# Random Walk Inference and Learning in A Large Scale Knowledge Base

Anshul Bawa

Adapted from slides by :

Ni Lao, Tom Mitchell, William W. Cohen

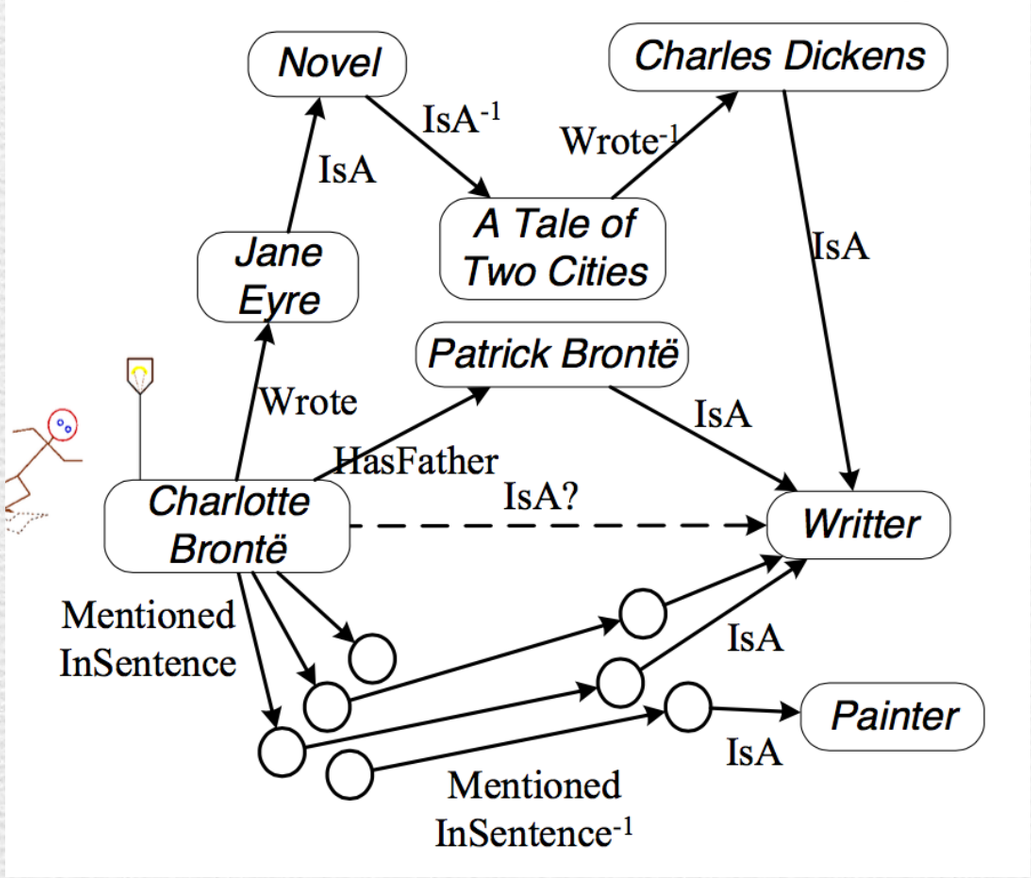
6 March 2017

# Outline

- Inference in Knowledge Bases
- The NELL project and N-FOIL
- Random Walk Inference : PRA
- Task formulation
- Heuristics and sampling
- Evaluation
- Class discussion

# Challenges to Inference in KBs

- Traditional logical inference methods too brittle - Robustness
- Probabilistic inference methods not scalable - Scalability



# NELL

Combines multiple strategies:

- morphological patterns
- textual context
- html patterns
- logical inference

| Predicate | Instance |
|---|---|
| cityInState | (troy, Michigan) |
| musicArtistGenre | (Nirvana, Grunge) |
| tvStationInCity | (WLS-TV, Chicago) |
| sportUsesEquip | (soccer, balls) |
| athleteInLeague | (Dan Fouts, NFL) |
| starredIn | (Will Smith, Seven Pounds) |
| productType | (Acrobat Reader, FILE) |
| athletePlaysSport | (scott shields, baseball) |
| cityInCountry | (Dublin Airport, Ireland) |

Half million confident beliefs

Several million candidate beliefs

# Horn Clause Inference

N-FOIL algorithm :

- start with a general rule
- progressively specialize it
- learn a clause
- remove examples covered

$$
\begin{aligned}
& AthletePlaysForTeam(x, y) \\
\wedge \quad & TeamPlaysInLeague(y, z) \\
\Rightarrow \quad & AthletePlaysInLeague(x,z)
\end{aligned}
$$

Computationally expensive

# Horn Clause Inference

Assumptions :

- Functional predicates only : No need for negative examples
- Relational pathfinding : Only clauses from bounded paths of binary relations

Small no. (~600) of high precision rules

# Horn Clause Inference

Issues :

- Still costly : N-FOIL takes days on NELL
- Combination by disjunction only : cannot leverage low-accuracy rules

# Horn Clause Inference

Issues :

- Still costly : N-FOIL takes days on NELL
- Combination by disjunction only : cannot leverage low-accuracy rules
- High precision but low recall

$$c \xrightarrow{teamAlsoKnownAs} c \xrightarrow{teamPlaysSport} c$$

$$c \xrightarrow{teamHomeStadium} c \xrightarrow{stadiumHomeToSport} c$$

$$c \xrightarrow{teamMember} c \xrightarrow{athletePlaysSport} c$$

$$c \xrightarrow{teamPlaysAgainstTeam} c \xrightarrow{teamPlaysSport} c$$

# Random Walks Inference

Labeled, directed graph

- each entity x is a node
- each binary relation R(x,y) is an edge labeled R between x and y
- unary concepts C(x) are represented as edge labeled “isa” between the node for x and a node for the concept C

# Random Walks Inference

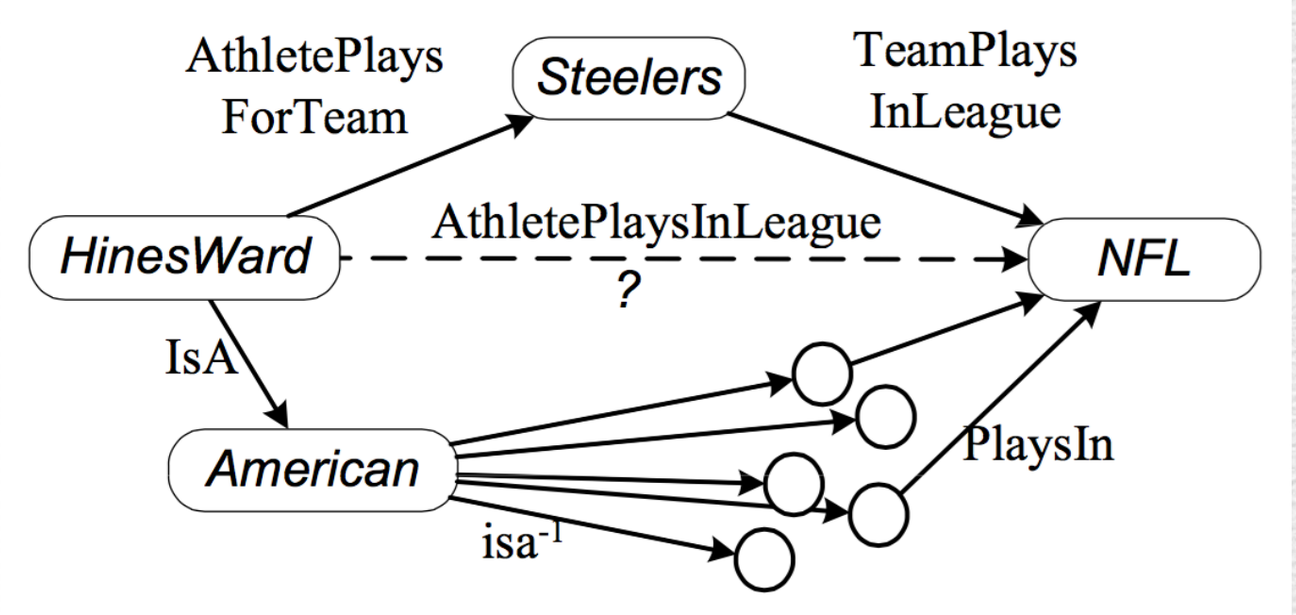
Labeled, directed graph

each entity x is a node

each binary relation R(x,y) is an edge labeled R between x and y

unary concepts C(x) are represented as edge labeled “isa” between the node for x and a node for the concept C

Given a node x and relation R, give ranked list of y

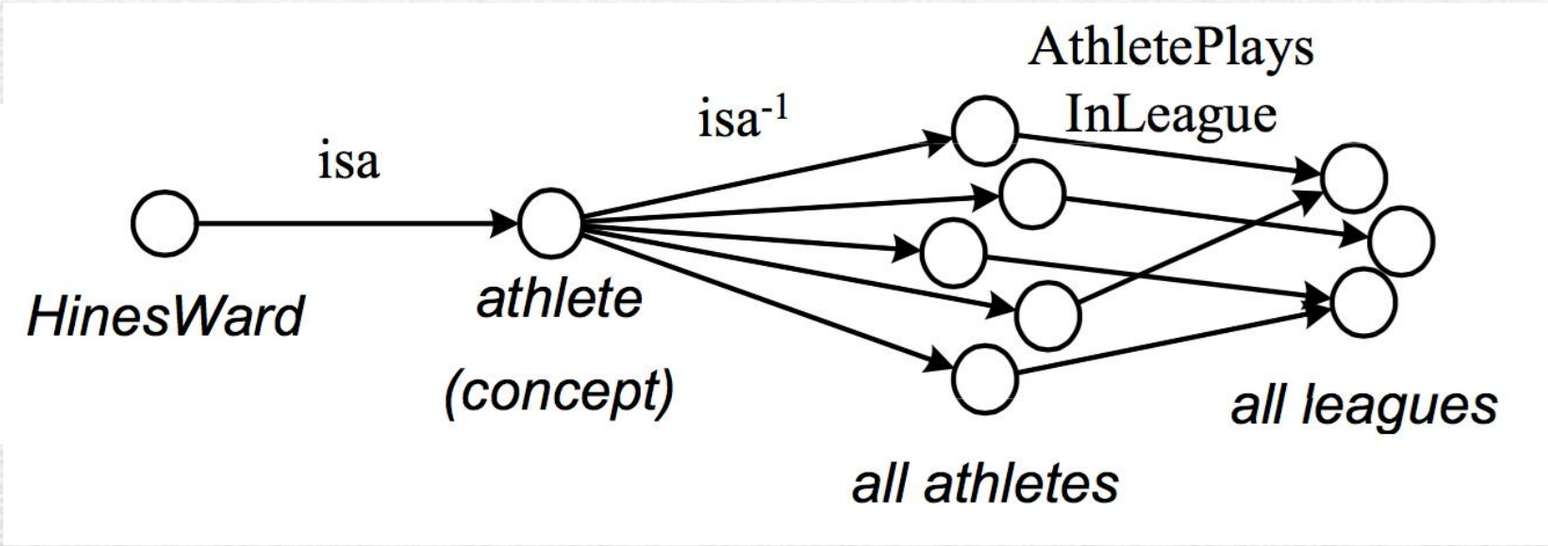


# Random Walks Inference : PRA

Logistic Regression over a large set of experts

Each expert is a bounded-length sequence of edge-labeled path types

$$
\begin{aligned}
& isa(x, c) \wedge isa^{-1}(c, x') \\
\wedge \quad & AthletePlaysInLeague(x', y) \\
\Rightarrow \quad & AthletePlaysInLeague(x, y)
\end{aligned}
$$



# Random Walks Inference : PRA

Logistic Regression over a large set of experts

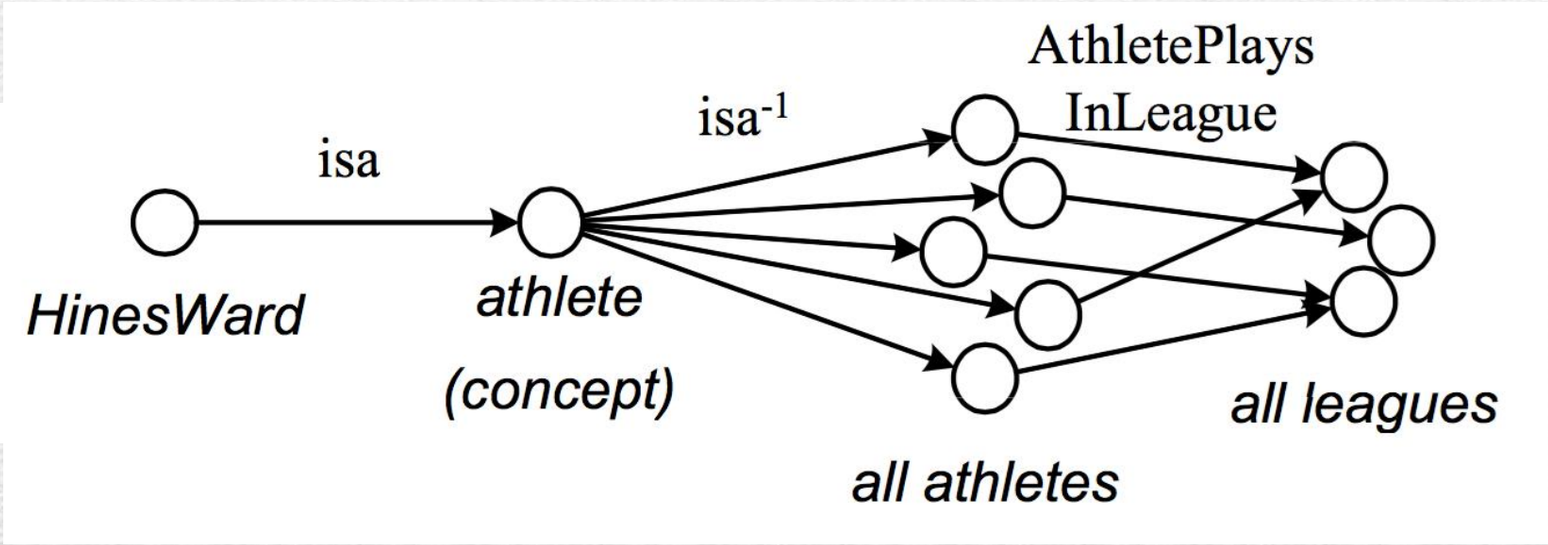
Each expert is a bounded-length sequence of edge-labeled path types

Expert scores are relational features

Score(y) = $|A_y| / |A|$

Many such low-precision high-recall experts

$$
\begin{aligned}
& isa(x, c) \wedge isa^{-1}(c, x') \\
\wedge \quad & AthletePlaysInLeague(x', y) \\
\Rightarrow \quad & AthletePlaysInLeague(x, y)
\end{aligned}
$$



# Random Walks Inference : PRA

+ Nupur

Logistic Regression over a large set of experts

Each expert is a bounded-length sequence of edge-labeled path types

+ Prachi

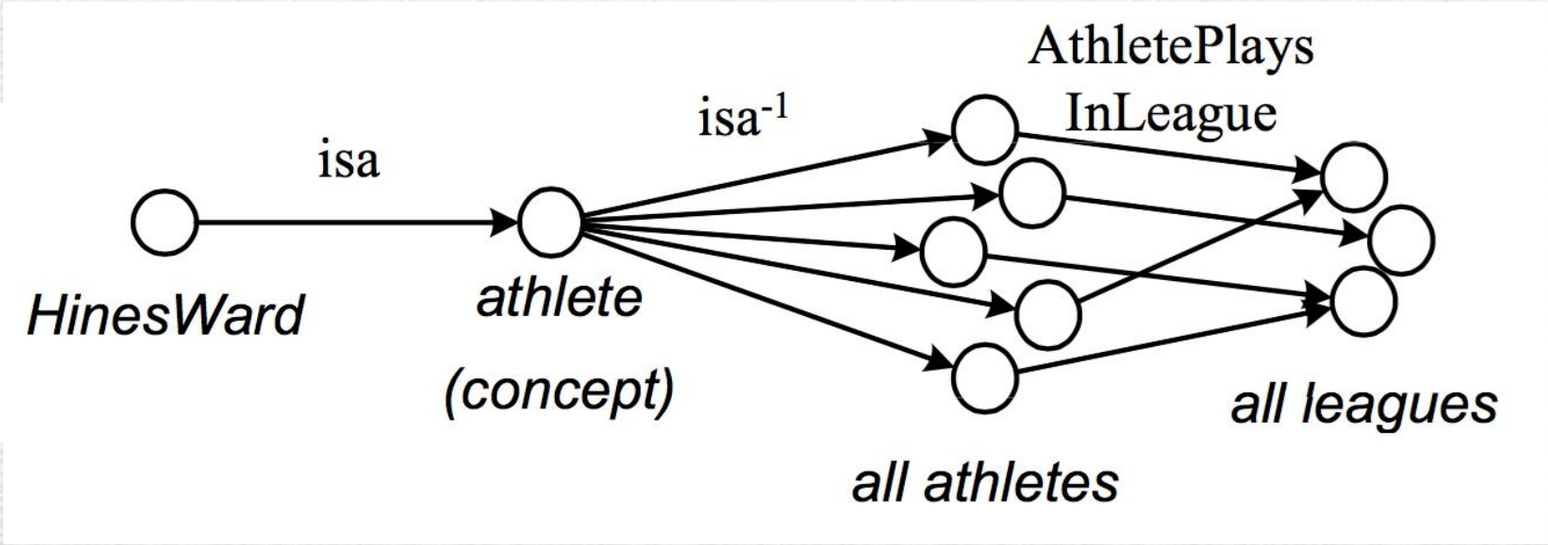
Expert scores are relational features

Score(y) = $|A_y| / |A|$

Many such low-precision high-recall experts

+ Rishabh

$$
\begin{aligned}
& isa(x, c) \wedge isa^{-1}(c, x') \\
\wedge \quad & AthletePlaysInLeague(x', y) \\
\Rightarrow \quad & AthletePlaysInLeague(x, y)
\end{aligned}
$$



# Path Ranking Algorithm

[Lao&Cohen ECML2010]

A relation path $P=(R_1, \ldots, R_n)$ is a sequence of relations

A PRA model scores a source-target node pair by a linear function of their path features

$$score(e; s) = \sum_{P \in \mathbf{P}_\ell} h_{s,P}(e)\theta_P$$

$$h_{s,P}(e) = \sum_{e' \in range(P')} h_{s,P'}(e') \cdot P(e|e'; R_\ell)$$

$$h_{s,P}(e) = \begin{cases} 1, & \text{if } e = s \\ 0, & \text{otherwise} \end{cases}$$

# Path Ranking Algorithm

[Lao&Cohen ECML2010]

Training :

For a relation R and a set of node pairs { $(s_i, t_i)$ }, we construct a training dataset D = { $(x_i, y_i)$ },where

– $x_i$ is a vector of all the path features for $(s_i, t_i)$, and
– $y_i$ indicates whether $R(s_i, t_i)$ is true
– $\theta$ is estimated using regularized logistic regression

# Link Prediction Task

Consider 48 relations for which NELL database has more than 100 instances

Two link prediction tasks for each relation

– AthletePlaysInLeague(HinesWard,?)
– AthletePlaysInLeague(?, NFL)

The actual nodes y known to satisfy R(x; ?) are treated as labeled positive examples, and all other nodes are treated as negative examples

# Captured paths/rules

Broad coverage rules

**athletePlaysSport**

$$c \xrightarrow{\text{isa}} c \xrightarrow{\text{isa}^{-1}} c \xrightarrow{\text{athletePlaysSport}} c \text{ (popular sports of all the athletes)}$$

$$c \xrightarrow{\text{athletePlaysInLeague}} c \xrightarrow{\text{superpartOfOrganization}} c \xrightarrow{\text{teamPlaysSport}} c \text{ (popular sports of a certain league)}$$

Accurate rules

**teamPlaysInCity**

$$c \xrightarrow{\text{teamHomeStadium}} c \xrightarrow{\text{stadiumLocatedInCity}} c \text{ (city of the team's home stadium)}$$

+ Gagan

+ Akshay

# Captured paths/rules

## Broad coverage rules

**athletePlaysSport**

$$c \xrightarrow{\text{isa}} c \xrightarrow{\text{isa}^{-1}} c \xrightarrow{\text{athletePlaysSport}} c \text{ (popular sports of all the athletes)}$$

$$c \xrightarrow{\text{athletePlaysInLeague}} c \xrightarrow{\text{superpartOfOrganization}} c \xrightarrow{\text{teamPlaysSport}} c \text{ (popular sports of a certain league)}$$

## Accurate rules

**teamPlaysInCity**

$$c \xrightarrow{\text{teamHomeStadium}} c \xrightarrow{\text{stadiumLocatedInCity}} c \text{ (city of the team's home stadium)}$$

# Captured paths/rules

Rules with synonym information

$$c \xrightarrow{\text{teamMember}} c \xrightarrow{\text{athletePlaysForTeam}} c \xrightarrow{\text{teamHomeStadium}} c$$

(home stadium of teams which share players with the query)

Rules with neighbourhood information

**teamPlaysInLeague**

$$c \xrightarrow{\text{teamPlaysSport}} c \xrightarrow{\text{players}} c \xrightarrow{\text{athletePlaysInLeague}} c$$

# Captured paths/rules

Rules with synonym information

+ Rishab

- Rishab

$$c \xrightarrow{\text{teamMember}} c \xrightarrow{\text{athletePlaysForTeam}} c \xrightarrow{\text{teamHomeStadium}} c$$

(home stadium of teams which share players with the query)

Rules with neighbourhood information

**teamPlaysInLeague**

$$c \xrightarrow{\text{teamPlaysSport}} c \xrightarrow{\text{players}} c \xrightarrow{\text{athletePlaysInLeague}} c$$

# Data-driven Path finding

Impractical to enumerate all possible paths, even for small length l

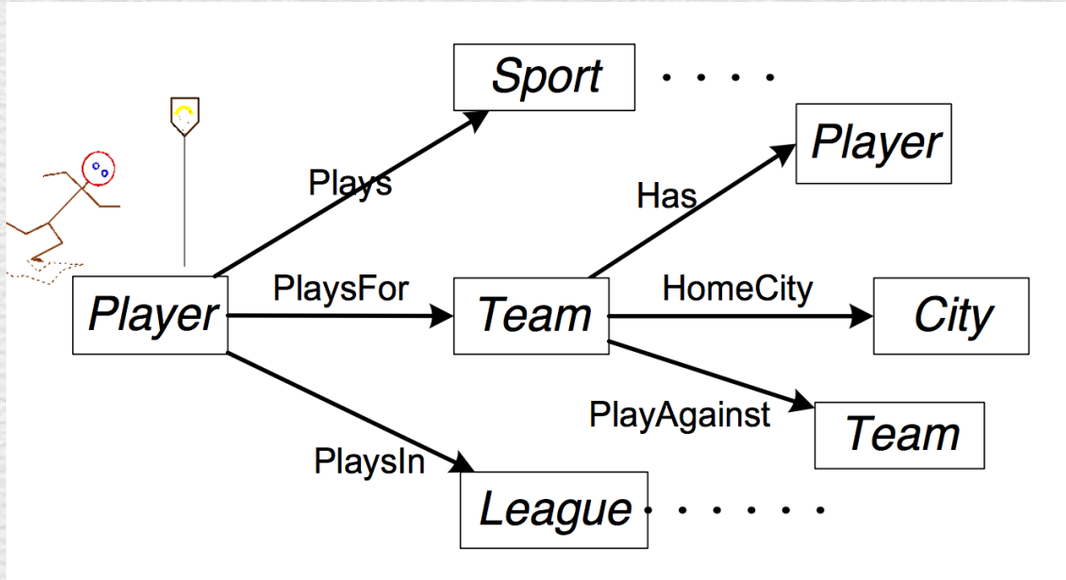
- Require any path to instantiate in at least α portion of the training queries, i.e. $h_{s,P}(t) \neq 0$ for any t
- Require any path to reach at least one target node in the training set

# Data-driven Path finding

Impractical to enumerate all possible paths, even for small length l

- Require any path to instantiate in at least α portion of the training queries, i.e. $h_{s,P}(t) \neq 0$ for any t
- Require any path to reach at least one target node in the training set

Discover paths by a depth first search : Starts from a set of training queries, expand a node if the instantiation constraint is satisfied



# Data-driven Path finding

Discover paths by a depth first search : Starts from a set of training queries, expand a node if the instantiation constraint is satisfied

Dramatically reduce the number of paths

Table 1: Number of paths in PRA models of maximum path length 3 and 4. Averaged over 96 tasks.

| | $\ell$=3 | $\ell$=4 |
|---|---|---|
| all paths up to length L | 15,376 | 1,906,624 |
| +query support$\geq \alpha = 0.01$ | 522 | 5016 |
| +ever reach a target entity | 136 | 792 |
| +$L_1$ regularization | 63 | 271 |

+ Dinesh

+ Surag

+ Haroun

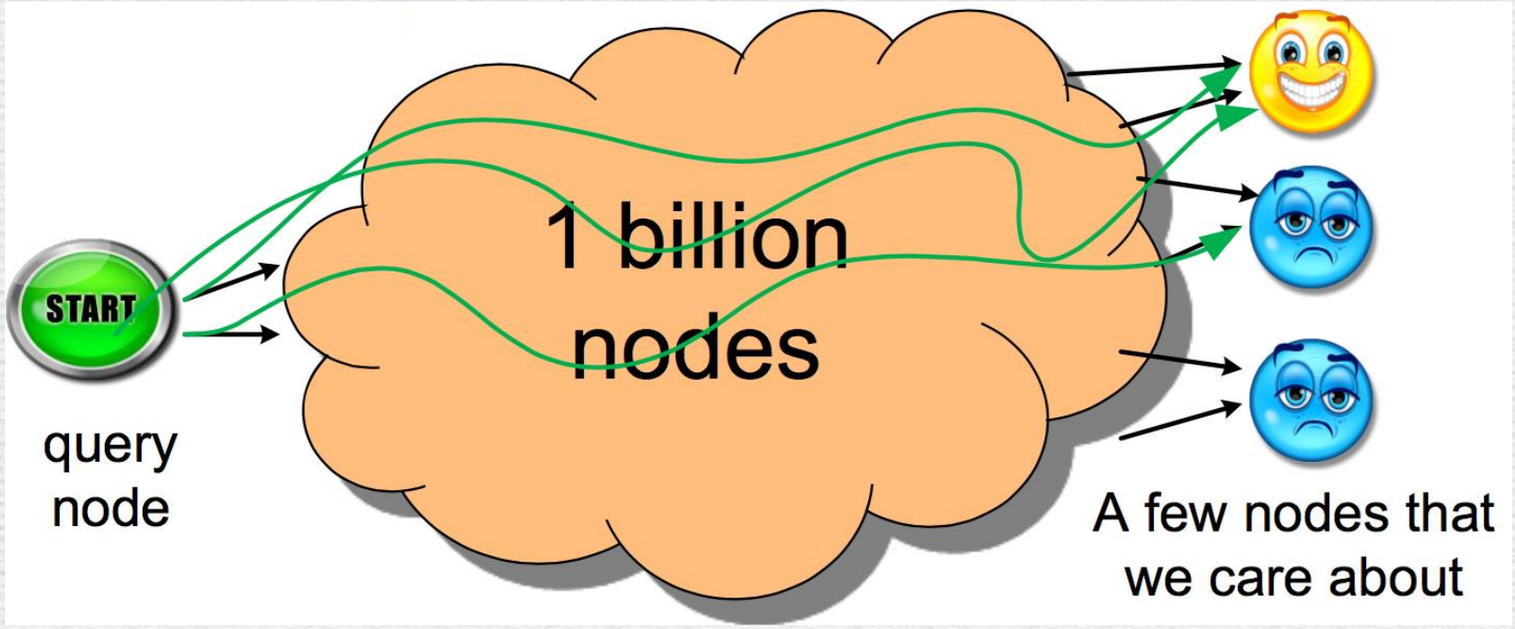
+ Nupur

# Low-Variance Sampling

[Lao&Cohen KDD2010]

Exact calculation of random walk distributions results in non-zero probabilities for many internal nodes

But computation should be focused on the few target nodes which we care about



# Low-Variance Sampling

[Lao&Cohen KDD2010]

A few random walkers (or particles) are enough to distinguish good target nodes from bad ones

Sampling walkers/particles independently introduces variances to the result distributions

# Low-Variance Sampling

Instead of generating independent samples from a distribution, LVS uses a single random number to generate all samples

Given a distribution P(x), any number r in [0, 1] corresponds to exactly one x value, namely

$$x = \arg\min_j \sum_{m=1..j} P(m) \leq r$$

# Low-Variance Sampling

Instead of generating independent samples from a distribution, LVS uses a single random number to generate all samples

Given a distribution P(x), any number r in [0, 1] corresponds to exactly one x value, namely

$$x = \arg\min_j \sum_{m=1..j} P(m) \leq r$$

To generate M samples from P(x), generate a random r in the interval [0, 1/M]

Repeatedly add the fixed amount 1/M to r and choose x values corresponding to the resulting numbers

+ Akshay + Nupur + Arindam

# Low-Variance Sampling

Instead of generating independent samples from a distribution, LVS uses a single random number to generate all samples

Given a distribution P(x), any number r in [0, 1] corresponds to exactly one x value, namely

$$x = \arg\min_j \sum_{m=1..j} P(m) \leq r$$

To generate M samples from P(x), generate a random r in the interval [0, 1/M]

Repeatedly add the fixed amount 1/M to r and choose x values corresponding to the resulting numbers

# Comparison

Inductive logic programming (e.g. FOIL)

– Brittle facing uncertainty

Statistical relational learning (e.g. MLNs, Relational Bayesian Networks)

– Inference is costly when the domain contains many nodes

– Inference is needed at each iteration of optimization

Random walk inference

– Decouples feature generation and learning : No inference during optimization

– Sampling schemes for efficient random walks : Trains in minutes, not days

– Low precision/high recall rules as features with fractional values : Doubles precision at rank 100 compared with N-FOIL

– Handles non-functional predicates

# Comparison

Inductive logic programming (e.g. FOIL)

– Brittle facing uncertainty

Statistical relational learning (e.g. MLNs, Relational Bayesian Networks)

– Inference is costly when the domain contains many nodes

– Inference is needed at each iteration of optimization

Random walk inference

– Decouples feature generation and learning : No inference during optimization

– Sampling schemes for efficient random walks : Trains in minutes, not days

– Low precision/high recall rules as features with fractional values : Doubles precision at rank 100 compared with N-FOIL

– Handles non-functional predicates

# Eval : Cross-val on training

Table 2: Comparing PRA with RWR models. MRRs and training times are averaged over 96 tasks.

| | $\ell$=2 | | $\ell$=3 | |
|---|---|---|---|---|
| | **MRR** | **Time** | **MRR** | **Time** |
| RWR(no train) | 0.271 | | 0.456 | |
| RWR | 0.280 | 3.7s | 0.471 | 9.2s |
| PRA | 0.307 | 5.7s | 0.516 | 15.4s |

Mean Reciprocal Rank : inverse rank of the highest ranked relevant result (higher is better)

# Eval : Cross-val on training

Table 2: Comparing PRA with RWR models. MRRs and training times are averaged over 96 tasks.

| | $\ell$=2 | | $\ell$=3 | |
|---|---|---|---|---|
| | **MRR** | **Time** | **MRR** | **Time** |
| RWR(no train) | 0.271 | | 0.456 | |
| RWR | 0.280 | 3.7s | 0.471 | 9.2s |
| PRA | 0.307 | 5.7s | 0.516 | 15.4s |

Mean Reciprocal Rank : inverse rank of the highest ranked relevant result (higher is better)

- Gagan
- Haroun

# Eval : Cross-val on training

Table 2: Comparing PRA with RWR models. MRRs and training times are averaged over 96 tasks.

| | $\ell$=2 | | $\ell$=3 | |
|---|---|---|---|---|
| | **MRR** | **Time** | **MRR** | **Time** |
| RWR(no train) | 0.271 | | 0.456 | |
| RWR | 0.280 | 3.7s | 0.471 | 9.2s |
| PRA | 0.307 | 5.7s | 0.516 | 15.4s |

Supervised training can improve retrieval quality (RWR)

RWR : One-parameter-per-edge label, ignores context

Path structure can produce further improvement (PRA)

# Eval : Effect of sampling

LVS can slightly improve prediction for both finger printing and particle filtering

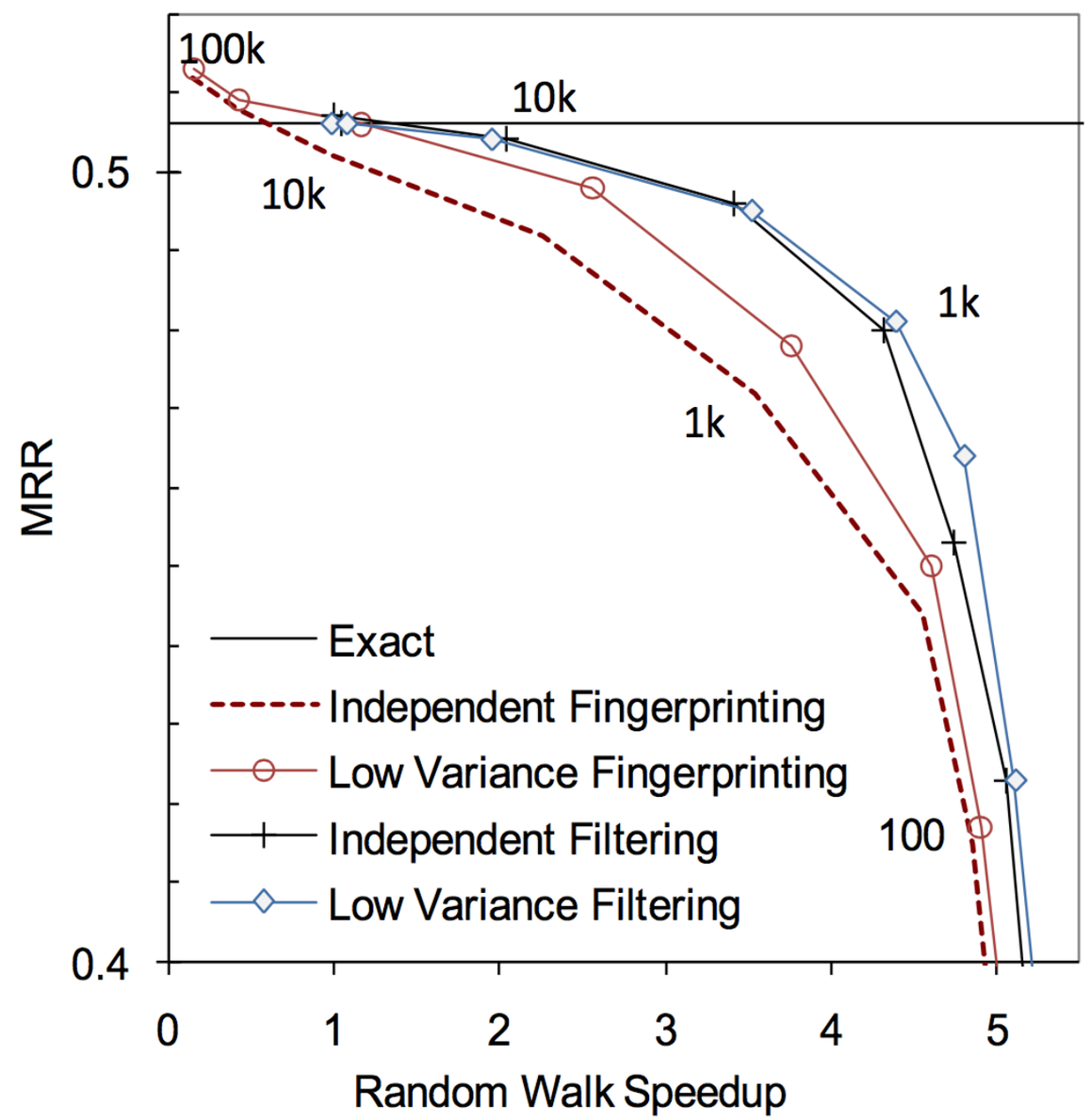



Figure 2: Compare inference speed and quality over 96 tasks. The speedup is relative to exact inference, which is on average 23ms per query.

+ Himanshu

- Surag

# AMT evaluation

| | Functional Predicates | | | | Non-functional Predicates | | | |
|---|---|---|---|---|---|---|---|---|
| | #Rules | p@10 | p@100 | p@1000 | #Rules | p@10 | p@100 | p@1000 |
| N-FOIL | 2.1(+37) | 0.76 | 0.380 | 0.007 | ---- | ---- | ---- | ---- |
| PRA | 43 | 0.79 | **0.668** | **0.615** | 92 | **0.65** | **0.620** | **0.615** |

Sorted the queries for each predicate according to the scores of their top-ranked results, and then evaluated precisions at top 10, 100 and 1000 queries

# Discussion

Dinesh : miss out on knowledge not present in the path. one hop neighbours as features?

Gagan : compare average values for highest ranked relevant result instead of MRR; comparison to MLNs

Rishab, Barun, Surag : Analysis of low MRR/errors

Rishab : low path scores for more central nodes

Shantanu : ignoring a relation in inferring itself? Same relation with different arguments

# Extensions

- Multi-concept inference : Gagan
- SVM classifiers : Rishab, Nupur, Surag
- Joint inference : Paper, Rishab, Gagan, Barun, Haroun
- Relation embeddings : Rishab
- Path pruning using horn clauses : Barun
- Target node statistics : Paper, Barun, Nupur
- Tree kernel SVM : Akshay

# Extensions

- Longer paths : Paper, Haroun
- Lexicalized paths : Paper
- Generalize paths to trees : Paper, Haroun
- No path between source and target; relation similarity : Arindam
- Information sharing across relations; MLN layer : Ankit
- Weigh edges by confidence : Surag
- Aligned graph on multiple data sources : Prachi

The End