RecSys 2017, Como, Italy

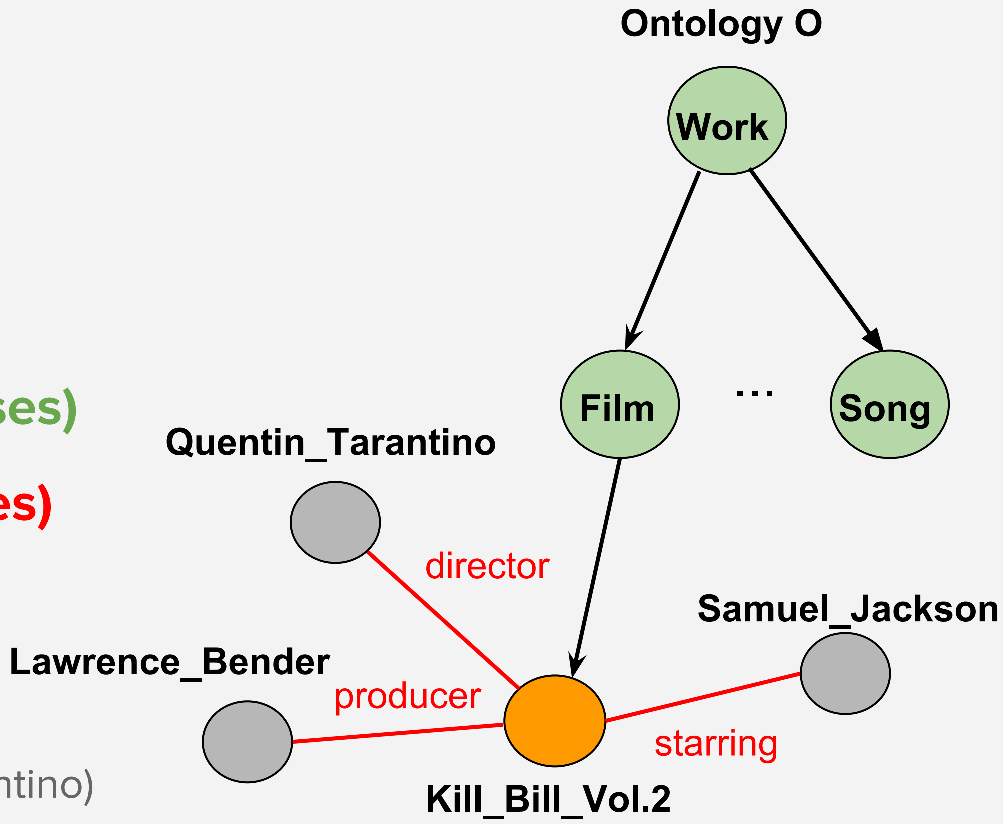
entity2rec

Learning User-Item Relatedness from Knowledge Graphs for Top-N Item Recommendation

Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy

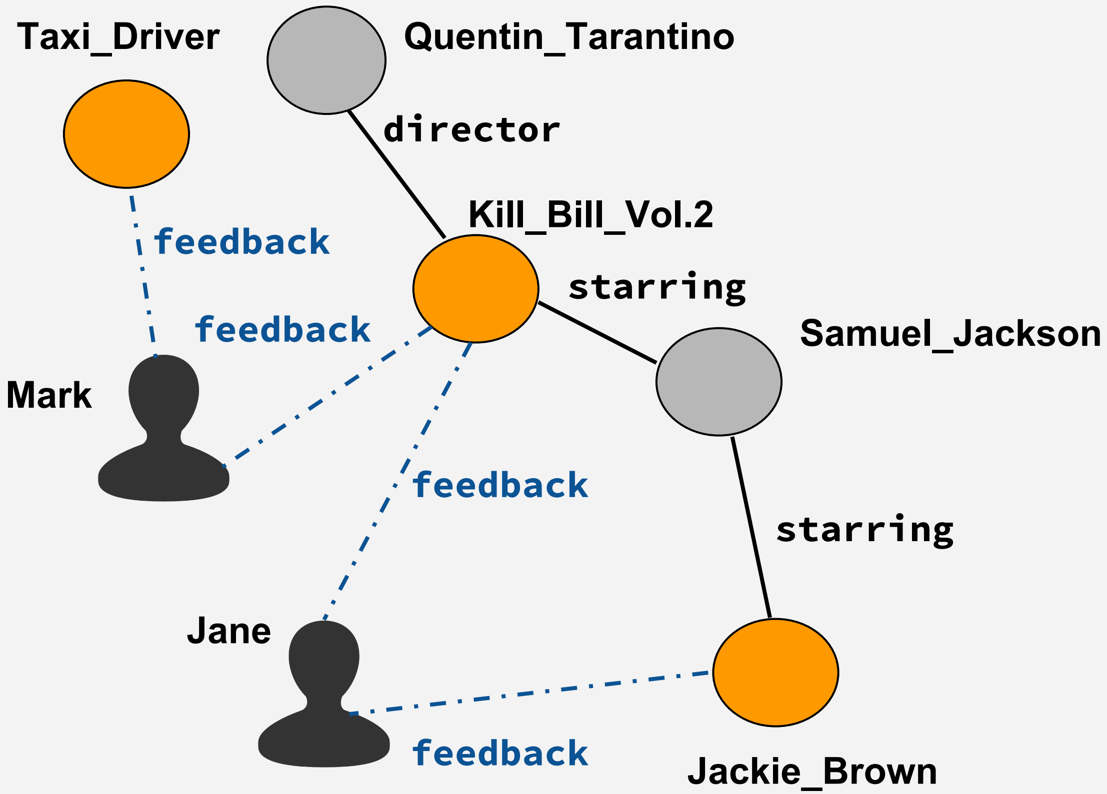
Knowledge Graph

- **K** = **(E, R, O)**
- **E** = entities
- **O** = ontology
- **O** defines **entity types Λ (classes)** and **relation types Γ (properties)**
- O: **Λ -> Γ**
- **R ⊂ E x Γ x E**
- (Kill_Bill_Vol.2, starring, Quentin_Tarantino)



Knowledge Graph for RS

- $U \subset E$ = users
- $I \subset E \setminus U$ = items
- ‘**feedback**’ property: (u, feedback, i)
- Collaborative and content information
- **Hybrid** recommender systems

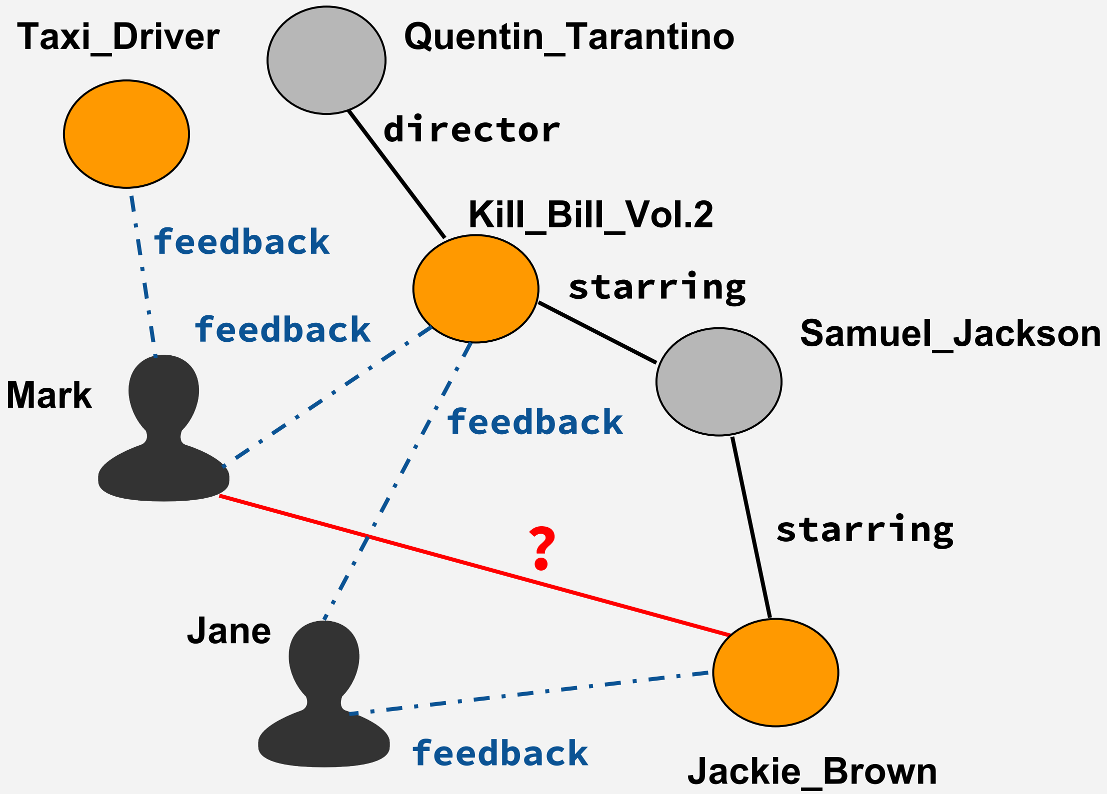


Objective

Learn **user-item relatedness** from the **Knowledge Graph** for top-N **item recommendation.**

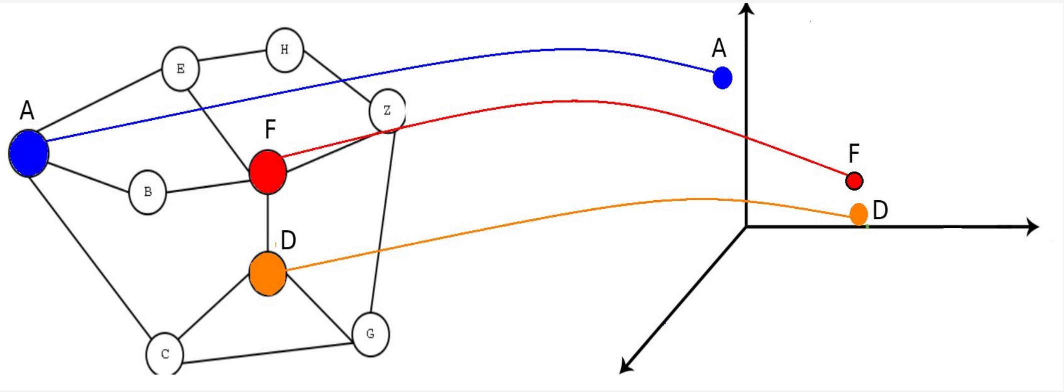
Learn entity features encoding:

- **Graph structure**: tightly connected entities should be more related
- **Semantics:** not all properties have the same importance

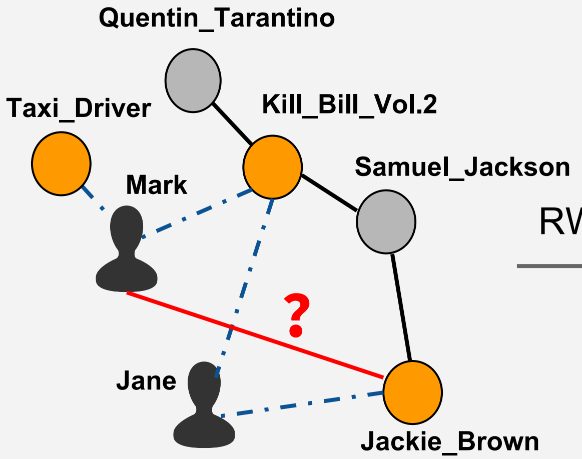


Graph structure: node2vec

- **Feature learning** from networks
- Adaptation of **word2vec** on **graph structures** using **random walks**
- Maps **nodes** in a graph into an **euclidean space** preserving the **graph structure**



Graph structure: node2vec



Graph

RW

Mark, Kill_Bill_Vol.2,
Quentin_Tarantino, ...

Jane, Kill_Bill_Vol.2,
Jurassic_Park …

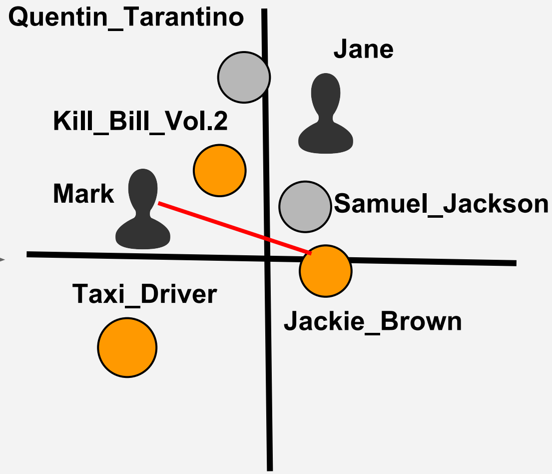
Kill_Bill_Vol.2, Samuel_Jackson,
Jurassic_Park, Jane…

Samuel_Jackson, Jurassic_Park,
Kill_Bill_Vol.2, …

…

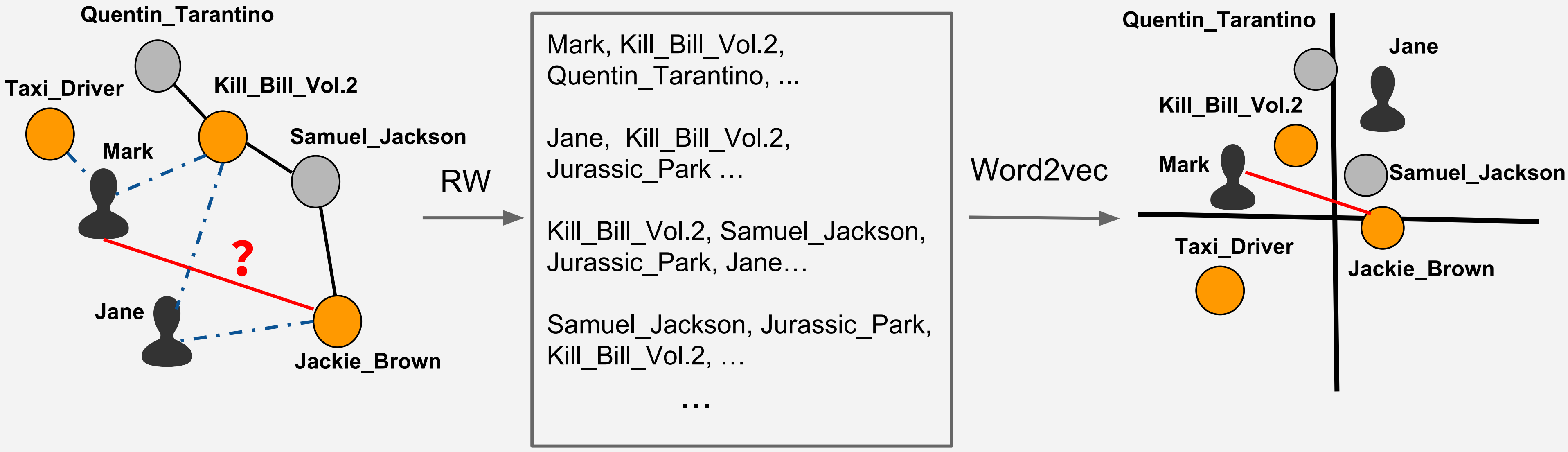
Text file with sequences of nodes

Word2vec



Node embeddings

Graph structure: node2vec



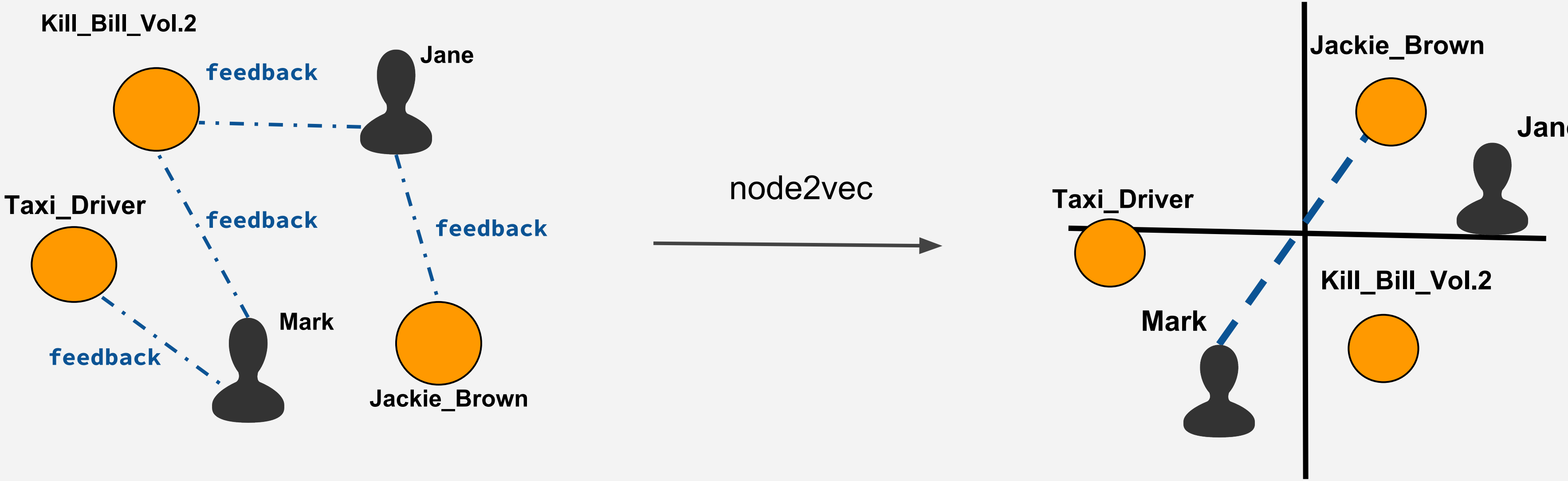
Graph

Text file with sequences of nodes

Node embeddings

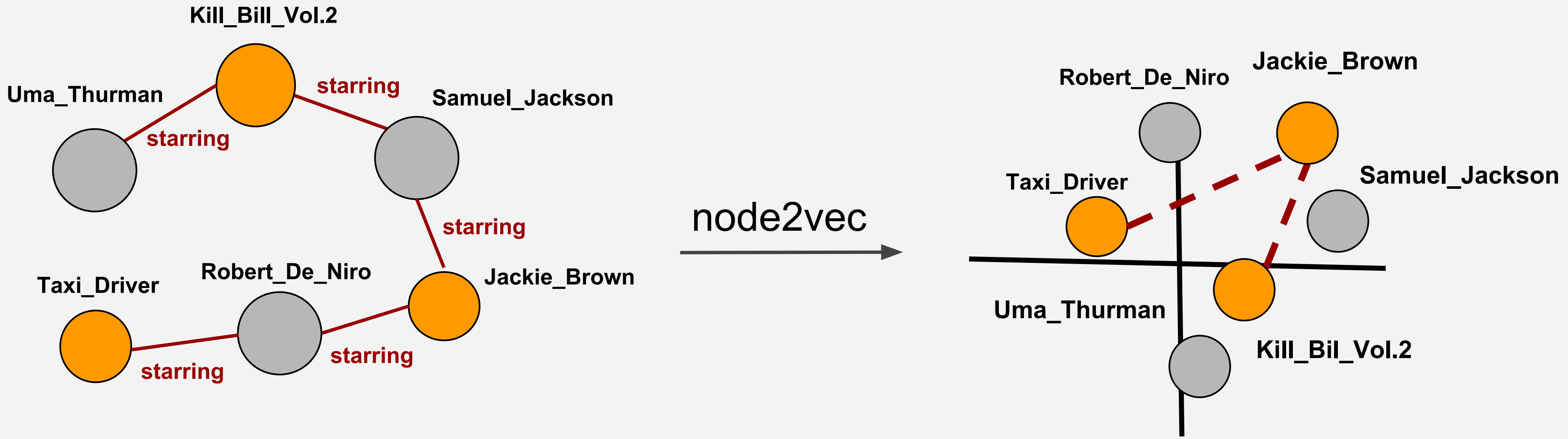
No semantics, not optimized for recommendations

Semantics: property-specific relatedness



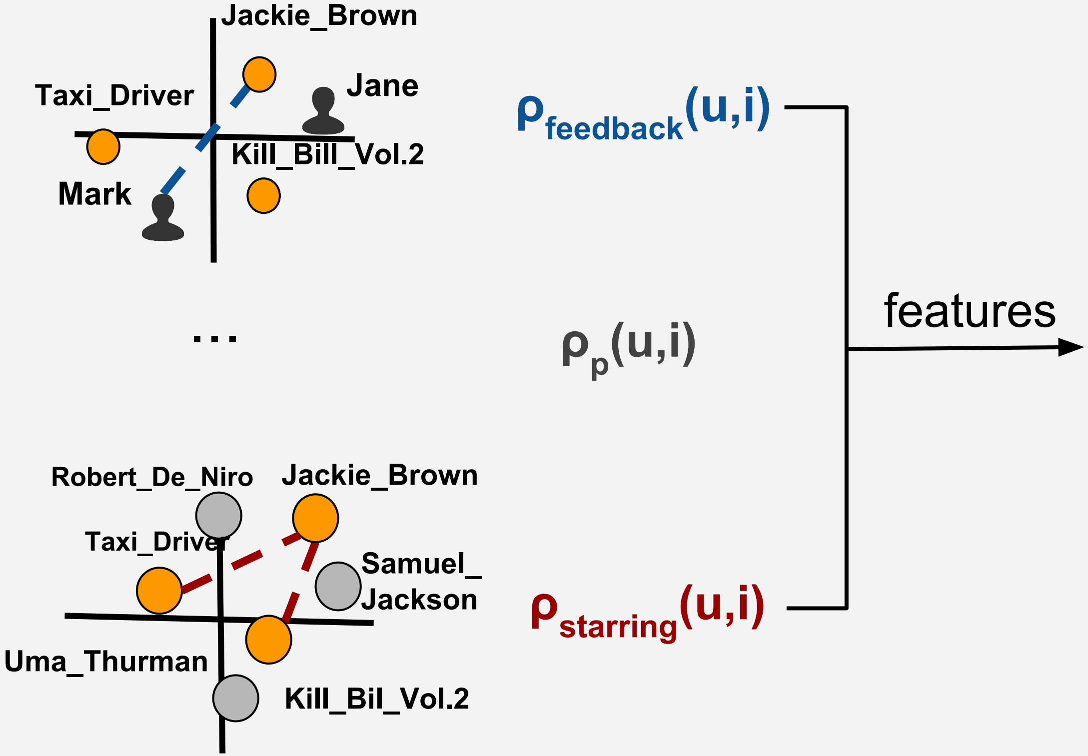
Collaborative filtering: $\rho_{feedback}(u,i) = d\ (\ x_{feedback}(u),\ x_{feedback}(i))$

Semantics: property-specific relatedness



Content filtering: $\rho_{starring}(u,i) = D\,(\,x_{starring}(i'),\, x_{starring}(i))$

Learning to rank

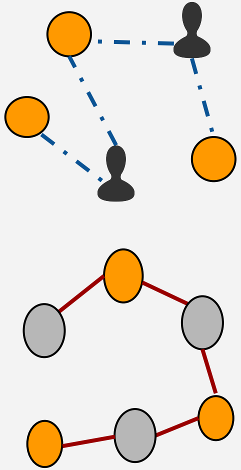


Learning to rank: $\boldsymbol{\rho(u,i) = f(\rho_p(u,i))}$

- Global user-item relatedness **ρ** as a **ranking function**
- **List-wise approach**: optimizing directly ranking with information retrieval metrics (e.g. P@N)

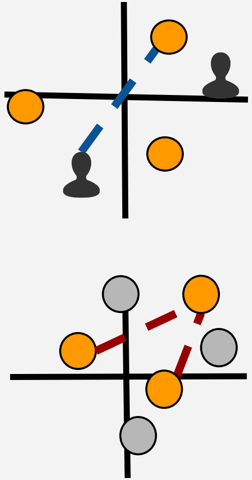
Approach

KG



node2vec

unsupervised



learning
to rank

supervised

$\rho(u,i) = f(\rho_p(u,i))$

top-N
items

Graphs:
property-specific
subgraphs

Features:
property-specific
relatedness scores

Ranking function:
global relatedness
score

Experimental setup

- **Movielens_1M:** 1,000,209 ratings, 3900 movies, 6040 users
- **DBpedia:** we leverage DBpedia mappings of Movielens items to build the knowledge graph. We use the DBpedia Ontology to define properties of the class ‘Film’. Some items are missing: **948978 ratings, 3225 movies, 6040 users**.
- **Evaluation protocol:** train, val test set split. Add 100 ‘false candidates’ to each user in the test set to simulate ‘real’ ranking scenario.
- **Baselines:** popular collaborative filtering algorithms: NMF, SVD, ItemKNN, MostPopular

Results

Model	P@5	P@10	MAP
entity2rec	**0.2814**	**0.2127**	**0.4232**
Most Popular	0.2154	0.1815	0.2907
NMF	0.1208	0.1150	0.1758
SVD	0.0543	0.0469	0.0888
ItemKNN	0.0463	0.0232	0.0990

entity2rec and baselines evaluated on the test set

Feature selection

Features	P@5	P@10	MAP
$\rho_{feedback}$	0.2317	0.1708	0.3550
$\rho_{dbo:director}$	0.0219	0.0211	0.0949
$\rho_{dbo:producer}$	0.0119	0.0241	0.0498
$\rho_{dbo:starring}$	0.0128	0.0372	0.0728
$\rho_{dct:subject}$	0.0285	0.0326	0.0688
$\rho_{dbo:writer}$	0.0061	0.0216	0.0472
entity2rec	0.2814	0.2127	0.4232

Performance using one property at the time in the learning to rank model vs global model

Conclusions

- Novel approach to learn user-item relatedness that leverages the **semantics** and the **graph structure** of the **knowledge graph** to provide **recommendations**
- **Hybrid** approach**,** includes both collaborative and content information
- **No feature engineering**, using an unsupervised feature learning stage to learn entity embeddings and derive property-specific relatedness scores and a supervised learning to rank stage to combine them
- **High ranking precision** on the Movielens dataset
- **No black box effect,** features have a clear interpretation and can be **easily configured** to a particular recommendation need
- **Parallelizable** and easily adaptable to an **online learning** context

Future work

- **Test** against **more competitive baselines** and on **more domains** (e.g. music, books, tourism...)
- New experiments to answer to questions such as:
  - what is the importance of content based features and how does it vary with different degrees of data sparsity?
  - is the unsupervised learning stage or the supervised learning to rank stage that affects the performance the most?
  - what is the best approach to adapt the system to an online learning context?
- **Parallelized** and adapted to **an online learning context**
- **Software implementation** has to be further **tested** and **extended**[1]

1: https://github.com/MultimediaSemantics/entity2rec

Thank you!

- Enrico Palumbo, Data Scientist, ISMB, PhD Student, Eurecom - PoliTo
- Mail: palumbo@ismb.it, enrico.palumbo@eurecom.fr
- Personal page: www.enricopal.github.io
- Slides: http://www.slideshare.net/EnricoPalumbo2
- Github: www.github.com/enricopal, www.github.com/MultimediaSemantics
- Twitter: https://twitter.com/enricopalumbo91

Turin, Italy, www.ismb.it

Sophia Antipolis, France, www.eurecom.fr

PasTime project

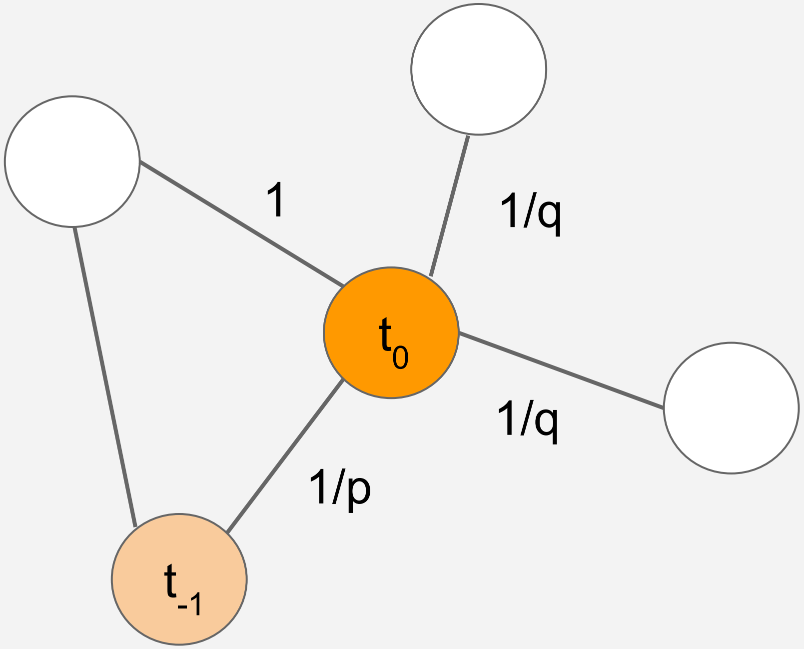
Graph structure: node2vec

Learning vector representation of nodes preserving the **local neighborhood** in the feature space.

Flexible definition of node **neighborhood** based on 2nd order **random walk** that is sensitive to different **connectivity** patterns.

p: return parameter

q: in-out parameter



Transition probability

Hyper-parameters optimization

P,Q	Learning to Rank	P@5	P@10	MAP
1,4	**LambdaMart**	**0.0791**	**0.0293**	**0.1717**
4,1	LambdaMart	0.0182	0.0193	0.0570
1,1	LambdaMart	0.0174	0.0188	0.0565
1,4	AdaRank	0.0134	0.0098	0.0278
4,1	AdaRank	0.0078	0.0083	0.0286
1,1	AdaRank	0.0109	0.0098	0.0358

Grid-search of learning to rank algorithm and (p,q) node2vec hyper-parameters on a validation set

Learning to Rank

- **AdaRank:** Xu, Jun, and Hang Li. "Adarank: a boosting algorithm for information retrieval." Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007.
  - Boosting algorithm, AdaBoost -> Ranking
  - Start from weak rankers and iteratively combines them to correct errors of the previous weak ranker
  - As a weak ranker they use the feature that has the optimal weighted ranking performance
- **LambdaMart:** Burges, Christopher JC. "From ranknet to lambdarank to lambdamart: An overview." Learning 11.23-581 (2010): 81
  - LambdaRank: directly defines the gradients without using the cost function
  - MART: boosting regression trees
  - LambdaMART: LambdaRank using MART as a ranking function

Baselines

- **ItemKNN with baselines**: *Yehuda Koren. Factor in the neighbors: scalable and accurate collaborative filtering. 2010. URL: http://courses.ischool.berkeley.edu/i290-dm/s11/SECURE/a1-koren.pdf*
- **SVD:** *Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. 2009.*
- **NMF:** *Xin Luo, Mengchu Zhou, Yunni Xia, and Qinsheng Zhu. An efficient non-negative matrix factorization-based approach to collaborative filtering for recommender systems. 2014.*
- **Python library:** http://surpriselib.com/