x,y) : \exists z_1, ..., z_m, y' : \overrightarrow{B} \wedge r(x,y')}$$

VU VRIJE UNIVERSITEIT AMSTERDAM

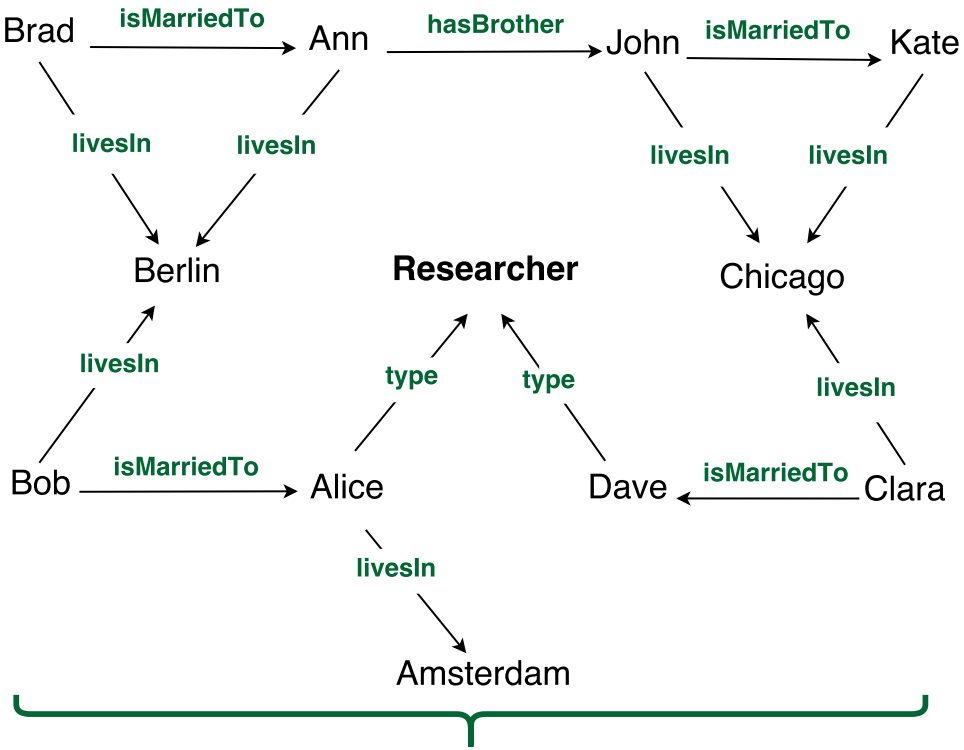
Evaluation

System	Top n	Predictions	Precision
ALEPH	7	2997	27%
AMIE	13	3180	66%
ALEPH	9	5031	26%
AMIE	29	5003	47%
ALEPH	17	8457	30%
AMIE	52	8686	45%

	Top 20 rules	Top 30 rules	All rules
Confidence	0.76	0.63	0.33
PCA Confidence	0.32	0.29	0.29

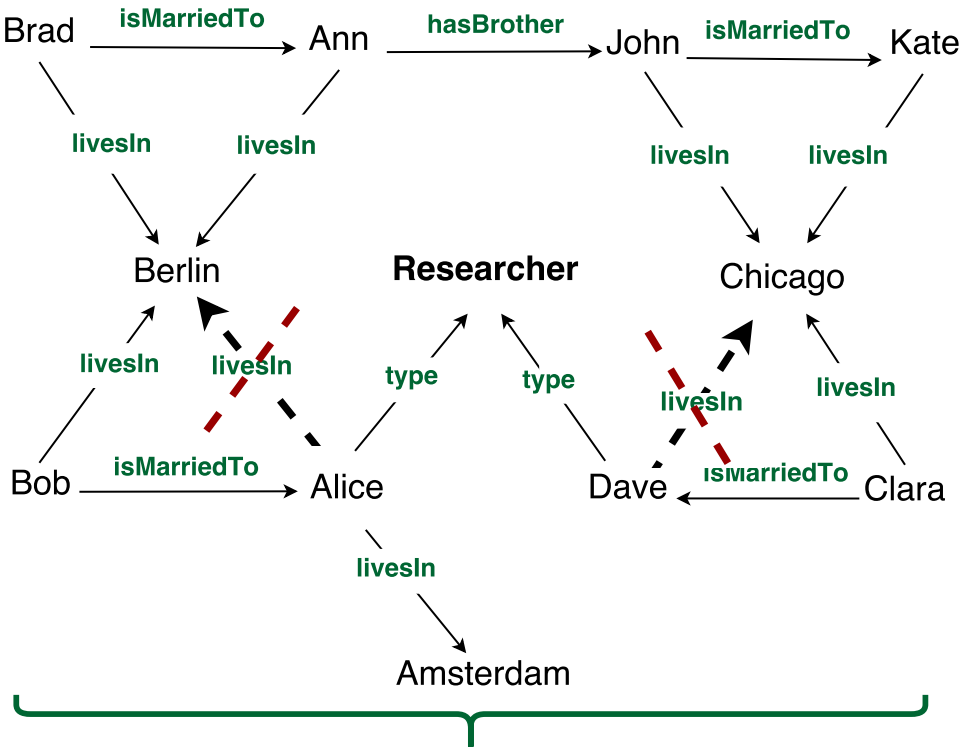
$$isCitizenOf(x, y) \Rightarrow livesIn(x, y)$$
$$hasAdvisor(x, y) \wedge graduatedFrom(x, z) \Rightarrow worksAt(y, z)$$
$$wasBornIn(x, y) \wedge isLocatedIn(y, z) \Rightarrow isCitizenOf(x, z)$$
$$hasWonPrize(x, G.\ W.\ Leibniz) \Rightarrow livesIn(x, Germany)$$

Link Prediction with Rules: Non-monotonic Learning



$$r: \; livesIn(X, Z) \leftarrow isMarriedTo(Y, X), livesIn(Y, Z)$$

Link Prediction with Rules: Non-monotonic Learning



$$r: \; livesIn(X, Z) \leftarrow isMarriedTo(Y, X), livesIn(Y, Z), not \; researcher(X)$$

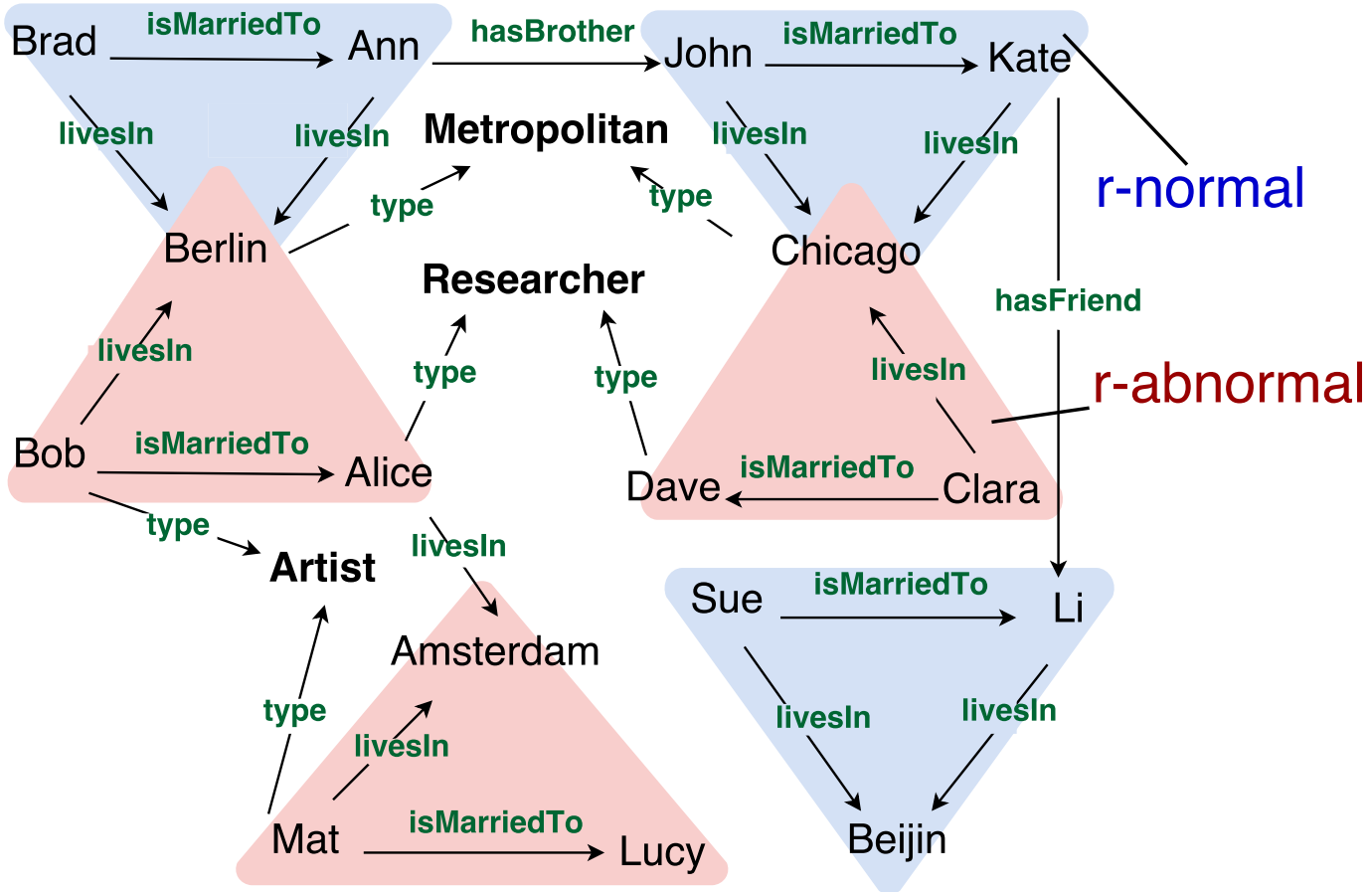
Link Prediction with Rules: Non-monotonic Learning

- We developed a technique to mine non-monotonic rules [1].
- Focus on **unary** rules (binary rules are converted with *propositionalization*)
- **Step 1.** Mine association Horn clauses from the KG. We can use FPGrowth, an off-the-shelf popular algorithm
- **Step 2.** Determine normal and abnormal substitutions for every rule (see next slide)
- **Step 3.** Find all exception candidates (=predicates for unary rules) for every rule
- **Step 4.** Rank exception candidates and select the best ones

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] M. H. Gad-Elrab, D. Stepanova, J. Urbani, and G. Weikum, "Exception-Enriched Rule Learning from Knowledge Graphs," in *ISWC*, 2016, pp. 234–251.

Abnormal substitutions

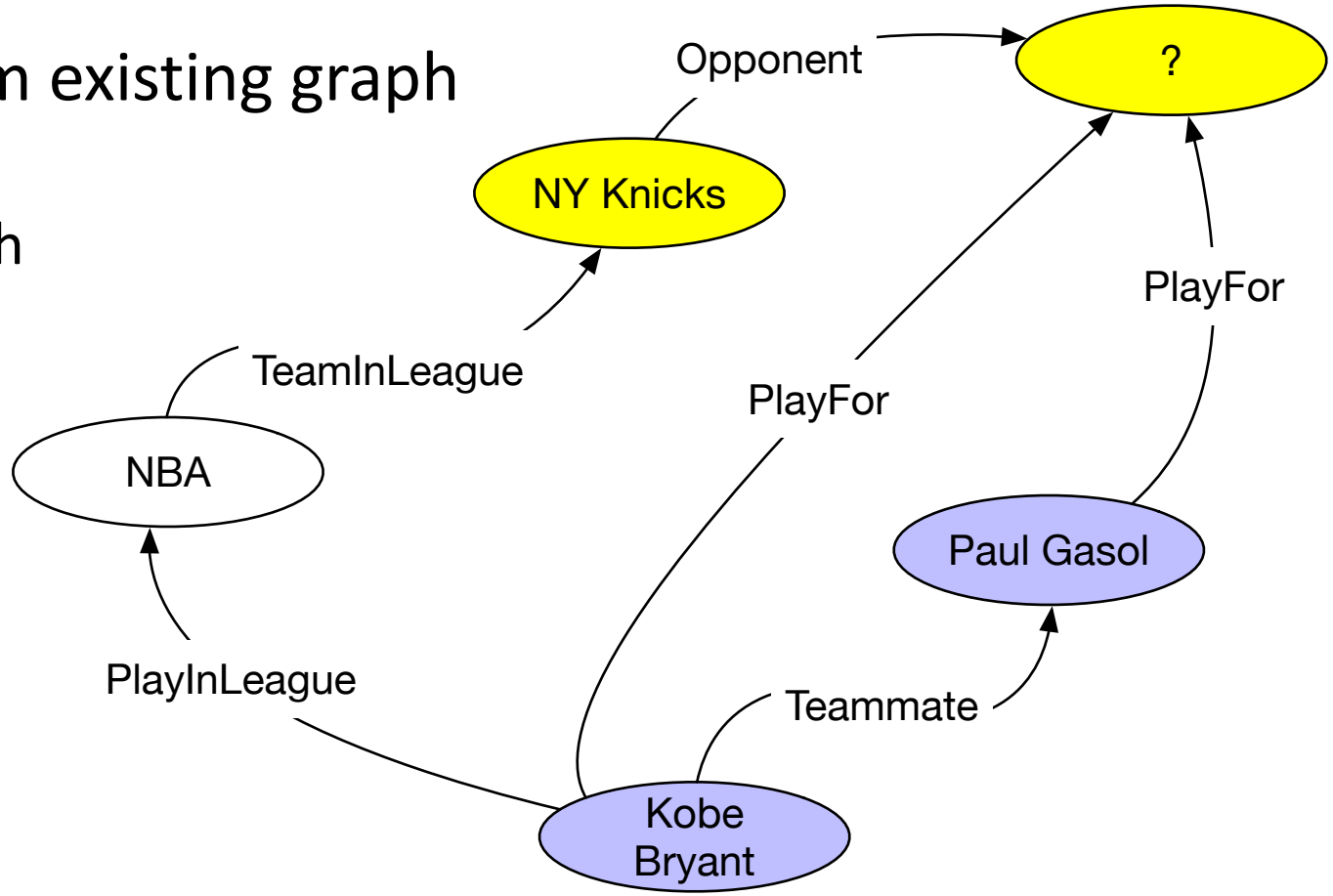


$r: \ livesIn(X, Z) \leftarrow isMarriedTo(Y, X), livesIn(Y, Z)$

VU VRIJE UNIVERSITEIT AMSTERDAM

Link Prediction

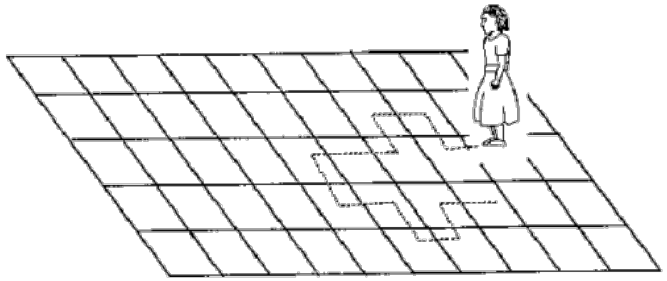
- **Goal:** Add knowledge from existing graph
  - No external source
  - Reasoning within the graph
- **Some solutions**
  - Rule-based methods
  - *Probabilistic models*
  - Factorization models
  - Embedding models



Link Prediction with Random Walks

- Rules are expensive to calculate and are “brittle”. [1] proposes a different approach, called Path Ranking Algorithm (PRA), which is a process based on *random walks*

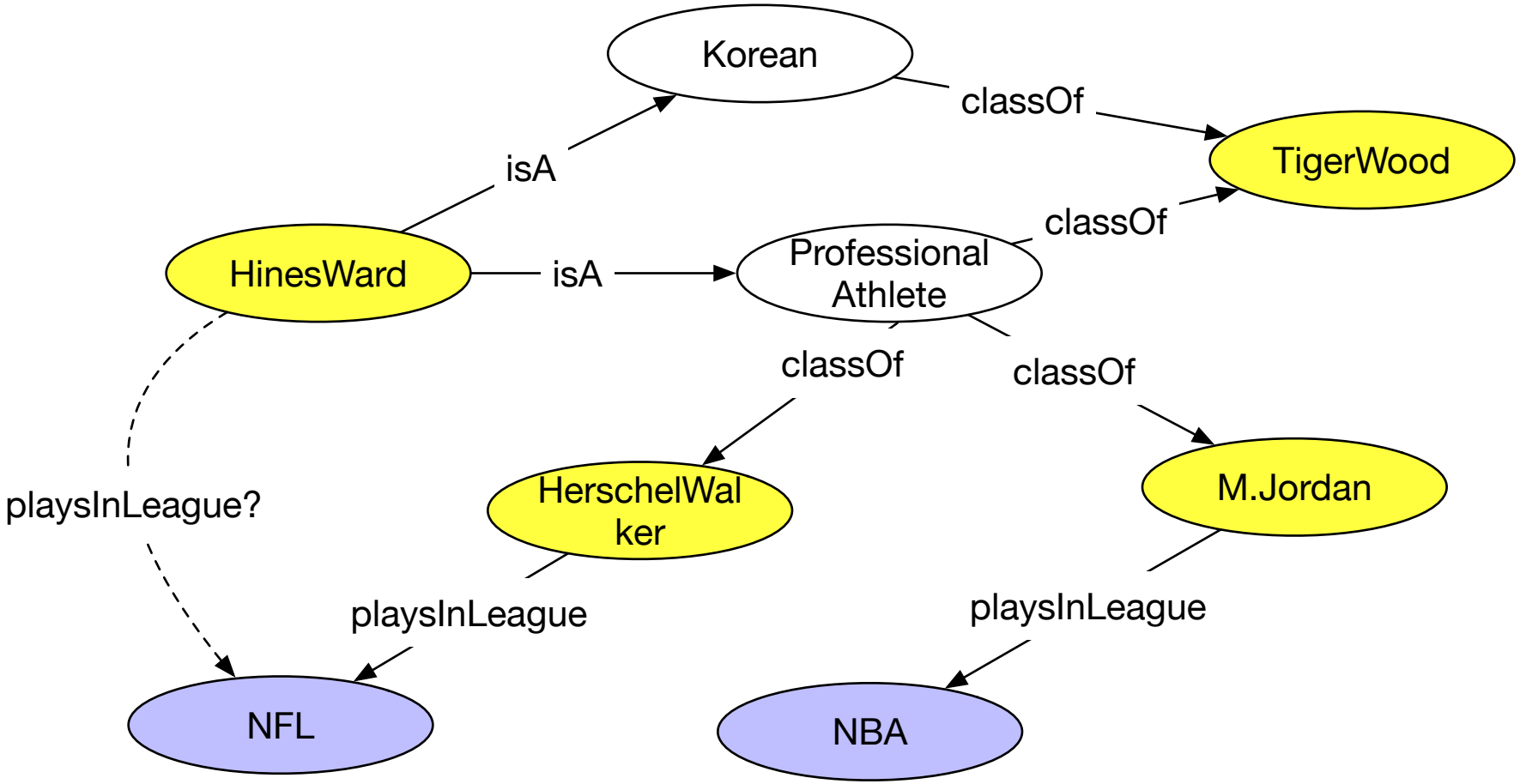
Definition: Given a graph and a starting point, we select a neighbor of it at random, and move to this neighbor; then we select a neighbor of this point at random, and move to it etc. The (random) sequence of points selected this way is a random walk on the graph. [2]



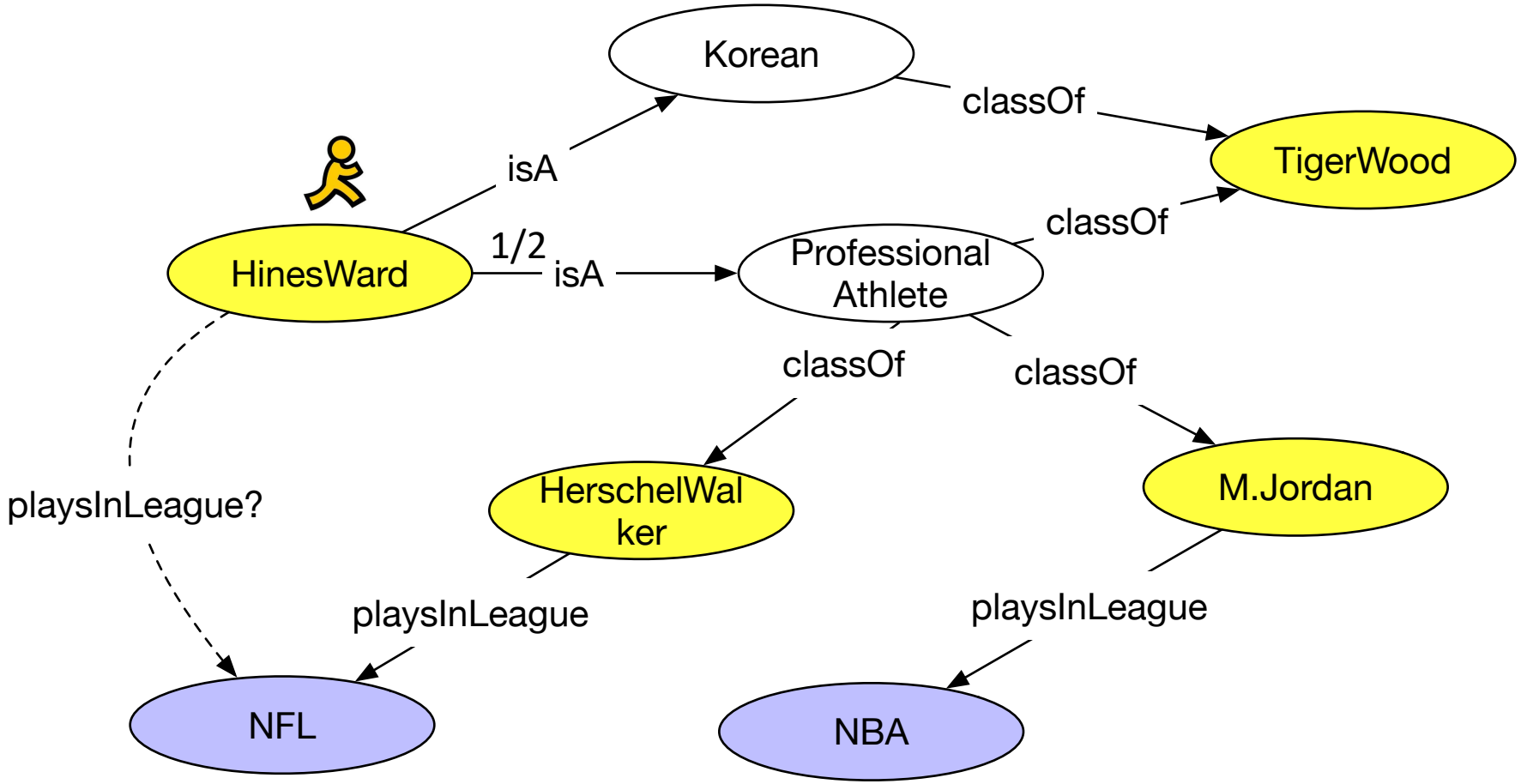
VU VRIJE UNIVERSITEIT AMSTERDAM

[1] N. Lao, T. Mitchell, and W. W. Cohen, “Random walk inference and learning in a large scale knowledge base,” in *EMNLP*, 2011, pp. 529–539.
[2] L. Lovász, “Random walks on graphs: A survey,” *Combinatorics, Paul erdos is eighty*, vol. 2, no. 1, pp. 1–46, 1993.

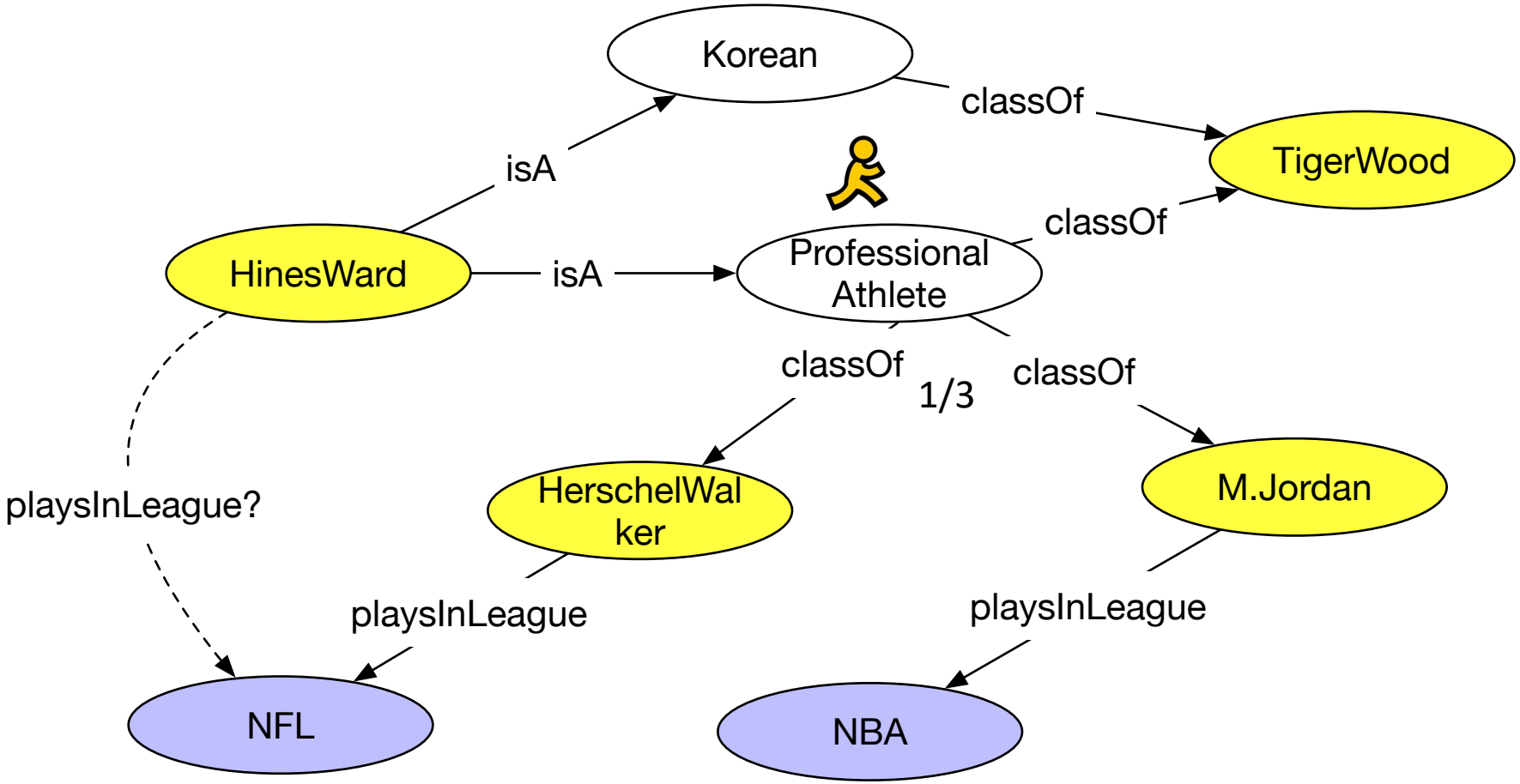
Link Prediction with Random Walks



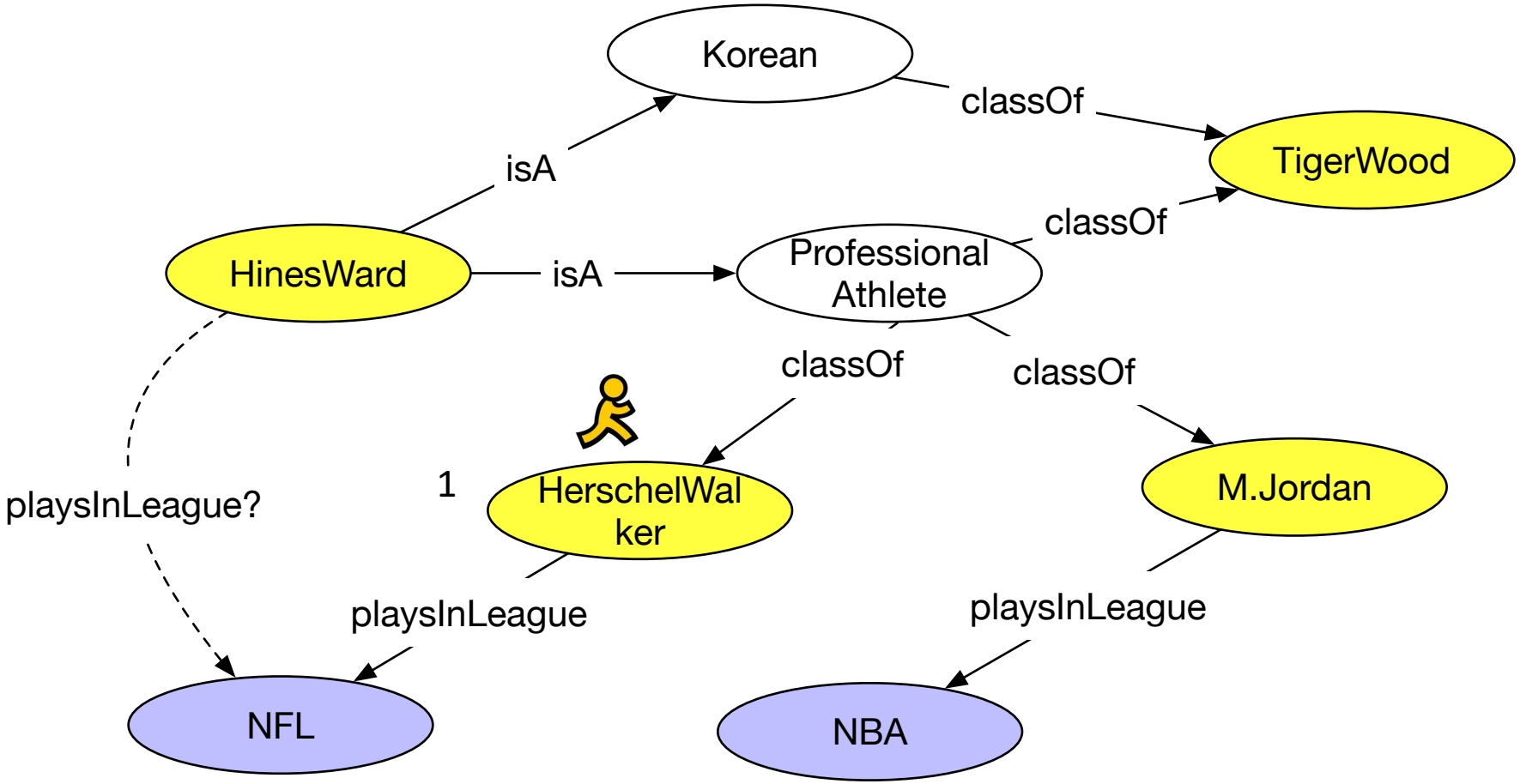
Link Prediction with Random Walks



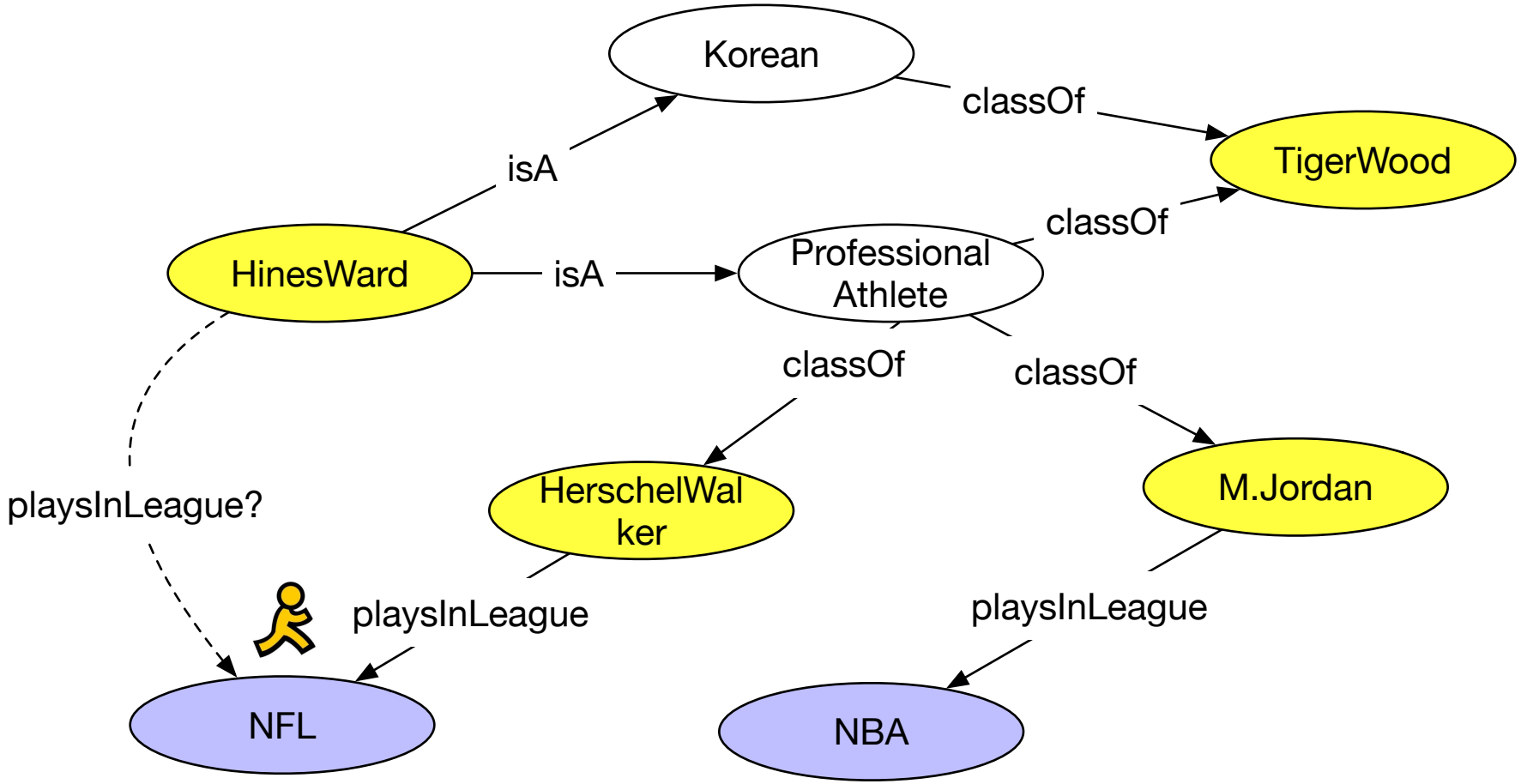
Link Prediction with Random Walks



Link Prediction with Random Walks



Link Prediction with Random Walks



What's the probability that Hines Ward plays in NFL? 1/2 * 1/3

VU VRIJE UNIVERSITEIT AMSTERDAM

Link Prediction with Random Walks

- Instead of doing random walks, we can constrain them following the links established by some rules we can mine from the graph

$$isA(x_1, c) \wedge classOf(c, x_2) \wedge playsInLeague(x_2, l) \rightarrow playsInLeague(x_1, l)$$

- We can group the rules by the predicate in the head (e.g. *playInLeague)*, and perform many random walks to estimate the probabilities with some training triples

Link Prediction with Random Walks

- Suppose we have *n* rules with *playsInLeague* has head. We can construct a feature vector of *n* elements and train a linear regression model with the objective function

$$score(e; s) = \sum_{P \in \boldsymbol{P}} h_{s,P}(e)\Theta_P$$ where $h_{s,P}(e)$represents the probability of reaching *e* from *s* following the path *p*

- The algorithm constructs one model per relation

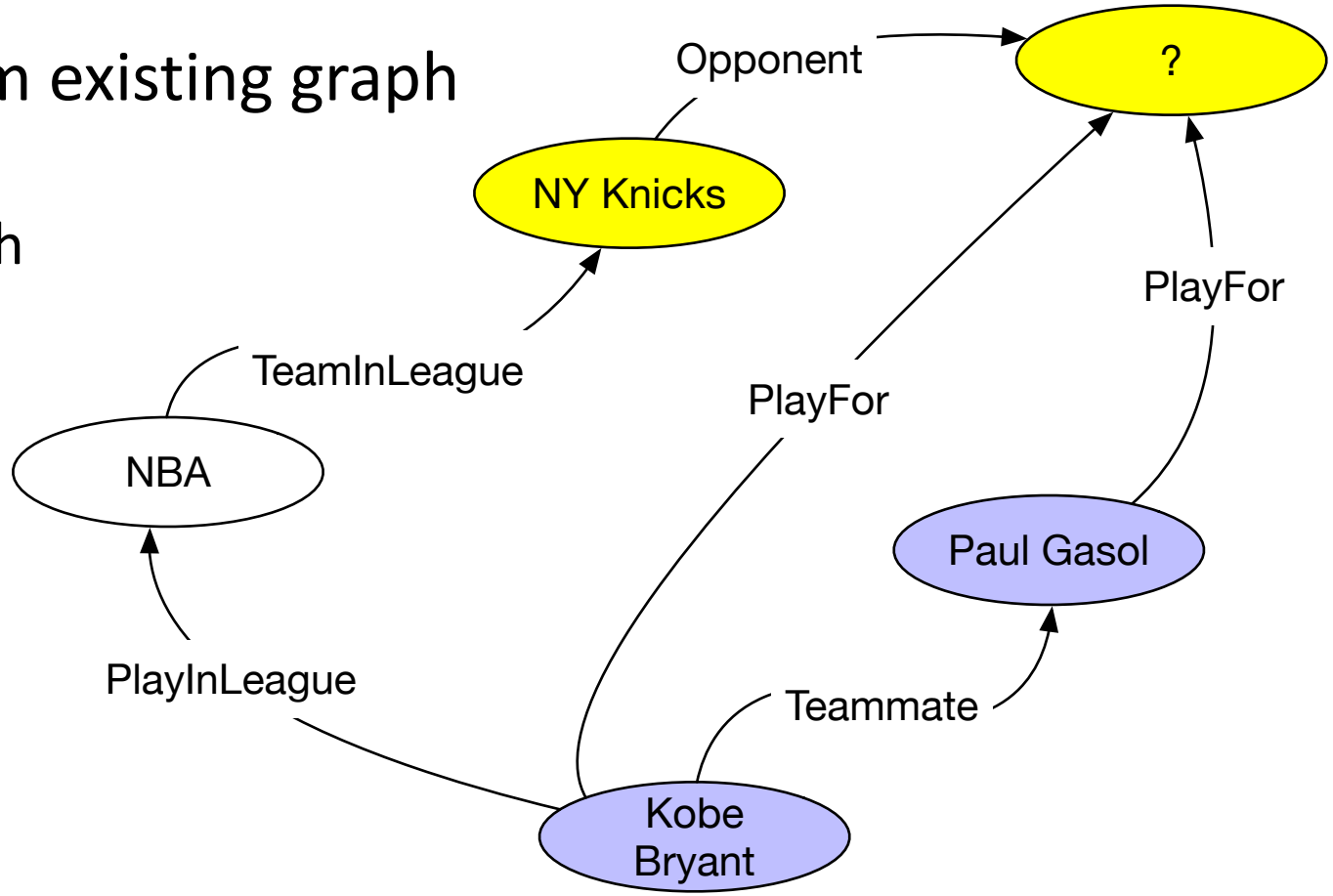
Link Prediction with Random Walks

Evaluation

- Used NELL KG
- Relied on human assessment (5 humans, 75% agreement) on sample queries of the type R(x,?)
- If we look at the top-10 results, the method performs as good as a N-FOIL -- an inductive baseline
- If we move to the top-100 the results get significantly better

Link Prediction

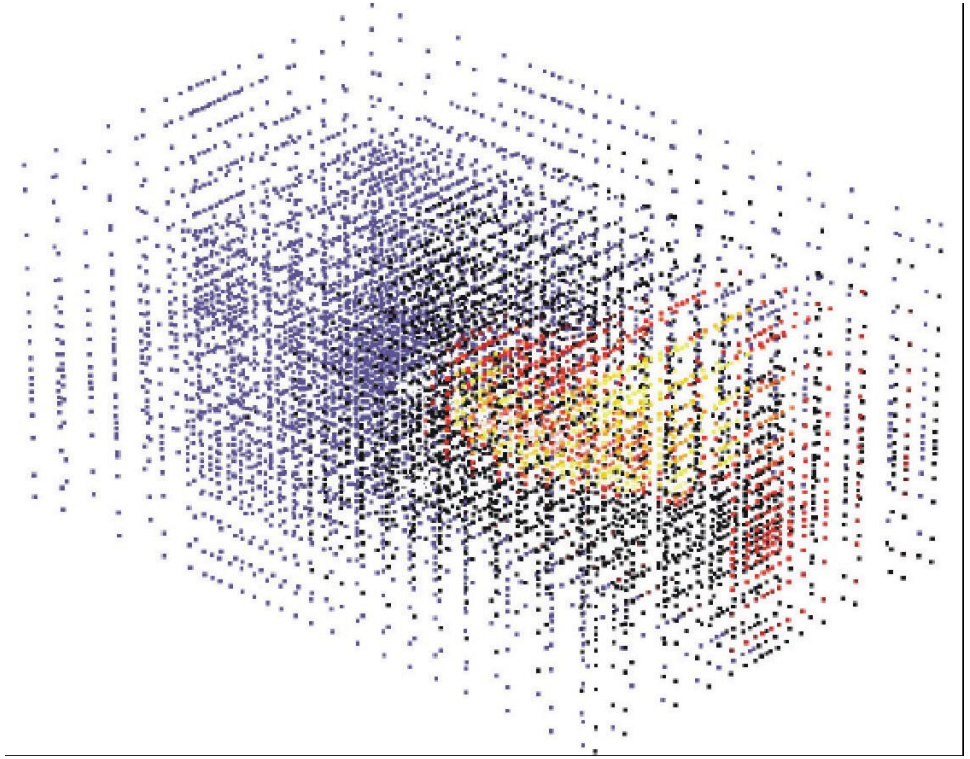
- **Goal:** Add knowledge from existing graph
  - No external source
  - Reasoning within the graph
- **Some solutions**
  - Rule-based methods
  - Probabilistic models
  - *Factorization models*
  - Embedding models



Link Prediction with Tensors

- **Definition:**
  - **1**-dimension array: vector
  - **2**-dimension array: matrix
  - **3 or more-dimensions** array: tensor

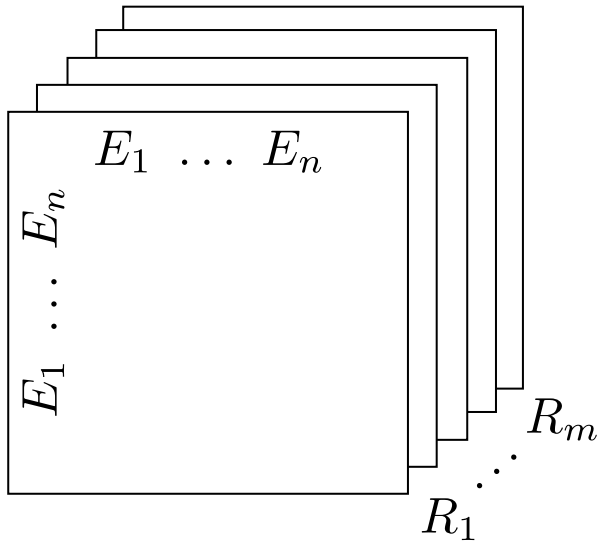
Multiple binary relations can be expressed naturally as a three-way tensor



VU VRIJE UNIVERSITEIT AMSTERDAM

Link Prediction with Tensors

- Tensors are well-studied data structures in Machine Learning [1]
- Suppose our knowledge graph has *m* relations and *n* entities
- We can assign two *modes (=ways)* to the entities and one *mode* to the relations
- The entries of the tensor are 1 when a relation between two entities exists. Otherwise, the value is 0



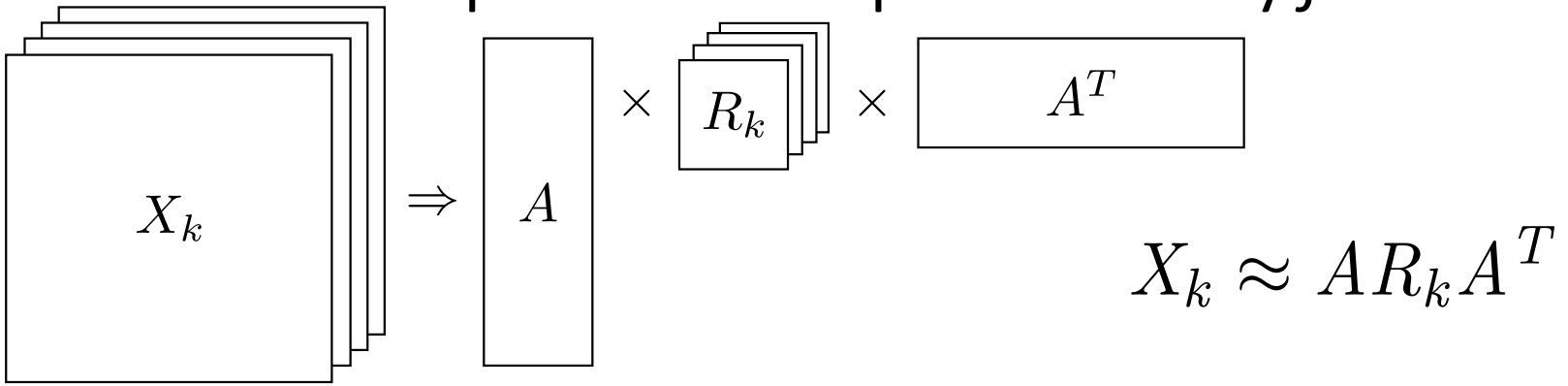
- E.g. (2,3,1) = 1 if $R_1(e_2,e_3)$ exists in the KG

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] T. Kolda and B. Bader, "Tensor Decompositions and Applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009.

Link Prediction with Tensors

- **RESCAL [1]** is a method that performs link predictions by *factorizing* the tensor



$$X_k \approx A R_k A^T$$

- **Let** *n* be the number of entities and *r* the number of dimensions
- **A** is a *nxr* matrix that represents the *global* entity-latent-components
- $\mathbf{R_k}$ is a asymmetric *rxr* matrix that specifies the interaction of the latent components per predicate

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] M. Nickel, V. Tresp, and H.-P. Kriegel, "A Three-Way Model for Collective Learning on Multi-Relational Data," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ACM, 2011, pp. 809–816.

Link Prediction with Tensors

The factorization is translated as an optimization problem

$$\min_{A,R_k} loss(A, R_k) + reg(A, R_k)$$

where *loss* is the loss function

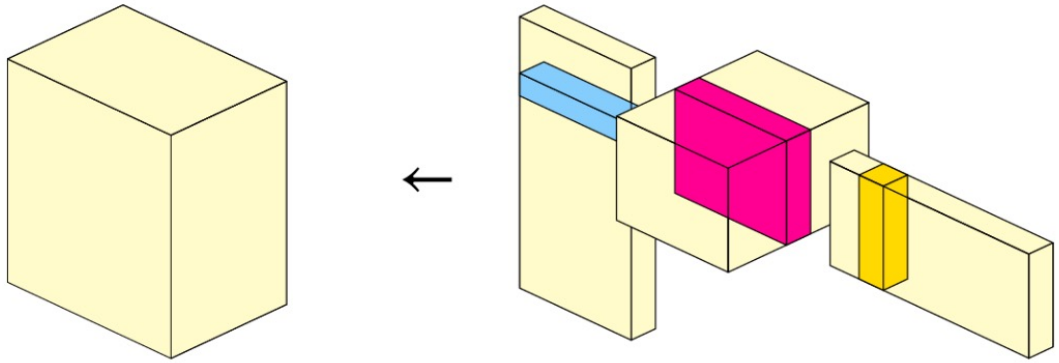
$$loss(A, R_k) = \frac{1}{2} \sum_k \|\mathcal{X}_k - A R_k A^T\|_F^2$$

and *reg* is the regularization function

$$reg(A, R_k) = \frac{1}{2} \lambda \left(\|A\|_F^2 + \sum_k \|R_k\|_F^2 \right)$$

The calculation is performed using the Alternating-least Square algorithm (ALS)

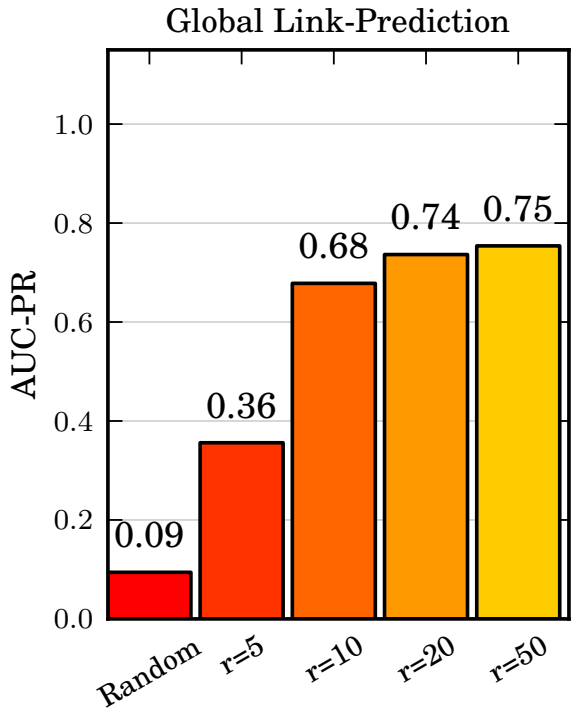
Link Prediction with Tensors



- **Basic procedure** of link prediction with RESCAL
  - **First,** we compute the "rank-reduced" factorization
  - **Second,** entries after reconstruction (i.e. $\hat{x}_{ijk} = \vec{a}_i^T R_k \vec{a}_j$) can be regarded as *confidence values* of the model that the corresponding triple exists

Factorizing YAGO

- As an experiment, the authors of [1] took a subset of YAGO of 2.6M entities and 64M facts
- Type of queries: p(?,o) over 38 relations



[1] M. Nickel, V. Tresp, and H.-P. Kriegel, “Factorizing YAGO: Scalable Machine Learning for Linked Data,” in *WWW*, 2012, pp. 271–280.

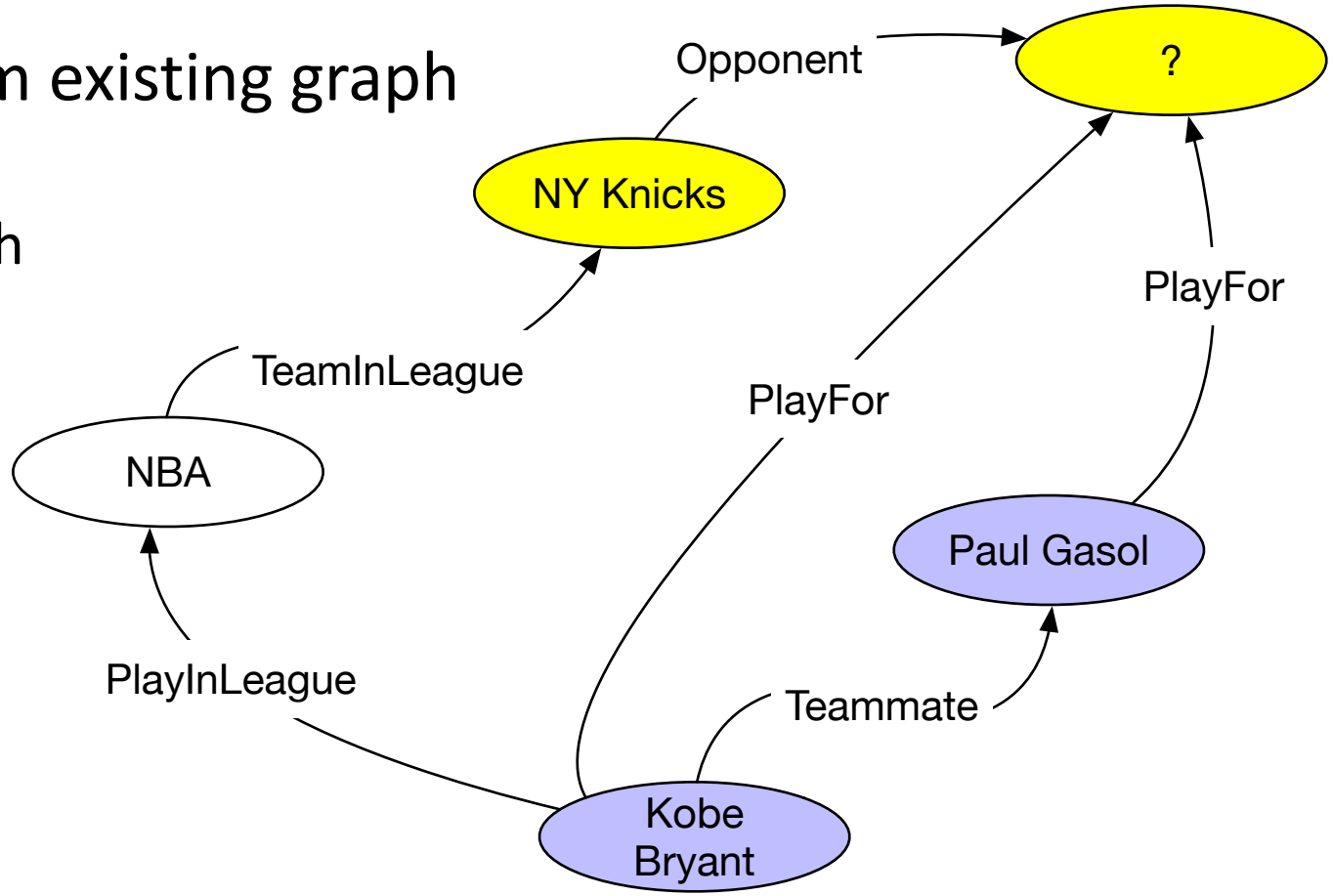
VU VRIJE UNIVERSITEIT AMSTERDAM

Link Prediction with Tensors

- Collective learning is performed via the entities' latent-component representation
- **A** is an matrix that contains **embeddings** of the entities. We can use in a similar way as for word embeddings (word2vec)
- A major problem of RESCAL is that it has to learn a very large number of parameters (esp. R). Not so cheap to compute

Link Prediction

- **Goal:** Add knowledge from existing graph
  - No external source
  - Reasoning within the graph
- **Some solutions**
  - Rule-based methods
  - Probabilistic models
  - Factorization models
  - *Embedding models*



Latent-features model

- *RESCAL* is a latent-features model. Predictions are conditionally independent w.r.t. a given set of global latent features [1]
- Number of parameters in the RESCAL model = $rd^2 + nd$ where *d* = # dimensions, *n* = # entities, *r* = # relations
- Can we lower the number of parameters?

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A Review of Relational Machine Learning for Knowledge Graphs," *arXiv:1503.00759* 2015

Latent distance models

- Latent distance models are a subclass of latent feature models where the likelihood of links depends on the distance between the latent representation of the entities
- Consider one relation (e.g. *friendsOf*) and two entities (e.g. Mark and John)
  - $friendsOf(Mark, John) = d(\boldsymbol{e}_{mark}, \boldsymbol{e}_{john})$ where *d* is a arbitrary distance measure (e.g. Euclidean distance) [1]
- See the connection with word2vec?

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *Journal of the american Statistical association*, vol. 97, no. 460, pp. 1090–1098, 2002.

TransE

- **TransE** [1] is a latent distance model with less parameters than RESCAL
- Every entity and relation get a vector of *d* elements.
- The distance in TransE is: $d(\boldsymbol{e}_i + \boldsymbol{r}_k, \boldsymbol{e}_j)$ which reads as: "given an entity and a relation, what is the closest vertex to their sum?
- The model has $rd + nd$ parameters, much less than RESCAL

[1] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating Embeddings for Modeling Multi-relational Data," in *NIPS* 2013.

TransE

Overview training

1. Create a vector of *d* elements for each entity $e \in E$ and $r \in R$
2. Take as input a set of triples $(h, l, t) \in S$ (*S* is our knowledge graph)
3. Minimize the function with stochastic gradient descent)

$$\mathcal{L} = \sum_{(h,\ell,t)\in S} \sum_{(h',\ell,t')\in S'_{(h,\ell,t)}} \left[\gamma + d(\boldsymbol{h}+\boldsymbol{\ell},\boldsymbol{t}) - d(\boldsymbol{h'}+\boldsymbol{\ell},\boldsymbol{t'})\right]_+$$

where

$$S'_{(h,\ell,t)} = \left\{(h',\ell,t)|h' \in E\right\} \cup \left\{(h,\ell,t')|t' \in E\right\}$$

TransE

Overview testing

1. Given an "unseen" triple $(h, l, t) \in S'$, we calculate $d(\boldsymbol{h} + \boldsymbol{l}, \boldsymbol{e}_v) for\ \ all\ \forall v \in E$ and rank the results
2. Then, we check the average position of *t* (mean rank) or whether *t* occurs in the top *k* positions (hits@k).
3. We can look at the position of t w.r.t. all other entities (*raw position)* or remove all the "answers" that are already explicit in the KB (*filtered position*)

TransE

- **Evaluation**

Dataset	WN				FB15k				FB1M	
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)		Mean Rank	Hits@10 (%)
Eval. setting	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Filt.*	*Raw*	*Raw*
Unstructured [2]	315	304	35.3	38.2	1,074	979	4.5	6.3	15,139	2.9
RESCAL [11]	1,180	1,163	37.2	52.8	828	683	28.4	44.1	-	-
SE [3]	1,011	985	68.5	80.5	273	162	28.8	39.8	22,044	17.5
SME(linear) [2]	545	533	65.1	74.1	274	154	30.7	40.8	-	-
SME(bilinear) [2]	526	509	54.7	61.3	284	158	31.3	41.3	-	-
LFM [6]	469	456	71.4	81.6	283	164	26.0	33.1	-	-
TransE	**263**	**251**	**75.4**	**89.2**	**243**	**125**	**34.9**	**47.1**	**14,615**	**34.0**

VU VRIJE UNIVERSITEIT AMSTERDAM

TransE

Open problems: Negation

- In TransE, we create random triples to use as evidence
- However, the OWA does not allow to make such assumption
- Some approaches proposed to leverage the schema to create negative examples:
  - *Functional properties* => if <a> <bornIn> <b>, then <a> <bornIn> <c> where b!=c is surely false
  - Class hierarchies => if <p> has a range C, then we can pick an instance of a class D!=C to construct negative samples
- These approaches introduce biases and do not work well because some classes are too popular and will get only few negative examples

TransE

Open problems: Data sparsity

- Some relations appear only a few times to be properly trained.
- This is an open problem. There is not yet an effective solution for that.

TransE

Open problems: Lack of established evaluation methodology

- There is not a single metric that can be representative for the overall performance of the method
- Datasets are too different from each others. TransE returns very different results depending on the used KG
  - Freebase 74% (hit@10)
  - Wordnet 94%
  - YAGO 22%
- The latest research is focusing on trying to replicate reasoning (logical rules) using these models!

TransE

Open problems: Scalability

- Hogwild! [1] is a well-know technique to parallelize the learning of *sparse* models

Algorithm 1 HOGWILD! update for individual processors

1: **loop**
2: Sample e uniformly at random from E
3: Read current state x_e and evaluate $G_e(x_e)$
4: **for** $v \in e$ **do** $x_v \leftarrow x_v - \gamma G_{ev}(x_e)$
5: **end loop**

E: training set
x_e: parameters that relate to *e*
G: gradient of the loss function on *e*

VU VRIJE UNIVERSITEIT AMSTERDAM

[1] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2011, pp. 693–701.

TransE

Open problems: Scalability

- Hogwild! [1] is a well-know technique to parallelize the learning of *sparse* models.
- It can be trivially expanded to an arbitrary number of threads. It works only if the problem is *sparse*
- Are KG embeddings an example of a sparse problem? Preliminary evaluation indicate it is