Knowledge Graph Embedding for Mining Cultural Heritage Data

Nada Mimouni and Jean-Claude Moissinac
–
Telecom ParisTech

Institut Mines Telecom

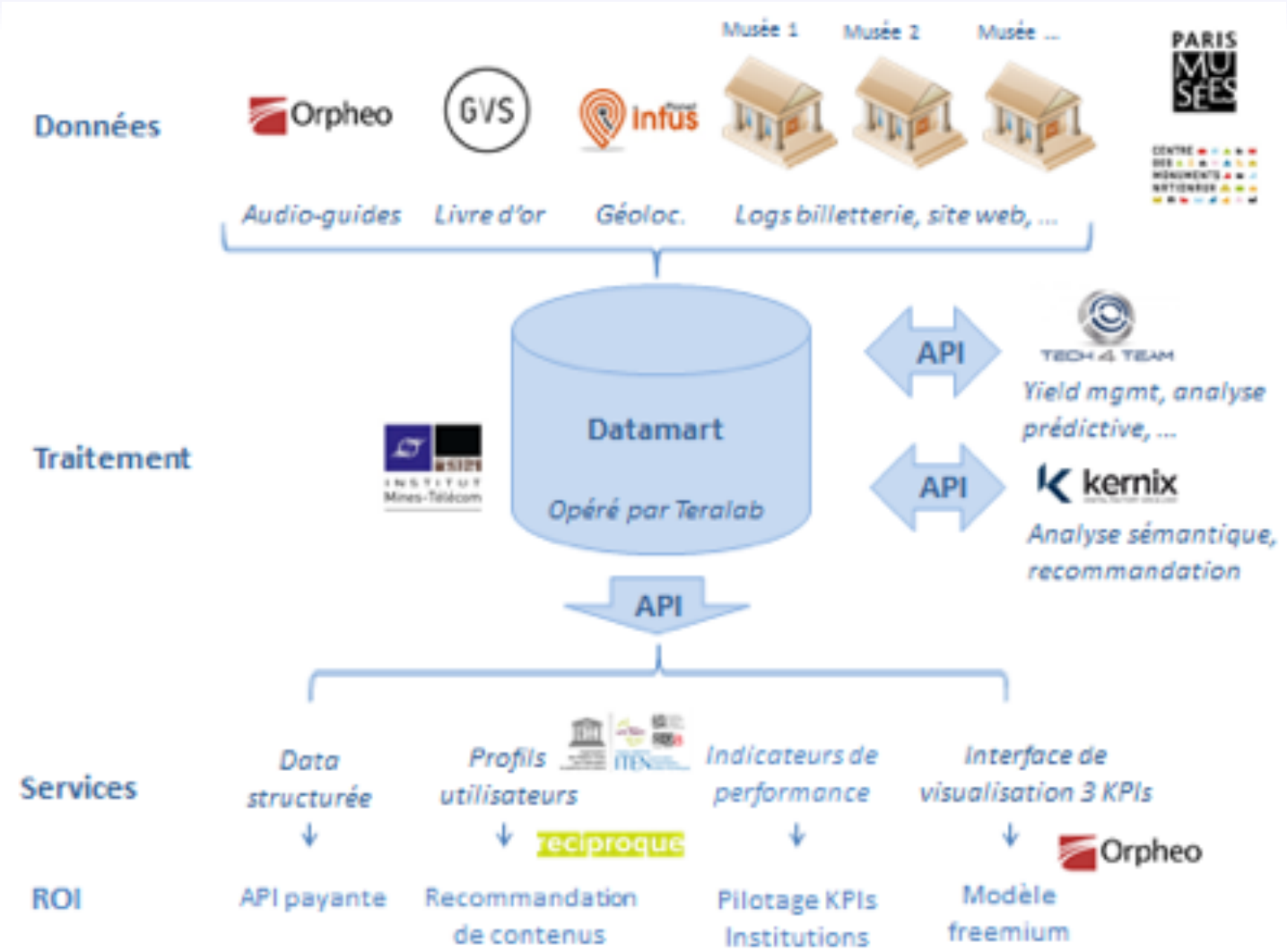
January 24th, 2019
DIG - LTCI

Outline

Outline

Project presentation

Project presentation



Outline

Data

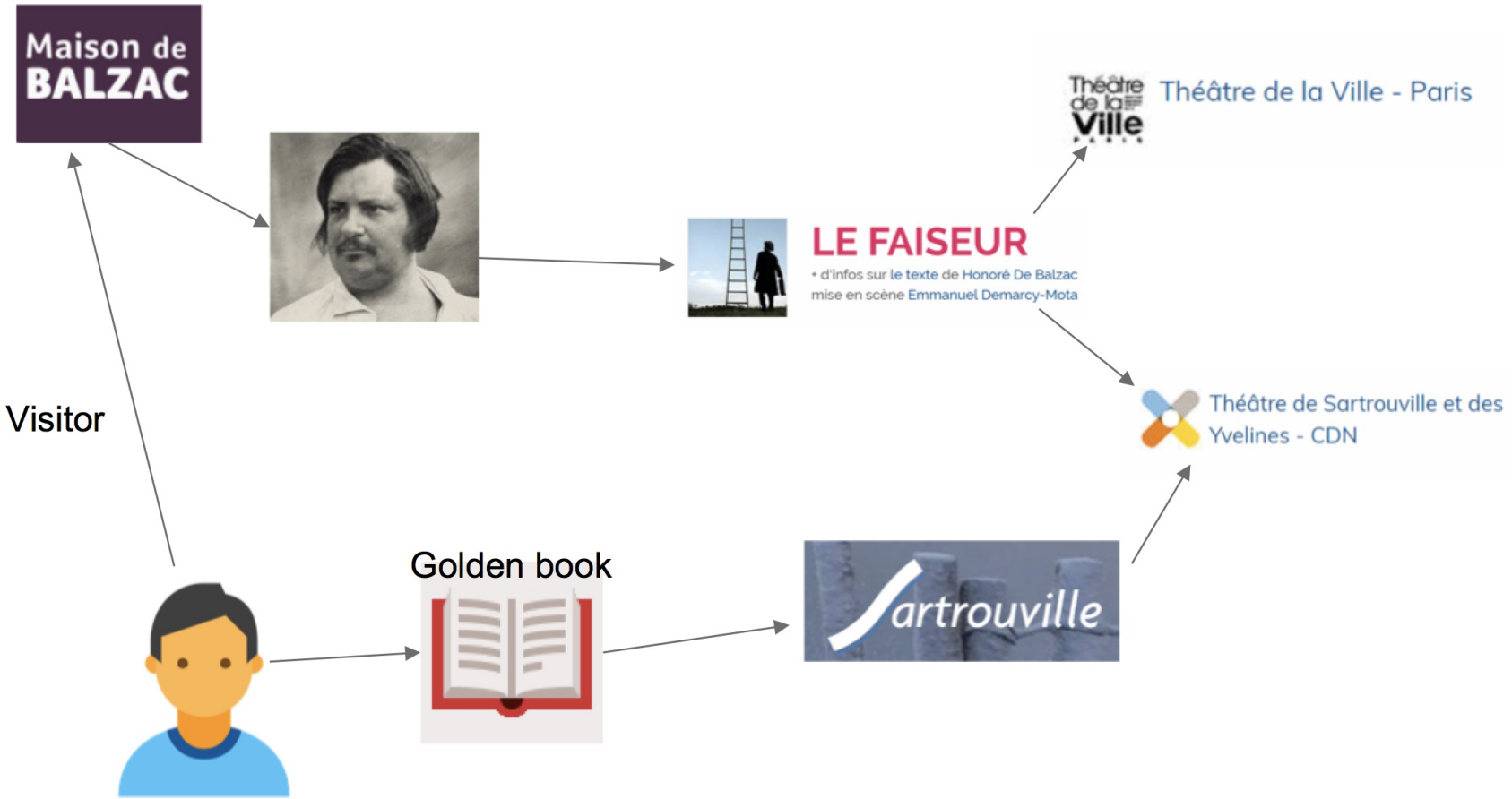
Gather data from institutions:

- Collect data respecting **privacy**
- Adopt homogeneous representations to make the data comparable
- Choose a model able to represent links between data

Rely on external data:

- *DataTourism*, tourist office data on places and events
- *OpenAgenda*, and other event calendar
- *Joconde* database, and other cultural data
- General knowledge bases: *DBPedia*, *Wikidata*, ...
- Geographical knowledge bases: *geonames*, data on *data.gouv.fr* ...

A simple example of links generation



Objectives

Questions:

- How to collect, integrate and enrich this complex and large amount of data?
- How to mine such type of data to extract useful information?

Hypothesis:

- Integrate external data source to enhance the quality of the original data;
- Limit the analysis to a specified context help boosting performance.

Approach

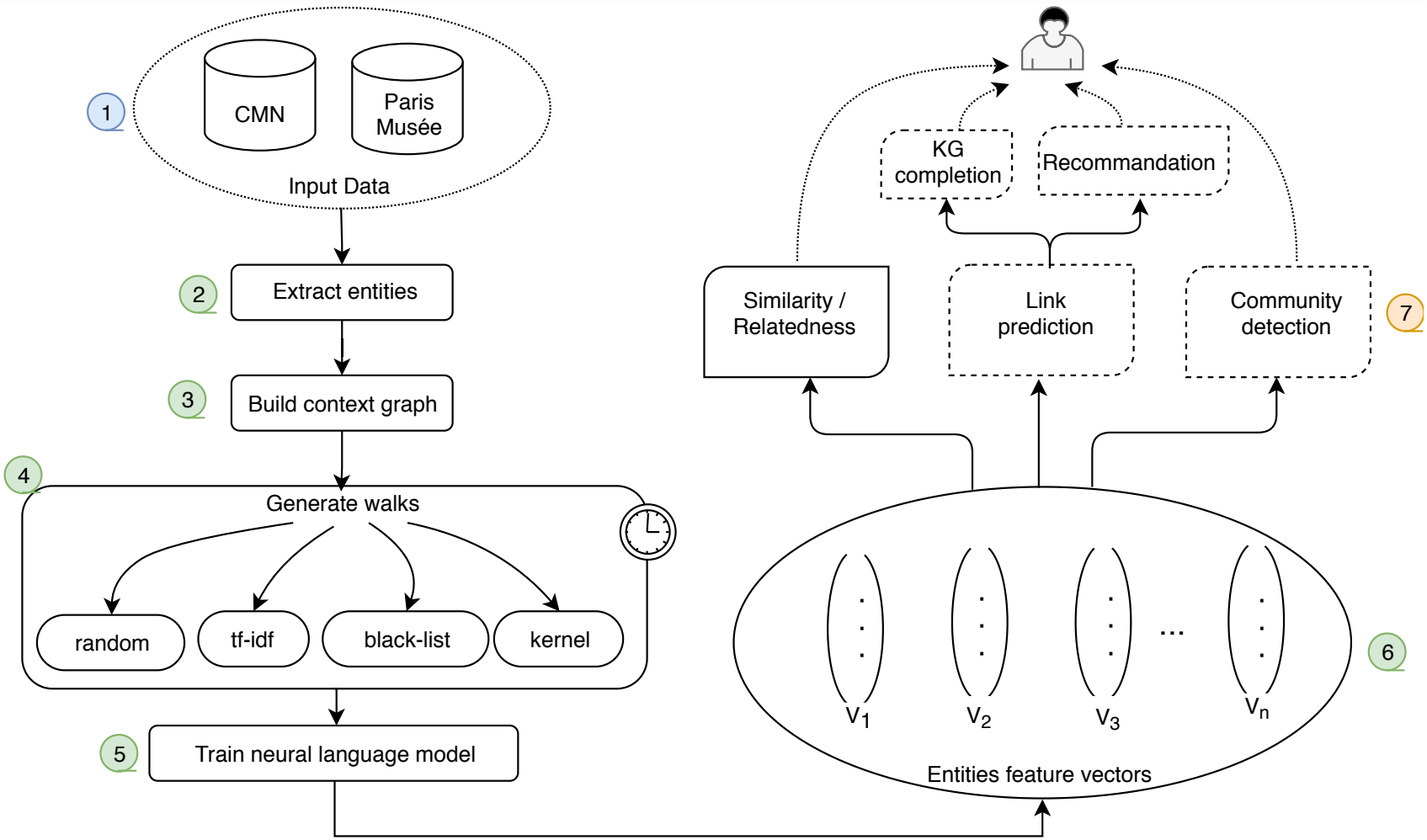
- Represent instances as a set of n-dimensional numerical feature vectors
- Use representation with different ML tasks

1. Adapt neural language model : Word2vec
2. Transform RDF graph into sequences of entities and relations (sentences)
3. Train the model and generate entity vectors

+ Conserve the information in the original graph
+ Semantically similar/related entities have close vectors in the embedded space
+ Generate a reusable model, that could be enriched with new entities

Outline

Knowledge graph embedding process



Extract entities (2)

Identify entities' URIs from input data

1. URI exist: read and identify URI from data files
2. URI ! exist: use entity name to build URI (dbpedia, frdbpedia, wikidata)

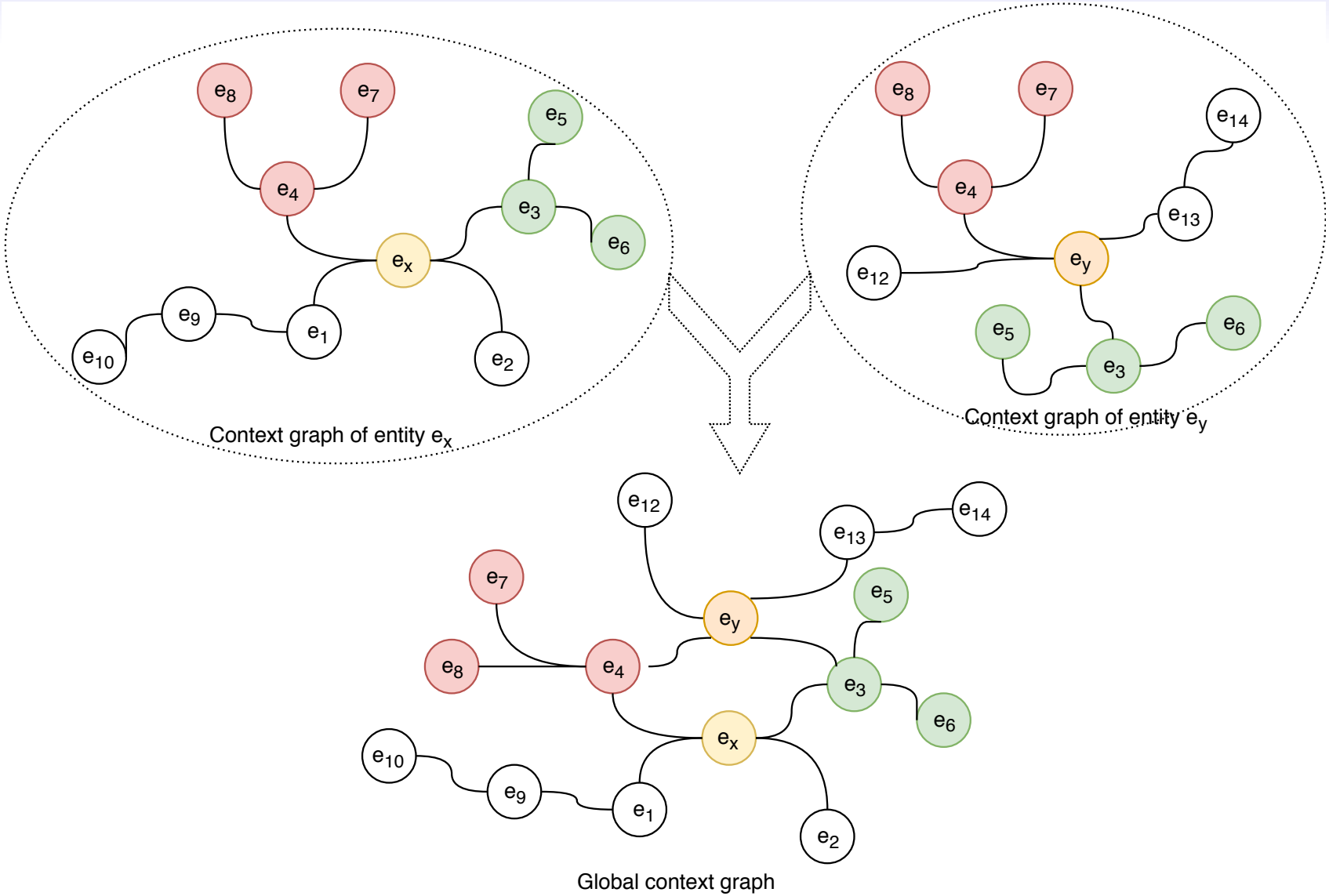

```
{
    "uris":{
        "http://fr.dbpedia.org/sparql":"http://fr.dbpedia.org/resource/Ch\u00e2teau_de_Voltaire"
    },
    "monument_nom":"Ch\u00e2teau de Voltaire \u00e0 Ferney",
    "monument_id":377,
    "monument_uri":"http://fr.dbpedia.org/page/Ch%C3%A2teau_de_Ferney-Voltaire",
    "monument_lat":"46,2581309",
    "monument_lon":"6,1043544",
    "monument_alias":[
        "Ch\u00e2teau de Voltaire",
        "Ch\u00e2teau de Voltaire",
        "Ch\u00e2teau de Voltaire",
        "Ch\u00e2teau de Voltaire"
    ],
    "monument_place":"Ferney"
},
{
    "uris":{
        "http://dbpedia.org/sparql":"http://www.wikidata.org/entity/Q13346772"
    },
    "monument_nom":"Site des m\u00e9galithes de Locmariaquer",
    "monument_id":381,
    "monument_lat":"47,5732972",
    "monument_lon":"-2,9574633",
    "monument_alias":[
        "M\u00e9galithes de Locmariaquer",
        "Cat\u00e9gorie:Site m\u00e9galithique de Locmariaquer"
    ]
},
```

Build context graph (3)

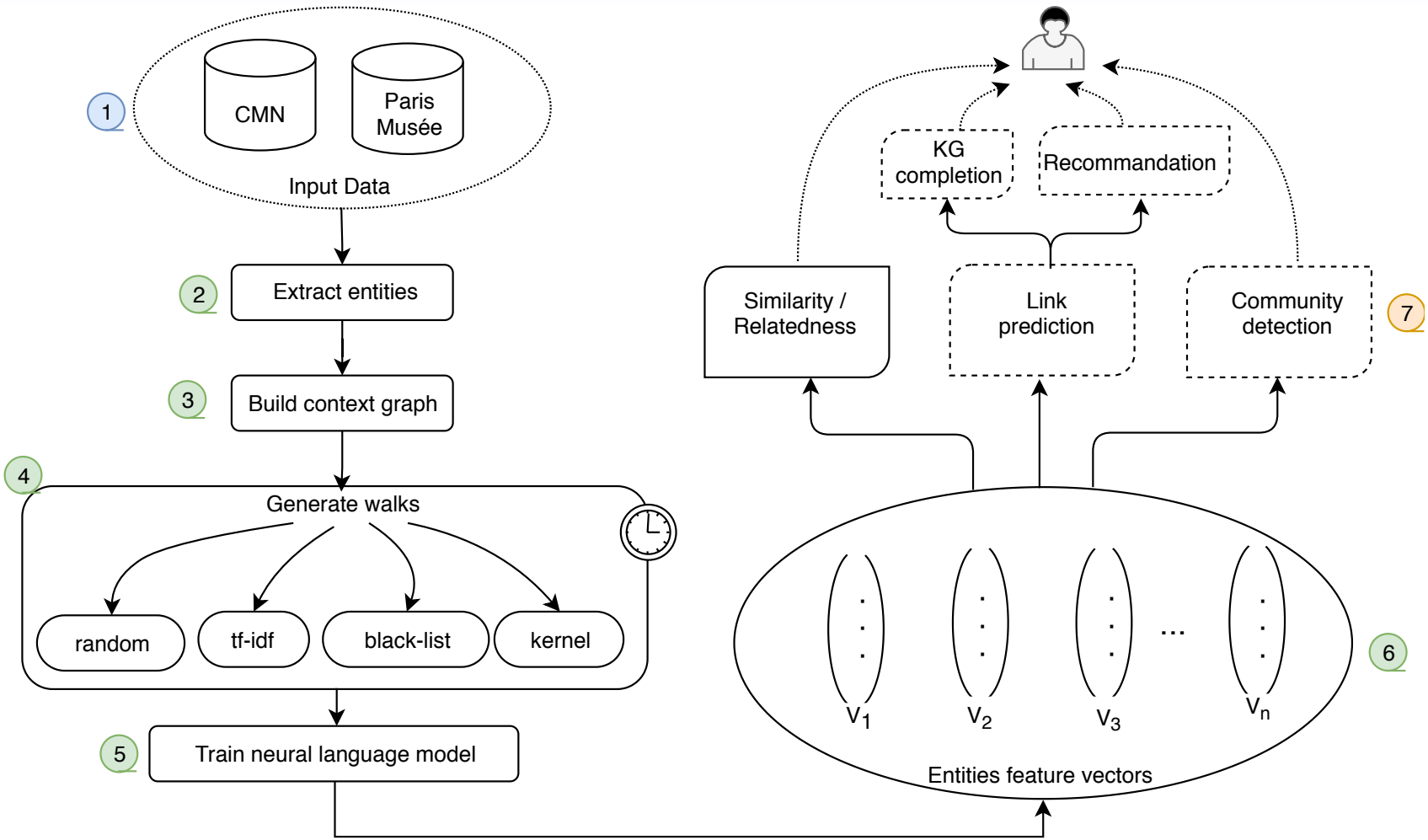
For each entity URI:

- Build context from a generalized **data source**, **'around'** the entity
- Data source: e.g. DBpedia
- 'around': get neighbours in the graph within α hops
  - Consider the undirected graph
  - α = 1 or 2
- Define a **black-list** to ignore predicates and objects:
  - very general, e.g. `<http://www.w3.org/2002/07/owl#Thing>`
  - non-informative, e.g. `<http://fr.dbpedia.org/resource/Modèle:P.>`
  - noisy, e.g. `<http://www.w3.org/2000/01/rdf-schema#comment>`

Merge context graphs (3)



Generate walks (4)



Random walk (4)

Intuition: all neighbours are equally important for an entity

- Specify walk parameters
  - nb-walks: number of walks (example: 500 walk)
  - depth: number of hops in the graph (2, 4, 8)
  - example: d=4 $\Rightarrow e \rightarrow p1 \rightarrow e1 \rightarrow p2 \rightarrow e2$
- Specify the list of entities (all entities in the global context graph / a predefined list)
- For each entity:
  1. get a random list of direct neighbours
  2. calculate the corresponding number of walks for each neighbour
  3. recursively..
- Adjust the number of walks according to specific cases:
  - if (nb-neighbours $<$ nb-walks) : divide, get the entire part of the division, sum-up the rest and add it to a randomly selected neighbour
  - if (nb-neighbours == 0) : transfer its nb-walks to another randomly selected neighbour

Tf-Idf graph walk (4)

Intuition:

Some neighbours are more important for an entity.
Prioritize important neighbours by weighting their predicates.

- Calculate **tf-idf** weights for predicates
- *tf*: evaluate the importance of a predicate p for an entity e
  - $t_o(p, e) =$ number of p occurrences for entity e
  - $t_p(e) =$ number of predicates associated with e
  - $tf(p, e) = t_o(p, e)/t_p(e)$
- *idf*: evaluate the importance of a predicate p on the whole graph
  - $D =$ number of entities in the graph
  - $d(p) =$ number of entities using predicate p
  - $idf(p) = \log(D/d(p))$
- $tfidf(p, e) = tf(p, e) * idf(p)$

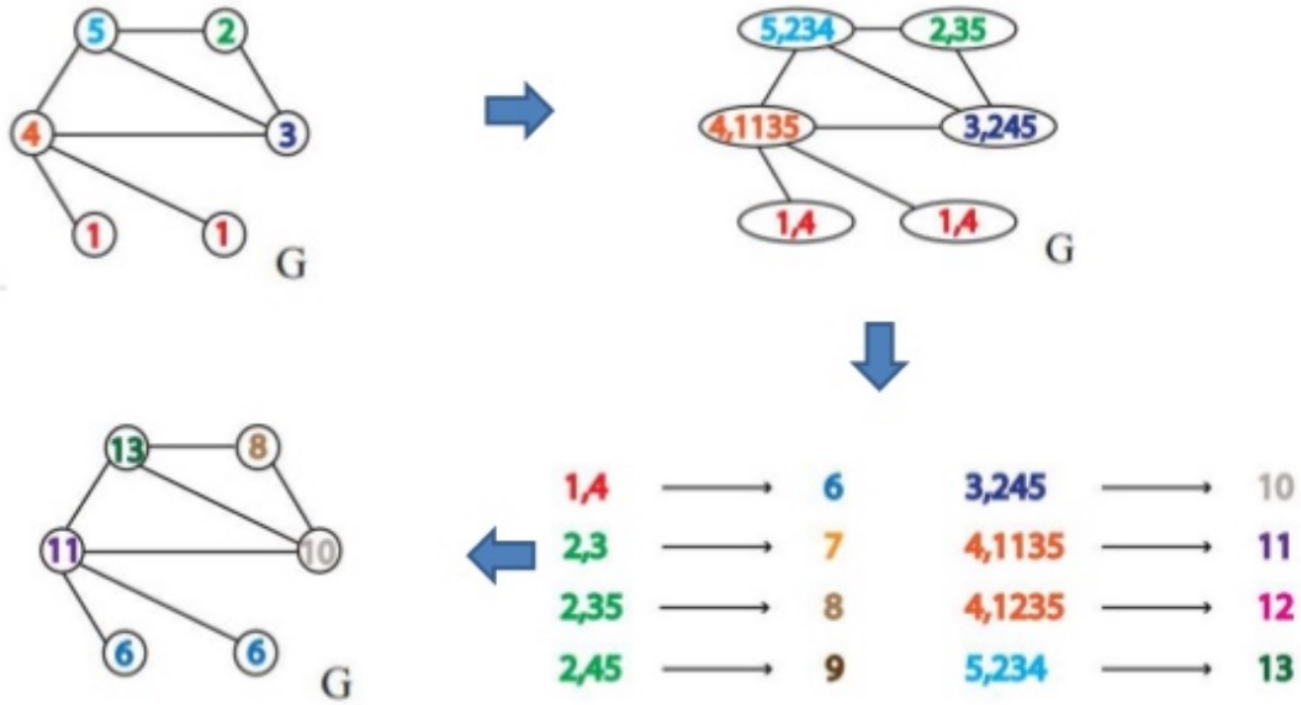
Black-list walk (4)

Intuition: some predicates are noisy (less important) for an entity

- Put weights on predicates:
  - predicate in the black-list: *weight* $= 0$ (to ignore)
  - other predicate: *weight* $= 1$ (to consider in the walk)
- Example:
  {`http://dbpedia.org/ontology/wikiPageWikiLink`}

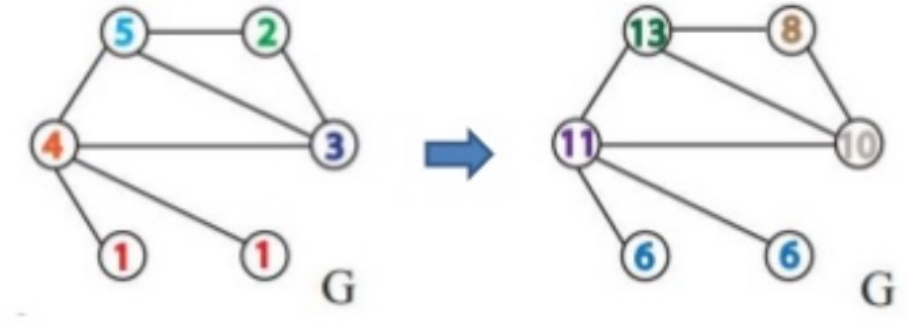
Weisfeiler-Lehman kernel (4)

Intuition: Weisfeiler-Lehman subtree RDF graph kernels capture (richer) information of an entire subtree in a single node.



de Vries, Gerben K. D., "A Fast Approximation of the Weisfeiler-Lehman Graph Kernel for RDF Data", ECML PKDD 2013.

Weisfeiler-Lehman kernel (4)

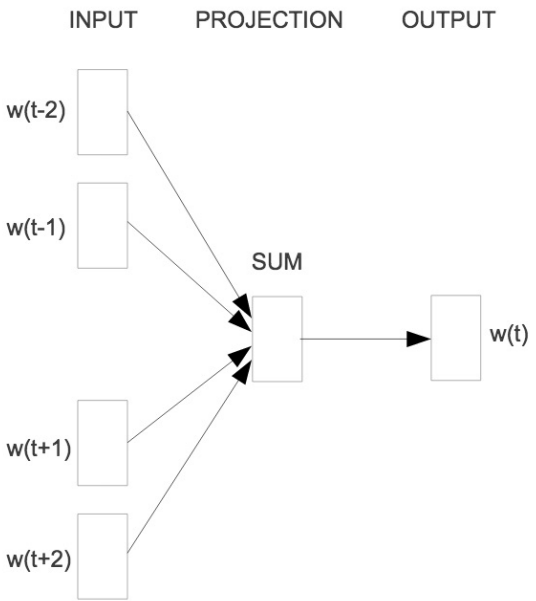


- For each iteration, for each entity in the graph, get random walks of depth d
- After 1 iteration, graph G sequences:
- $1->6->11; 1->6->11->13; 1->6->11->10;$ ...
- $4->11->6; 4->11->13; 4->11->10;$
  $4->11->10->8;$ ...

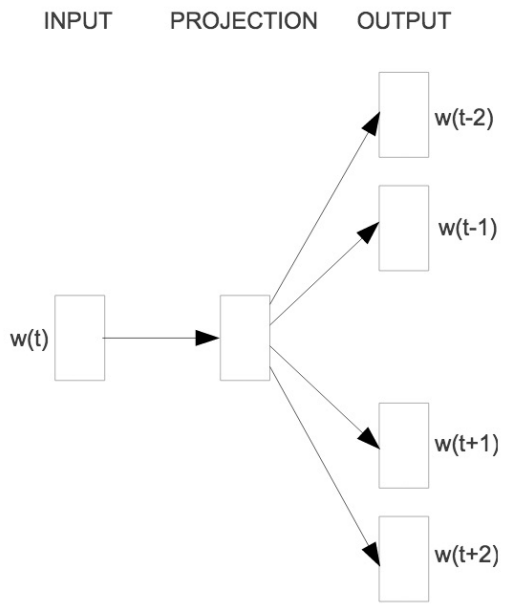
Ristoski, Paulheim, "RDF2Vec: RDF Graph Embeddings for Data Mining", ISWC 2016.

Neural language model (5,6)

- Word2vec
- A two-layer neural net that processes text
- Input: a text corpus (sentences)
- Output: a set of vectors (feature vectors for words in that corpus)
- Create neural embeddings for any group of discrete and co-occurring states $\rightarrow$ RDF data



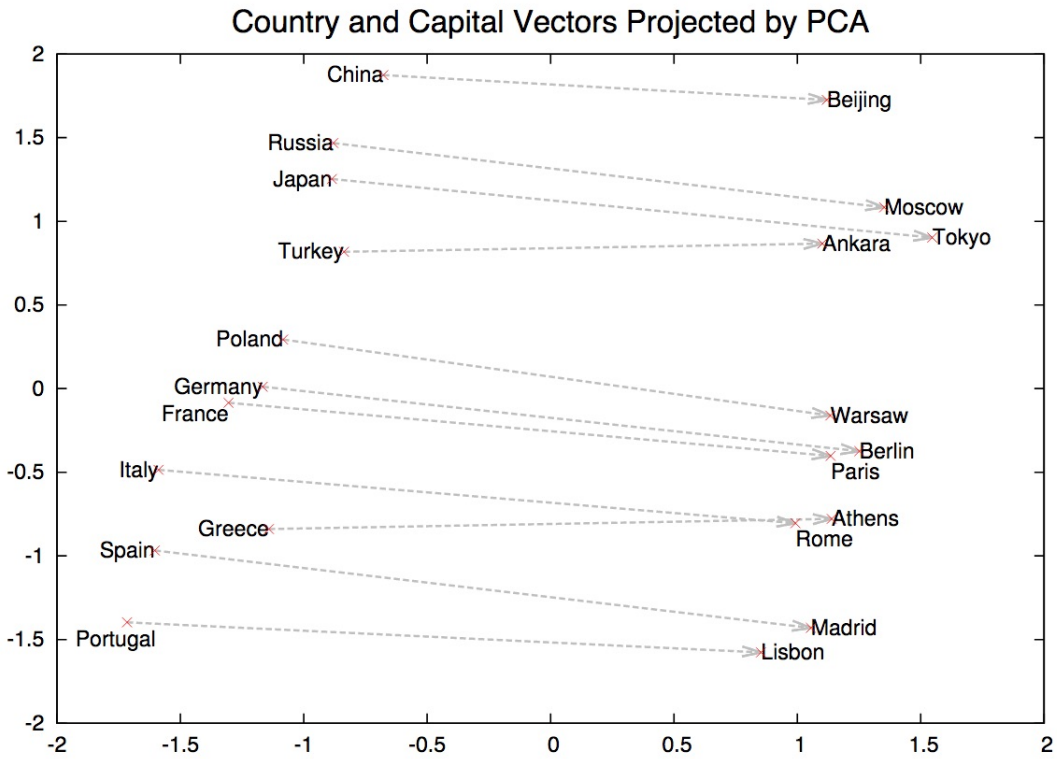
CBOW



Skip-gram

Neural language model (5,6)

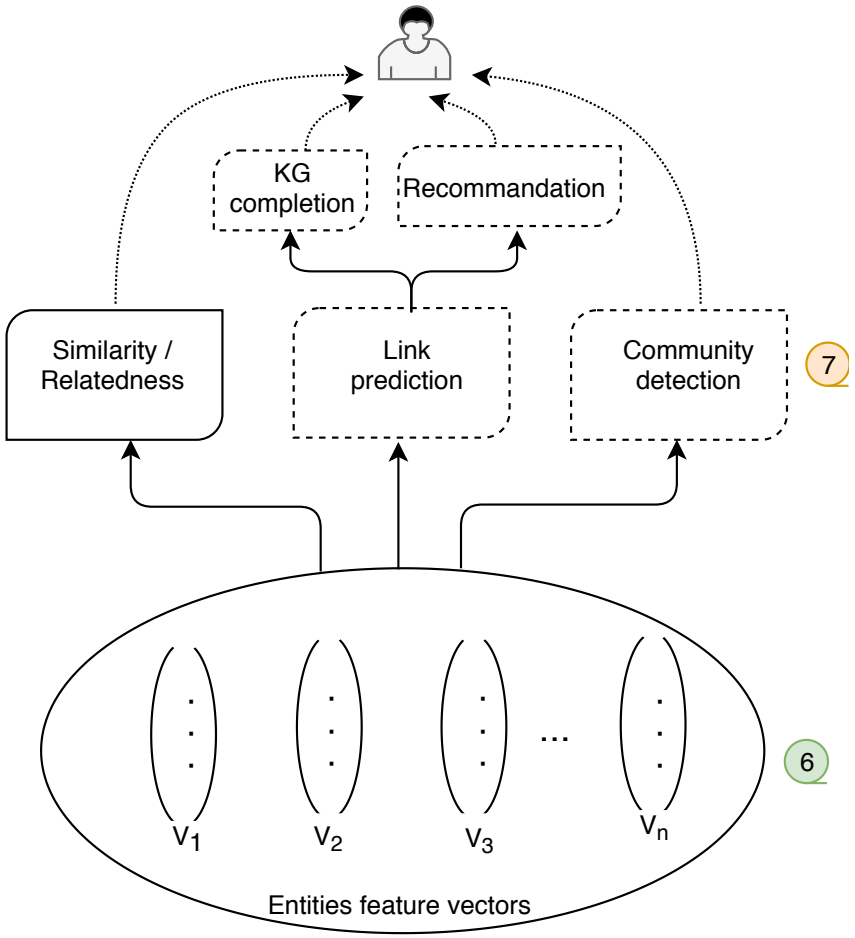
- Similar words cluster together in the embedding space
- Operations on vectors:
  - *Madrid - Spain = Beijing - China*
  - *Madrid - Spain + China = Beijing*



Mikolov,Tomas et al., "Distributed Representations of Words and Phrases and their Compositionality", NIPS 2013.

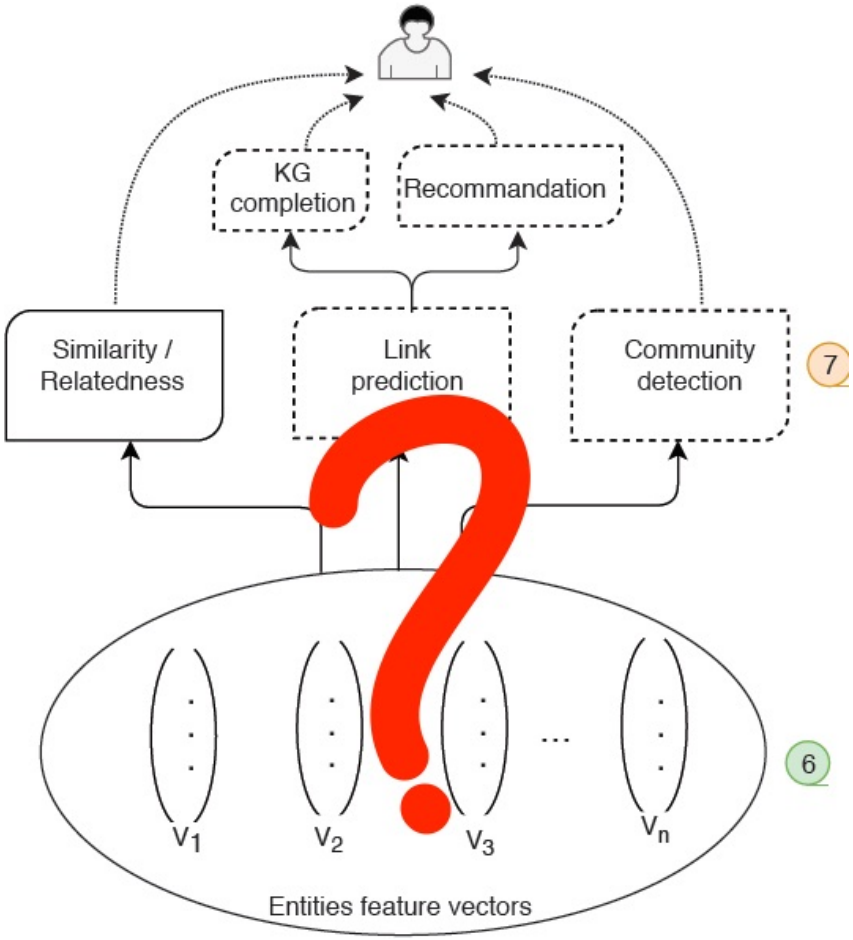
Using the model (7)

- N-dimensional numerical vector representation of entities
- $V_e = (v_1, v_2, ..., v_i, ..., v_n)$



Using the model (7)

- N-dimensional numerical vector representation of entities
- $V_e = (v_1, v_2, ..., v_i, ..., v_n)$



Outline

Entity similarity and relatedness

The outcome of our first experiments:

1. We discover **hidden interesting information** (non-trivial things) $\rightarrow$ can lead to **new knowledge** (facts)
2. We find rather trivial things, but nothing wrong

Task

Try to understand and interpret the semantic relation behind the similarity/relatedness measure returned by the model

- For strong similarities between two entities:
  - find the shortest path between them
  - the shortest random walks used with these entities and that connect them
- This path could give a form of ”explanation” of their connection

Similarity examples

Get top similar entities to 'dbr:Abbaye-de-Charroux'

1. 'dbr:Benest' (0.9997649192810059)
   - in fact, 'Benest' is close to 'Abbaye-de-Charroux'
   - a 'general' direct link exist: 'dbo:wikiPageWikiLink'
   - could be found by another walk : 'long' and 'lat'
   - specify this general type of link : e.g. 'Benest -> is-close-to -> Abbaye-de-Charroux'
2. 'dbr:Abbaye-Saint-Sauveur-de-Charroux' (0.999472081661224 4)
   - a 'general' direct link exist: 'dbo:wikiPageRedirects'
   - 'Abbaye-Saint-Sauveur-de-Charroux -> same-as -> Abbaye-de-Charroux'
3. 'dbr:Baudri-de-Bourgueil' (0.9998940825462341)
   - Charroux is a Benedictine abbey
   - Baudri is a religious of the order of benediction that has greatly changed the monastic practice
   - A non-trivial link to analyse...

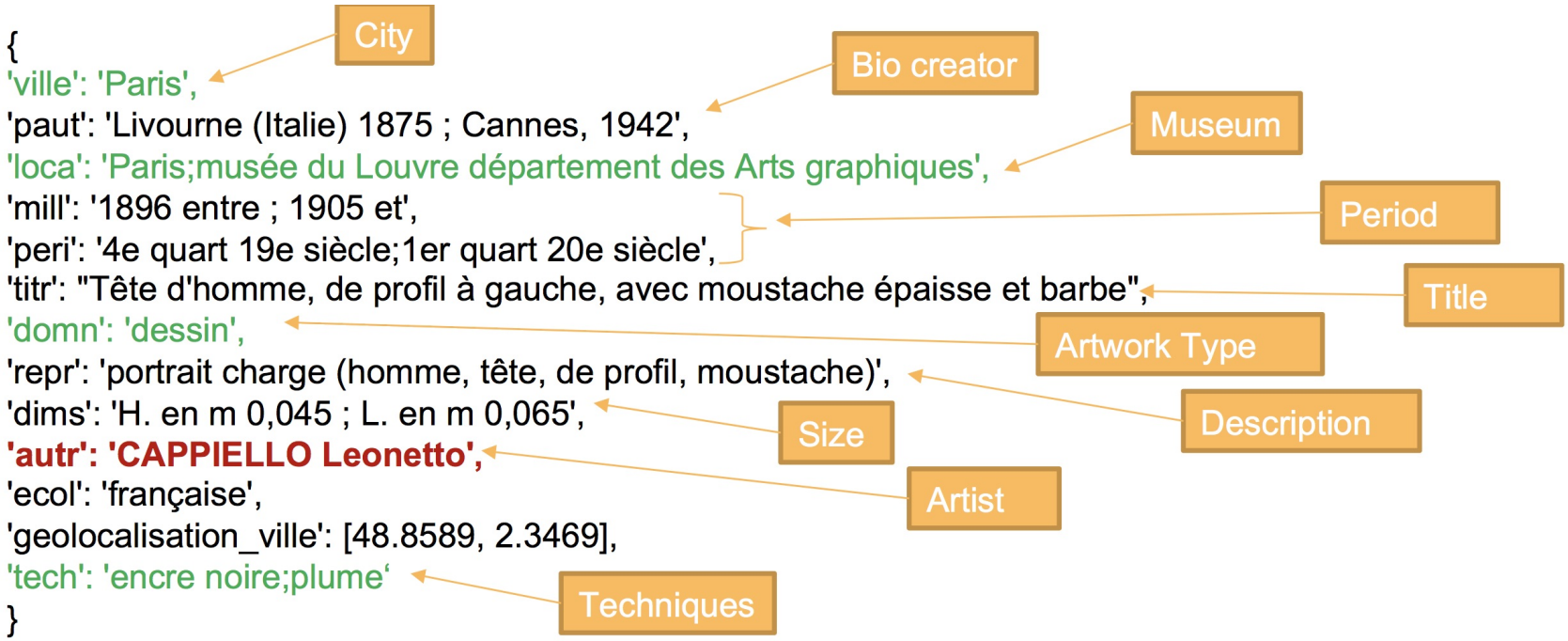
Similarity examples

An event about 'Beethoven' is organised in 'Musée Bourdelle' → Transpose this event to other museums ?

???? is to 'Musée Balzac' what 'Beethoven' is to 'Musée Bourdelle'

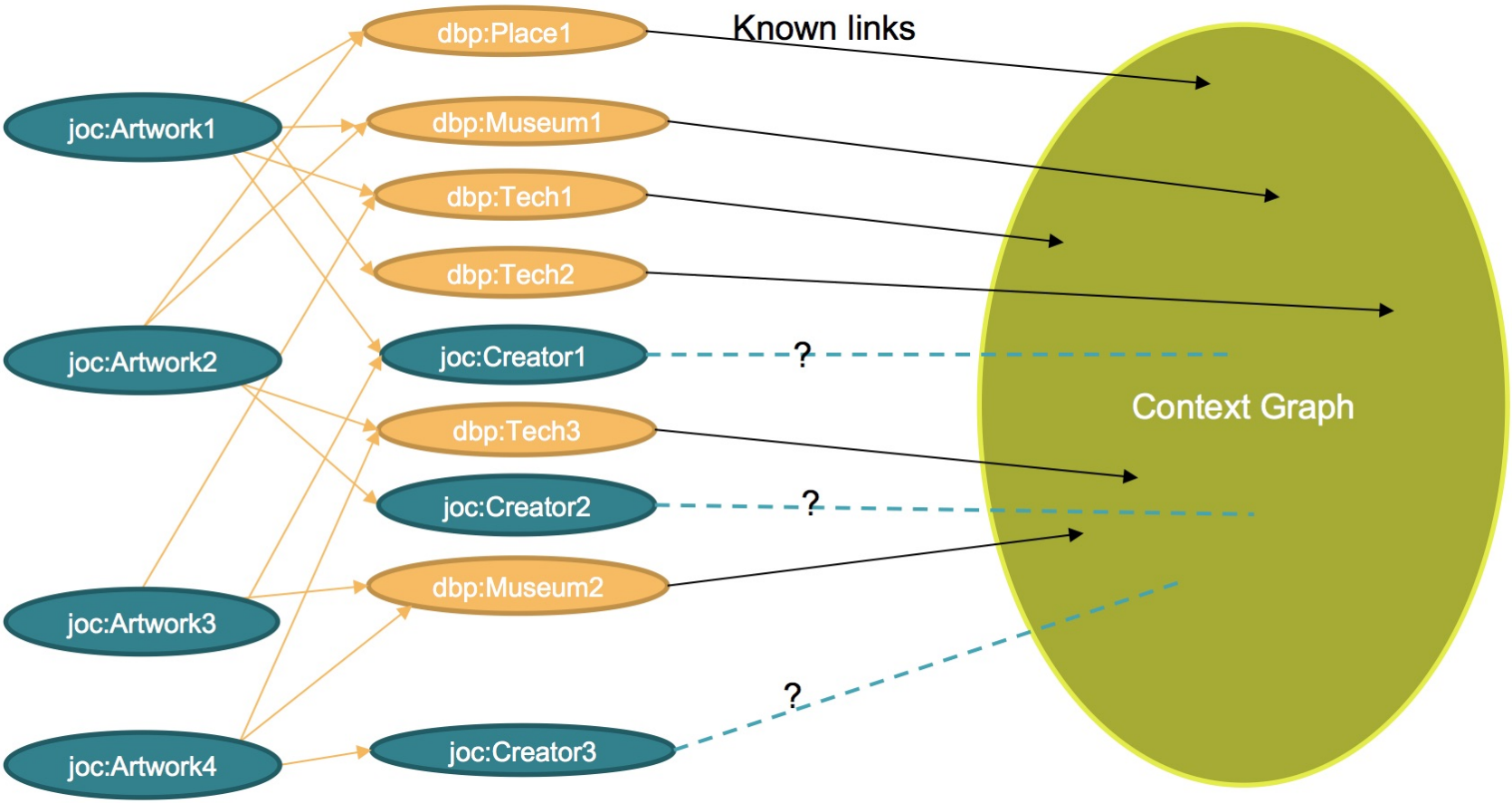
1. 'Romeo-Void', 0.827353477478027 3
2. 'Era-(musical-project)', 0.8242164850234985,
3. 'Spectrum-(band)', 0.8137580156326294,
4. 'Oladad', 0.8116711378097534,
5. 'Time-Crash-(band)', 0.8114833831787109,
6. 'Ellegarden', 0.810987114906311,
7. 'John-Mayer-Trio', 0.8100252151489258,
8. 'Motion-Trio', 0.8094779253005981,
9. 'The-Bala-Brothers', 0.806811392307281 5

Entity matching: Joconde data



Joconde: $\approx$ 600000 artworks, $\approx$ 10000 techniques, $\approx$ 1000 places (museums...), $\approx$ 60000 creators

Entity matching: idea



Entity matching: idea

Manual linking	Artworks covered
30 creators	153985
30 museums	376480
16 techniques	534324*
13 domains	662829*

* Some artworks are in several domains or relatives to several techniques

- Manual linking is a starting point to enrich the links between data from Joconde and our Context Graph
- Hypothesis: helps to get better walks between entities in Joconde and the Context Graph

Outline

Conclusion

- Integrate cultural data from different heterogeneous sources
- Use an adaptation of a neural language model for entity embedding
- Build numerical model that can serve to calculate similarities
- The output of the model can be used with different ML tasks

Thank you