Rule mining with Numerical Predicates

Plate-Forme Intelligence Artificielle

Atelier Decade

DEcouverte de Connaissances et Apprentissage dans les Données graphEs

Armita Khajeh nassiri, Nathalie Pernelle, Fatiha Saïs

LISN - CNRS - Paris Saclay University - University Sorbonne Paris Nord

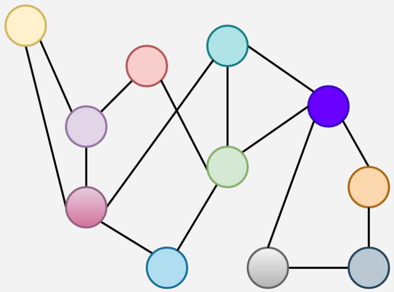
Overview

- Rule mining : some techniques
- Our Approach: Rules with numerical predicates
- KG completion: Rules vs KG Embedding
- Ongoing and Future work

Overview

- **Rule mining : some techniques**
- Our Approach: Rules with numerical predicates
- KG completion: Rules vs KG Embedding
- Ongoing and Future work

Input: Knowledge graph



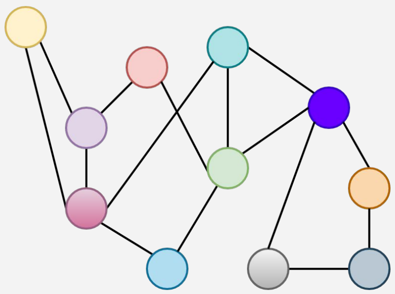
RDF Knowledge graph KG

{(h, r, t) $\subseteq$ E × R × E}

Ex: (Barack_Obama, marriedTo, Michel_Obama)

Input: Knowledge graph

LISN LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE



RDF Knowledge graph KG

$\{(h, r, t) \subseteq E \times R \times E\}$

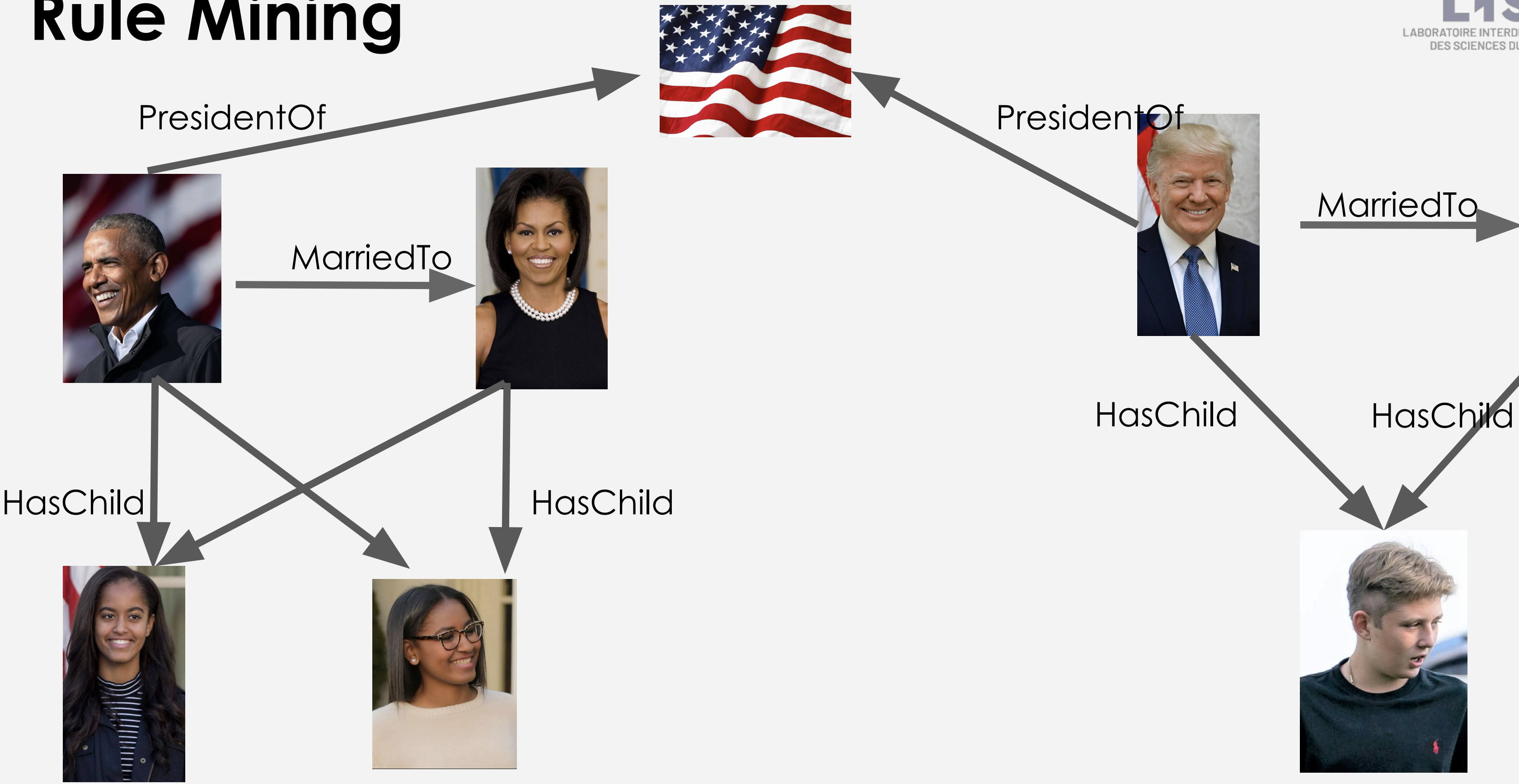
Ex: (Barack_Obama, marriedTo, Michel_Obama)

A **horn rule** or implication : $B1 \land B2 \land \ldots \land Bn \Rightarrow r(x, y)$

Body

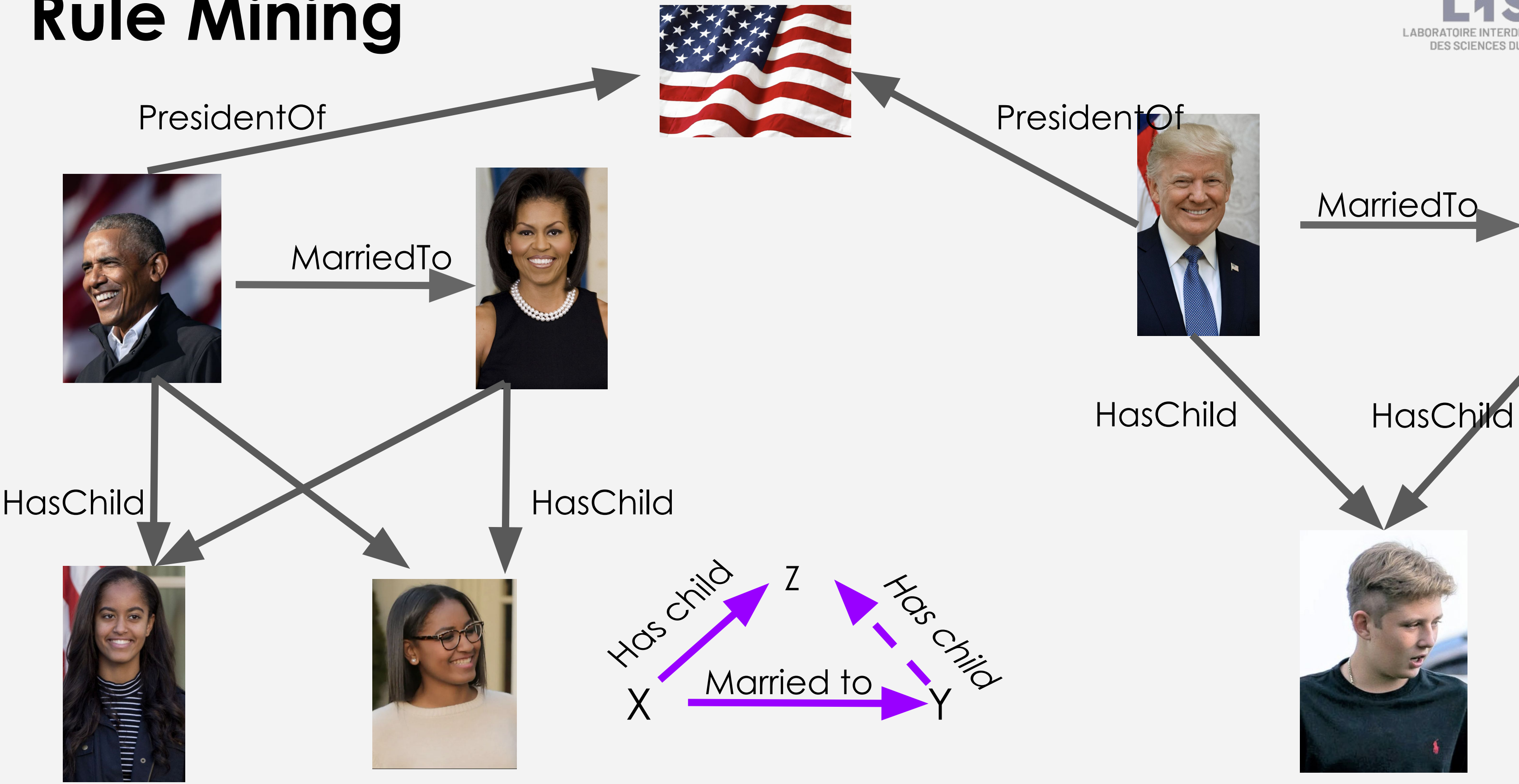
conclusion

Rule Mining



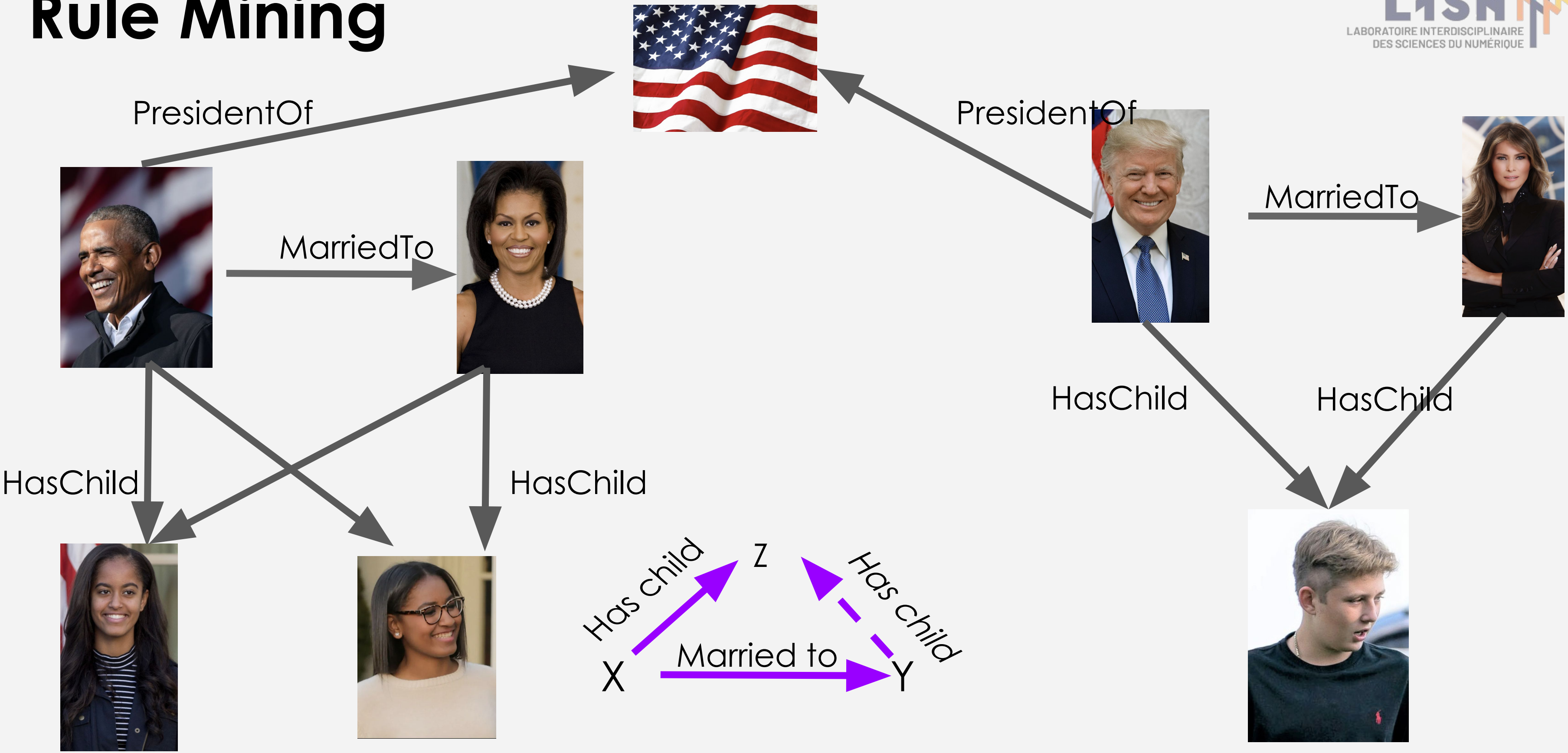
PresidentOf
PresidentOf
MarriedTo
MarriedTo
HasChild
HasChild
HasChild
HasChild

Rule Mining



$$\text{hasChild}(X,Y) \wedge \text{marriedTo}(X,Z) \rightarrow \text{hasChild}(Z,Y)$$

Rule Mining



hasChild(X,Y) ∧ marriedTo(X,Z) → hasChild(Z,Y)

→ Quality measures : **confidence** , **support** , **head Coverage**

Rule Mining

Rule mining techniques can be categorized into :

- Generate and test Techniques

- Divide and Conquer Techniques

Rule Mining

Rule mining techniques can be categorized into :

- Generate and test Techniques

- **Divide and Conquer Techniques : TILDE**

1. Tilde: Top-down induction of first-order logical decision trees. *Artificial Intelligence, Blockeel, H., & De Raedt, L. (1998)*

Rule Mining

Rule mining techniques can be categorized into :

- **Generate and test Techniques : AMIE3 (AMIE+,AMIE), RUDIK, AnyBURL**

- Divide and Conquer Techniques : TILDE

1. AMIE3: Fast and exact rule mining with amie 3. In A. Harth et al. (Eds.), The semantic web, *Blockeel, Lajus, J., Galárraga, L., & Suchanek, F. (2020)*
2. *Rudik: Robust Discovery of Positive and Negative Rules in Knowledge-Bases. VLDB Endow. Ortona, S., Meduri, V. V., & Papotti, P. (2018)*
3. *AnyBURL: Anytime Bottom-Up Rule Learning for Knowledge Graph, IJCAI, Meilicke, C., Chekol, M. W., Ruffinelli, D., & Stuckenschmidt, H. (2019)*

Rule Mining - Generate and test Techniques

Heuristic technique with backtracking :

1- Consider a candidate rule

2- Compute quality measures for this rule

3- **Refine** the rule to generate more candidates and test

Guarantees to find rules that fulfill quality measures and the language bias

Rule Mining - Language Bias

Language bias is a trade off between **expressivity** and **performance**

- No Reflexive atoms : loves(Barack, Barack)
- Connected : diedIn(x, y) ⇒ wasBornIn(w, z)
- Closed : marriedTo(x, y) ∧ worksAt(x, z) ⇒ marriedTo(y, x)

Rule Mining - AMIE

- Start from all possible rules of the form **=> r (x,y)**
- Refine: adding dangling atom, closing atom, instantiated atom

child (a,e) ∧ sibling(e, b) => child (a, b)

Rule Mining - AMIE

- Start from all possible rules of the form **=> r (x,y)**
- Refine: adding dangling atom, closing atom, instantiated atom

$$\text{child}(a,e) \wedge \text{sibling}(e, b) \Rightarrow \text{child}(a, b)$$

=> child (a, b)

Add dangling atom — sibling(e, **b**) => child (a, **b**)

Rule Mining - AMIE

LISN LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

- Start from all possible rules of the form **=> r (x,y)**
- Refine: adding dangling atom, closing atom, instantiated atom

child (a,e) ∧ sibling(e, b) => child (a, b)

	=> child (a, b)
Add dangling atom	sibling(e, b) => child (a, b)
Add closing atom	child (**a**,e) ∧ sibling(e, b) => child (**a**, b)

Rule Mining - AMIE

- Start from all possible rules of the form **=> r (x,y)**
- Refine: adding dangling atom, closing atom, instantiated atom
- Optimization:

AMIE3 has managed to speed up the rule mining approach by a factor of 15 compared to other state of the art.

Exhaustive and efficient !

Rule Mining - RUDIK

- Logical rules like AMIE, but:
    - can also mine negative rules: motherOf (m, c) ⇒ ¬ fatherOf (m, c)
    - can mine relations between literals:
      rel(a, b) such that *rel* ∈ {<, ≤, ≠, ≥, >} and *a, b* are numeric

      Ex: hasAge(X, a) ∧ hasAge(Y, b) ∧ a > b ⇒ notChild(X, Y)

Overview

- Rule mining : some techniques
- **Our Approach: Rules with numerical predicates**
- KG completion: Rules vs KG Embedding
- Ongoing and Future work

On the Shoulders of AMIE

- We propose a post-processing step to the rules mined by AMIE

- **Objective** : add atoms with numerical predicates to the rule such that the <u>confidence is increased</u>

- Challenge: Numerical predicates take a wide range of values

On the Shoulders of AMIE

- We propose a post-processing step to the rules mined by AMIE

- **Objective** : add atoms with numerical predicates to the rule such that the <u>confidence is increased</u>

- Challenge: Numerical predicates take a wide range of values

How to choose "good" intervals?

On the Shoulders of AMIE

LISN LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

minhc and minConf are satisfied

shares_border_with(a,b) => diplomatic_relation (a,b)

place_of_work(a,b) => place_of_birth (a,b)

child (a,e) ∧ sibling(e, b) => child (a, b)

country (f, b) ∧ employer(a,f) => residence(a,b)

country_of_citizenship(f,b) ∧ screenwriter (a, f) => country_of_origin (a, b)

student_of(a ,b) => doctoral_advisor (a,b)

Running example

shares_border_with(a,b) => diplomatic_relation (a,b)

place_of_work(a,b) => place_of_birth (a,b)

child (a,e) ∧ sibling(e, b) => child (a, b)

country (f, b) ∧ employer(a,f) => residence(a,b)

country_of_citizenship(f,b) ∧ screenwriter (a, f) => country_of_origin (a, b)

student_of(a ,b) => doctoral_advisor (a,b)

Running example

LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

place_of_work (a,b) => place_of_birth(a,b)

Enrich the rule by trying every numerical predicate with every variable in rule

has_Population
has_GDP
inflation_rate
date_of_birth
date_of_death
mass_Kilogram
vertical_depth_meter

a, b

Running example

LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

place_of_work (a,b) => place_of_birth(a,b)

has_Population (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_Population (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_GDP (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_GDP (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
date_of_birth (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
date_of_birth (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
…

Running example

place_of_work (a,b) => place_of_birth(a,b)

has_Population (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_Population (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_GDP (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
has_GDP (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
date_of_birth (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
date_of_birth (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
…

Run SPARQL queries, prune those that have no chance of satisfying the minhc.

Running example

place_of_work (a,b) => place_of_birth(a,b)

~~**has_Population (a, X)** ∧ place_of_work (a,b) => place_of_birth(a,b)~~
has_Population (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
~~**has_GDP (a, X)** ∧ place_of_work (a,b) => place_of_birth(a,b)~~
has_GDP (b, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
date_of_birth (a, X) ∧ place_of_work (a,b) => place_of_birth(a,b)
~~**date_of_birth (b, X)** ∧ place_of_work (a,b) => place_of_birth(a,b)~~
…

Run SPARQL queries, prune those that have no chance of satisfying the minhc.

Running example

date_of_birth (a,X) $\wedge$ place_of_work (a,b) => place_of_birth(a,b)

- Turn into a classification problem, based on functionality of the conclusion predicate

$$\forall a : \text{date\_of\_birth}(a, X) \wedge \text{place\_of\_work}(a, y) \wedge \text{place\_of\_birth}(a,z)$$

$$\begin{cases} \text{Class} = 1 & \text{if } y = z \\ \text{Class} = 0 & \text{if } y \neq z \end{cases}$$

Running example

LABORATOIRE INTERDISCIPLINAIRE DES SCIENCES DU NUMÉRIQUE

date_of_birth (a,X) $\wedge$ place_of_work (a,b) => place_of_birth(a,b)

- Classify instances based on:

$\forall a$: date_of_birth(a, X) $\wedge$ place_of_work(a, y) $\wedge$ place_of_birth(a,z)

Class =1 if $y=z$
Class =0 if $y \neq z$

Use a supervised binning technique that discretizes the values of the numerical predicate based on the class labels

Running example

date_of_birth (a,X) ∧ place_of_work (a,b) => place_of_birth(a,b)

Supervised binning: Optimal binning, MDLP, Entropy, etc.

-inf 1800	1801-1820	1820 1945	1945 inf

Running example

date_of_birth (a,X) ∧ place_of_work (a,b) => place_of_birth(a,b)

- Sort the intervals based on "not event rate" to opt for higher confidence

~~1945 inf~~	1801-1820	1820 1945	-inf 1800

- For each interval, try exclude it from the rule

  Prune: Whenever "not event rate" less than the confidence of parent rule

Running example

date_of_birth (a,X) ∧ place_of_work (a,b) => place_of_birth(a,b)

- Sort the intervals based on "not event rate" to opt for higher confidence

~~1945 inf~~	1801-1820	1820 1945	-inf 1800

- For each interval, try exclude it from the rule
  Prune: Whenever "not event rate" less than the confidence of parent rule

- Recompute the quality measures of head coverage and confidence

- Keep the new rule if confidence increased wrt. Parent conf and minhc satisfied

Running example

place_of_work (a,b) => place_of_birth(a,b)

date_of_birth(a,y) $\wedge$ y $\notin$ [1945, ∞] $\wedge$ place_of_work(a,b) => place_of_birth(a,b)

date_of_birth(a,y) $\wedge$ y$\notin$[1801, 1820] $\wedge$ place_of_work(a,b) => place_of_birth(a,b)

On the shoulders of AMIE

We have implemented options for merging intervals and keeping new rules

- Merge the biggest consecutive intervals
- Merge all intervals with same predicate and variable

The quality measures should be re-computed.

On the shoulders of AMIE

As long as the max number of atoms defined by the user allows:

- Add atoms with different numerical predicates

population(b, y) ∧ y∉ [-∞, 100K] ∧ date_of_birth (a,z) ∧ z ∉ [1990, ∞] ∧ place_of_work (a,b) => place_of_birth(a,b)

- Prune: Predicates with variables that did not pass the minhc in previous step.

On the shoulders of AMIE

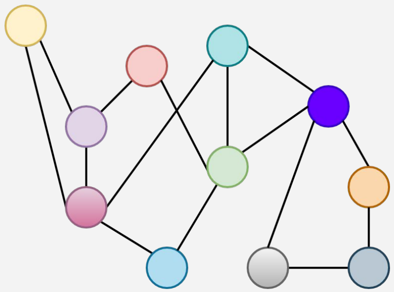
Dataset:

	FB15K-237Num	LitWD19K
#Entities	12,493	18,986
#Relations	237	182
#Attributes	116	151
#StruTriples	27,899	288,933
#AttrTriples	82,992	63,951
#Train		260,039
#Test	10,359	14,447
#Valis	10,359	14,447
#RulesAmie	**32017**	**737**
#EnrichedRules	**71200**	**4180**

Overview

- Rule mining : some techniques
- Our Approach: Rules with numerical predicates
- **KG completion: Rules vs KG Embedding**
- Ongoing and Future work

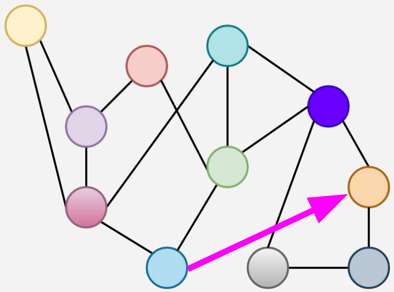
Refinement Tasks



Different Refinement tasks:

- Triple Classification
- Relation Prediction
- **KG Completion**
- Data Linking
- Error Detection
- ...

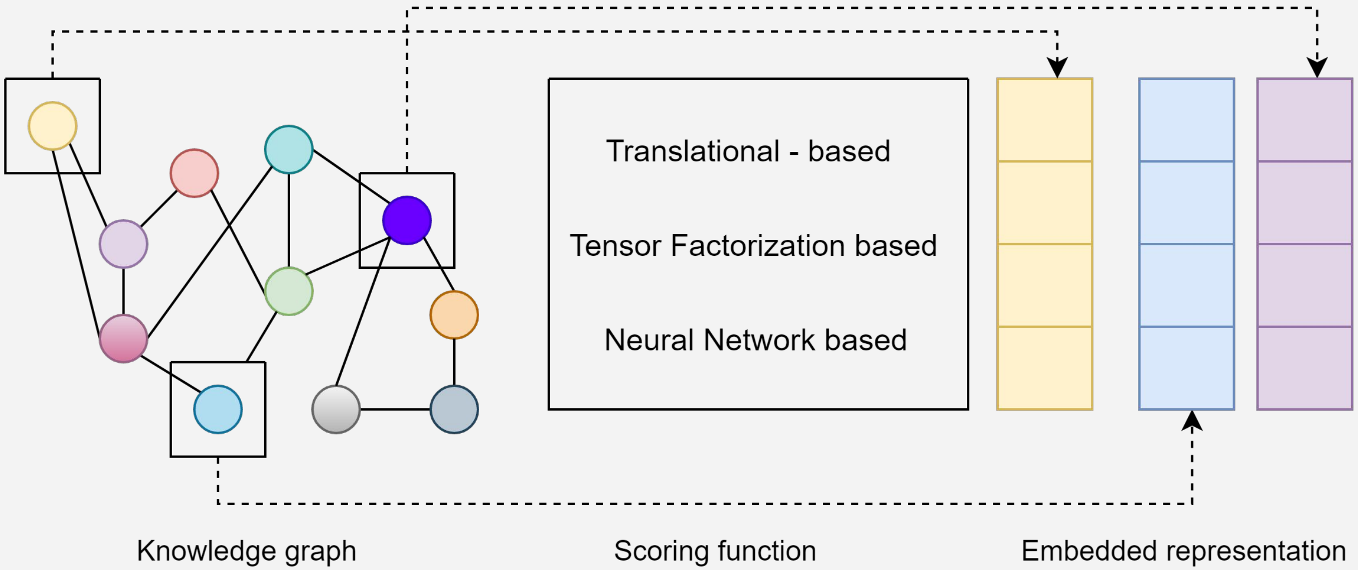
Refinement Task: KG Completion



KG Completion or Link Prediction

Predict missing links: (?, r, t)
(h, r, ?)

Knowledge graph embeddings:

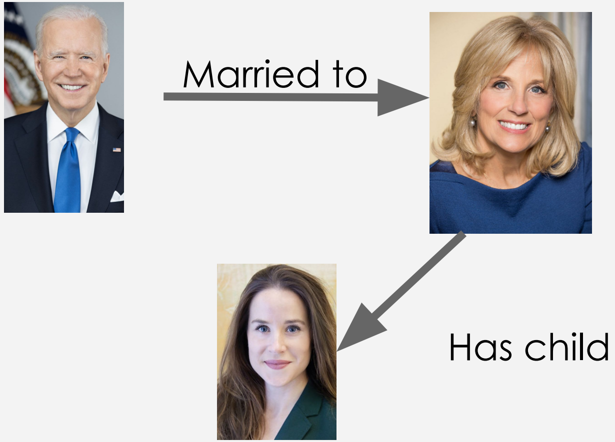


Rule Mining- KG completion

Prediction:

RULE: hasChild(X,Y) $\wedge$ marriedTo(X,Z) $\rightarrow$ hasChild(Z,Y)

X= Joe, Y= Jill, Z= Ashley : hasChild(Joe, Ashley)



Rule Mining- KG completion

date_of_birth(a,y)∧ y ∉ [1945, ∞] ∧ place_of_work (a,b) => place_of_birth(a,b)

place_of_birth(Einstein, ?)

Rule Mining- KG completion

$date\_of\_birth(a,y) \land y \notin [1945, \infty] \land place\_of\_work(a,b) \Rightarrow place\_of\_birth(a,b)$

place_of_birth(Einstein, ?)

- Run SPARQL queries, get all possible candidates for the tail of the test triple
- Aggregate over the answers

Rule Mining- KG completion

place_of_birth(Einstein, ?)

R1 [conf =0.8]	[New Jersey, Bern]
R2 [conf=0.6]	[Ulm]
R3 [conf= 0.4]	[Ulm, Berlin]

Rule Mining- KG completion

place_of_birth(Einstein, ?)

Max Confidence

R1 [conf =0.8]	[New Jersey, Bern]

R2 [conf=0.6]	[Ulm]

R3 [conf= 0.4]	[Ulm, Berlin]

Pick one at random

Rule Mining- KG completion

place_of_birth(Einstein, ?)

Most Frequent

R1 [conf =0.8] [New Jersey, Bern]

R2 [conf=0.6] [Ulm]

R3 [conf= 0.4] [Ulm, Berlin]

Overview

- Rule mining : some techniques
- Our Approach: Rules with numerical predicates
- KG completion: Rules vs KG Embedding
- **Ongoing and Future work**

Ongoing and Future work

- Exploring different binning techniques

- Studying the effects of merging intervals on the KG completion task

- KG completion: More/better aggregate strategies

- More optimizations in the pipeline

Ongoing and Future work

- Exploring different binning techniques

- Studying the effects of merging intervals on the KG completion task

- KG completion: More/better aggregate strategies

- More optimizations in the pipeline

Thank you! Questions and suggestions are more than welcome!