Karlsruhe Institute of Technology

AIFB

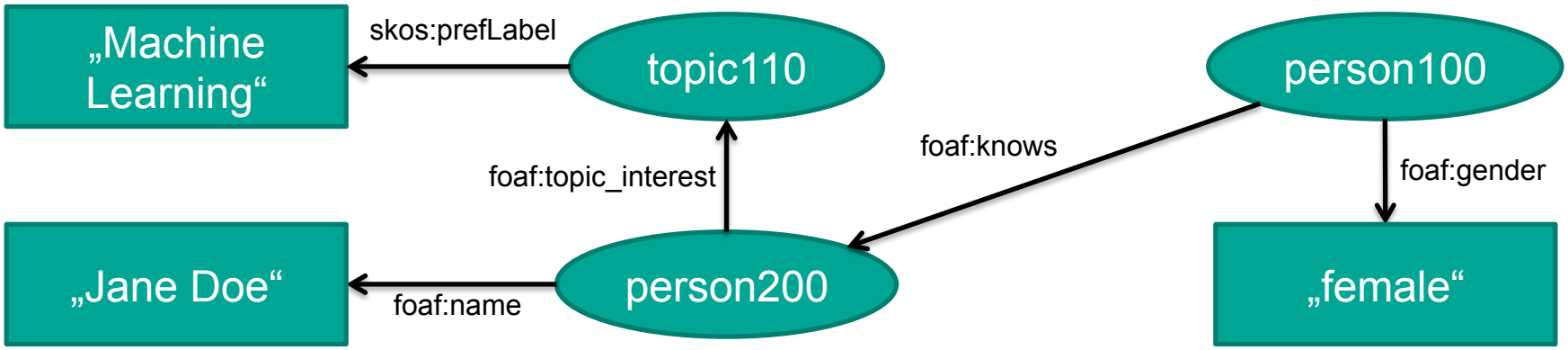
Uta Lösch - Stephan Bloehdorn - Achim Rettinger*

GRAPH KERNELS FOR RDF DATA

KNOWLEDGE MANAGEMENT GROUP
INSTITUTE OF APPLIED INFORMATICS AND FORMAL DESCRIPTION METHODS (AIFB)

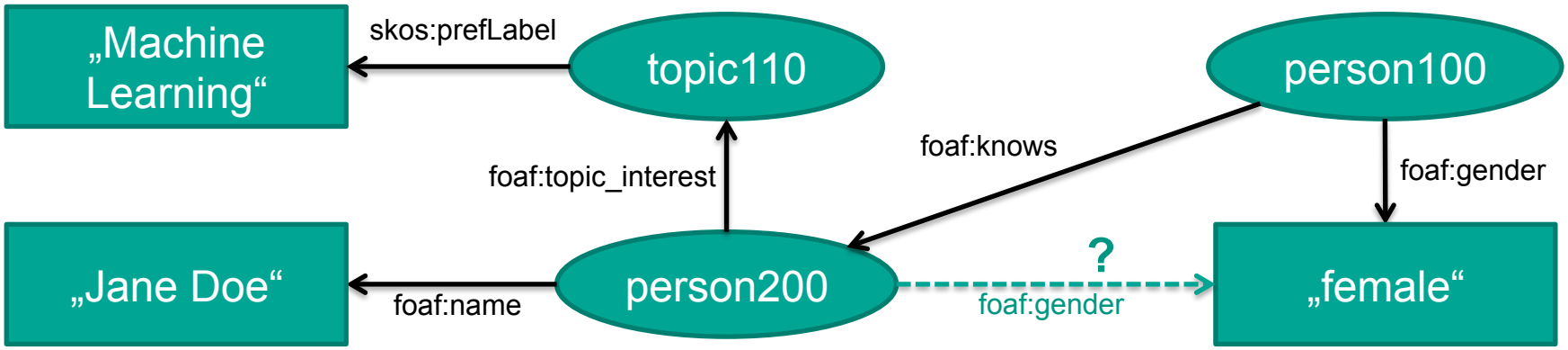
The Vision

Given any data in RDF format…



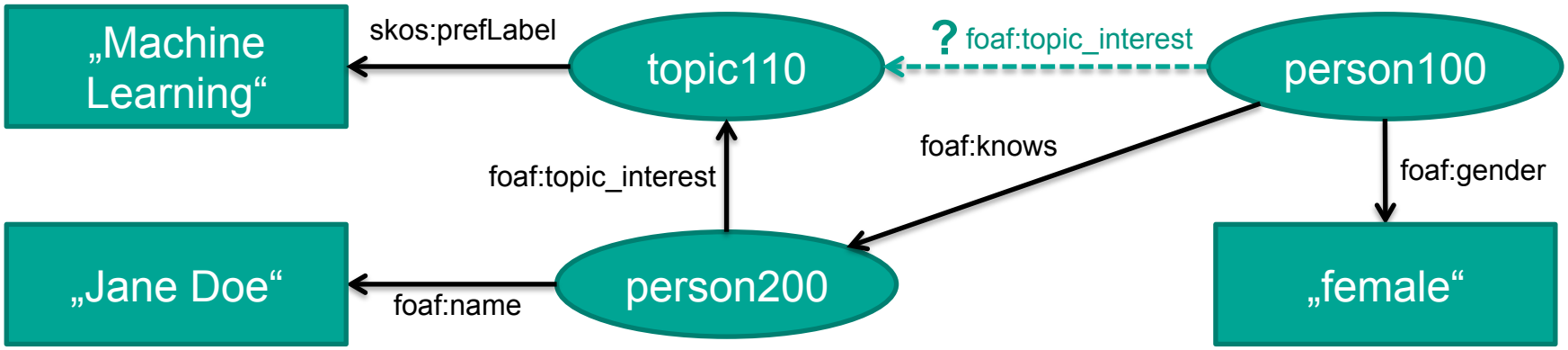
…solve any standard statistical relational learning task, like…

The Learning Tasks (I)



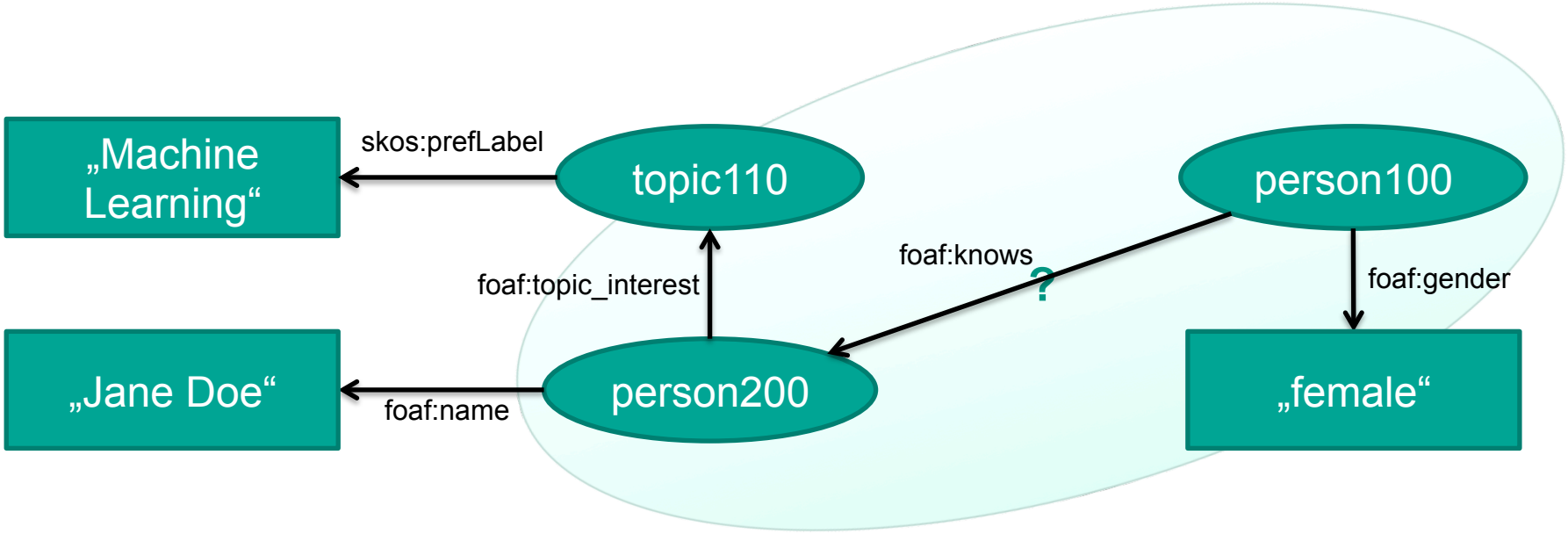
… property value prediction, …

The Learning Tasks (II)



… link prediction, …

The Learning Tasks (III)



… clustering,…

… or class-membership prediction, entity resolution, ...

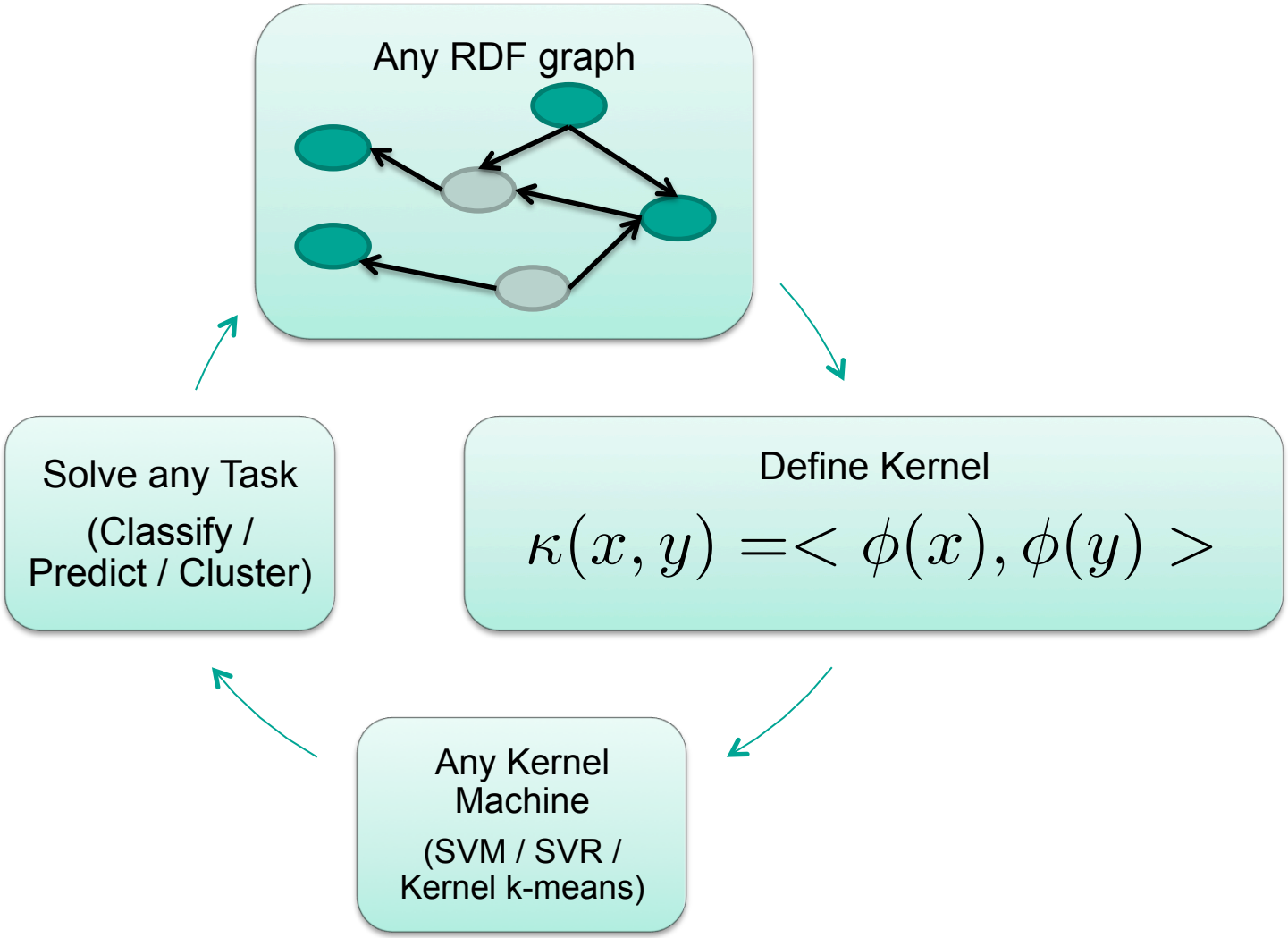
The constraints

… while

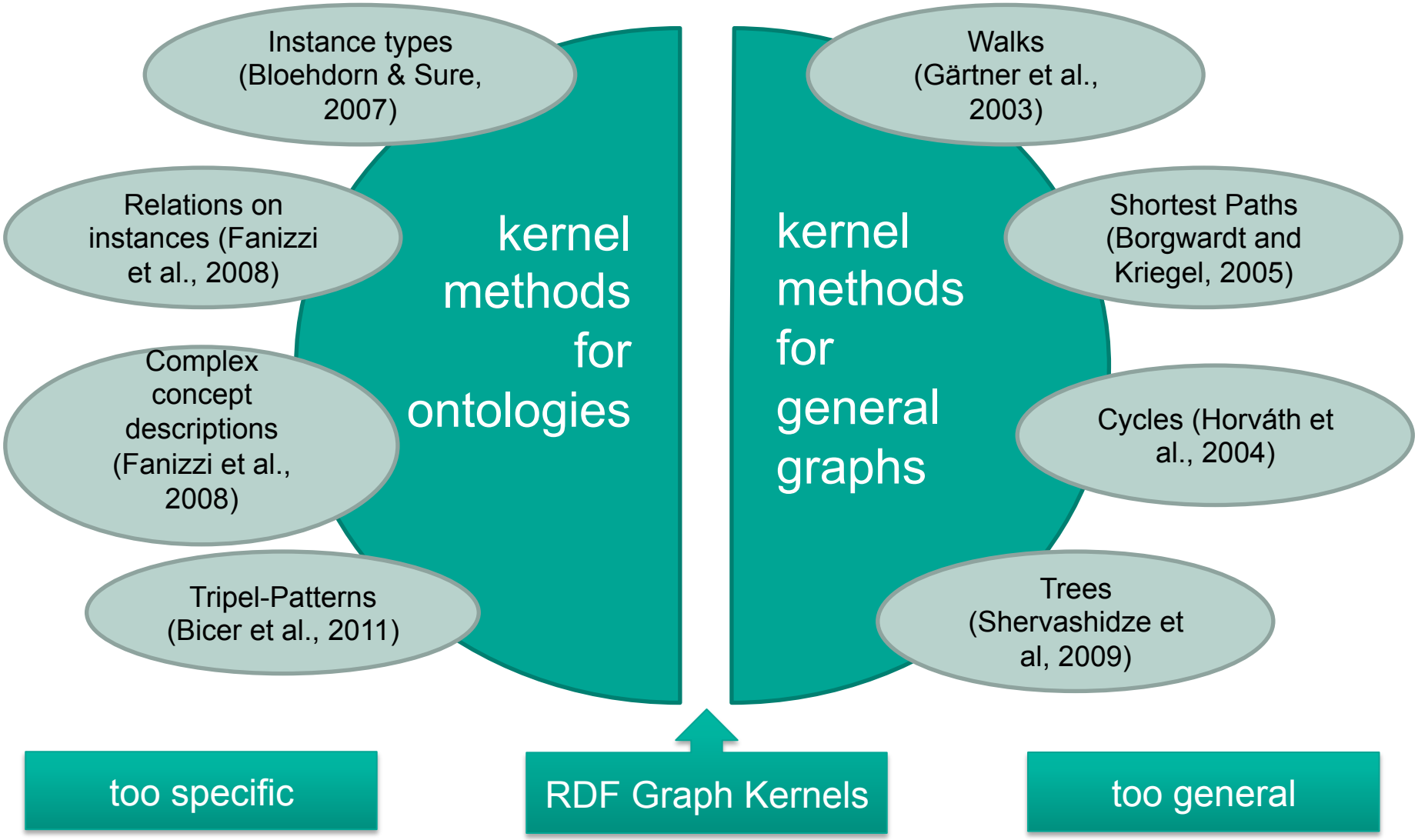
- using readily available learning algorithms
- exploiting specifics of RDF graphs
- relying on the graph structure and labels only
- avoiding manual effort as much as possible.

RELATED WORK

The (good old) Kernel Trick



The Gap


Instance types (Bloehdorn & Sure, 2007)
Relations on instances (Fanizzi et al., 2008)
Complex concept descriptions (Fanizzi et al., 2008)
Tripel-Patterns (Bicer et al., 2011)
kernel methods for ontologies
Walks (Gärtner et al., 2003)
Shortest Paths (Borgwardt and Kriegel, 2005)
Cycles (Horváth et al., 2004)
Trees (Shervashidze et al, 2009)
kernel methods for general graphs
too specific
RDF Graph Kernels
too general

The Goal

Define kernel functions, which

- can be used with ANY kernel machine,
- can handle ANY RDF graph,
- exploit the specifics of RDF.

Overview

Motivation

Related Work

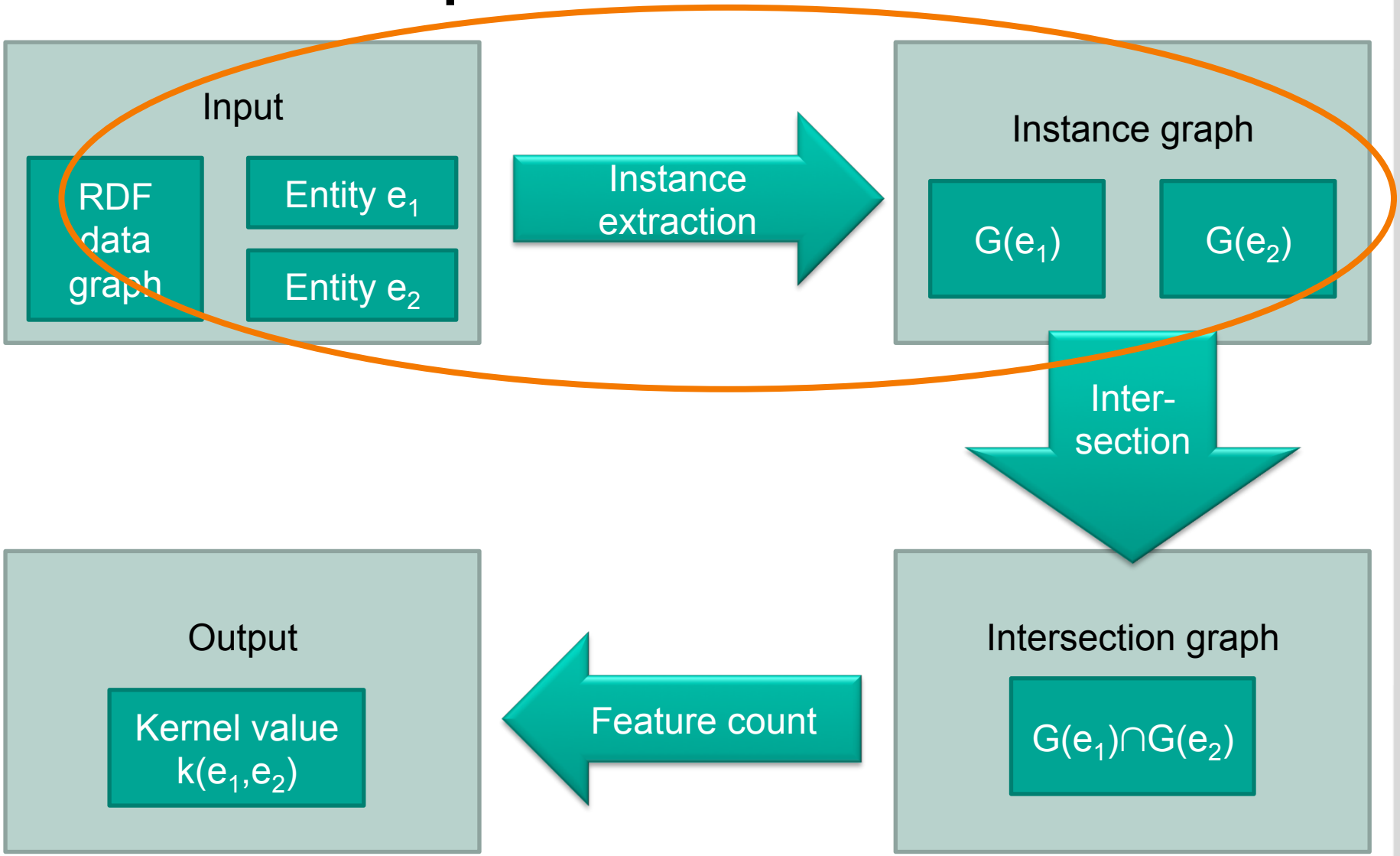
Proposed family of RDF kernel functions based on

- Intersection Graphs
- Intersection Trees

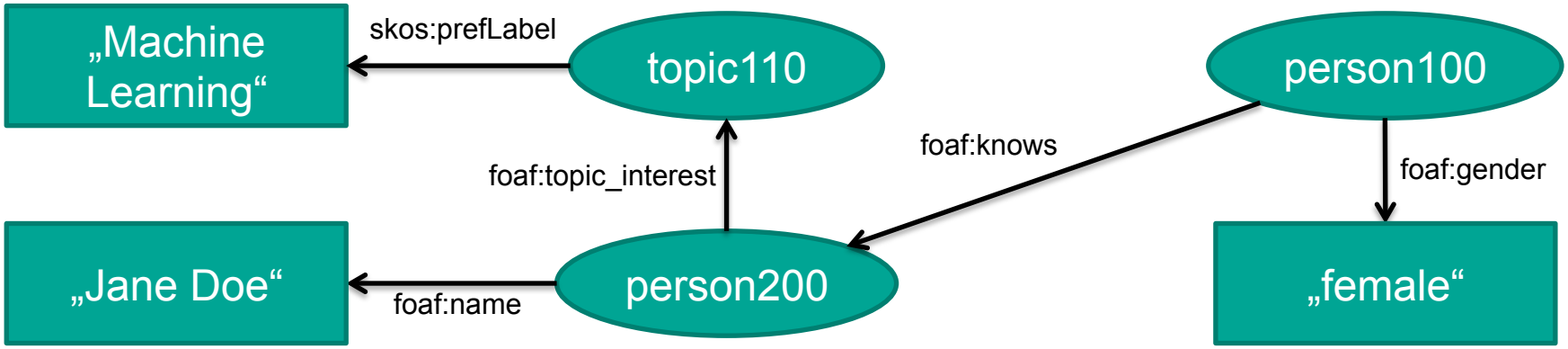
Empirical evaluation on

- Property Value Prediction
- Link Prediction

Intersection Graph

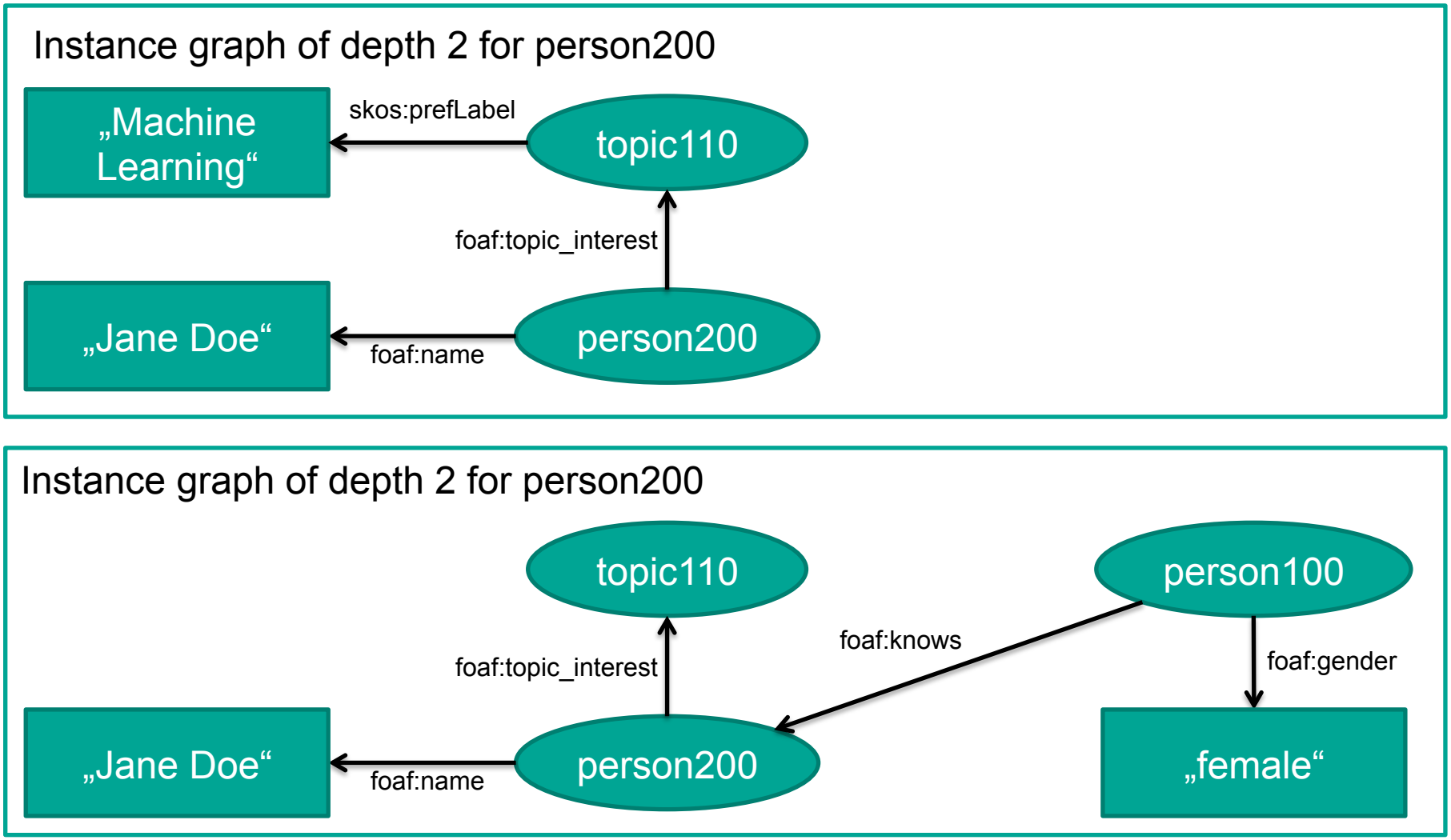


Instance graph

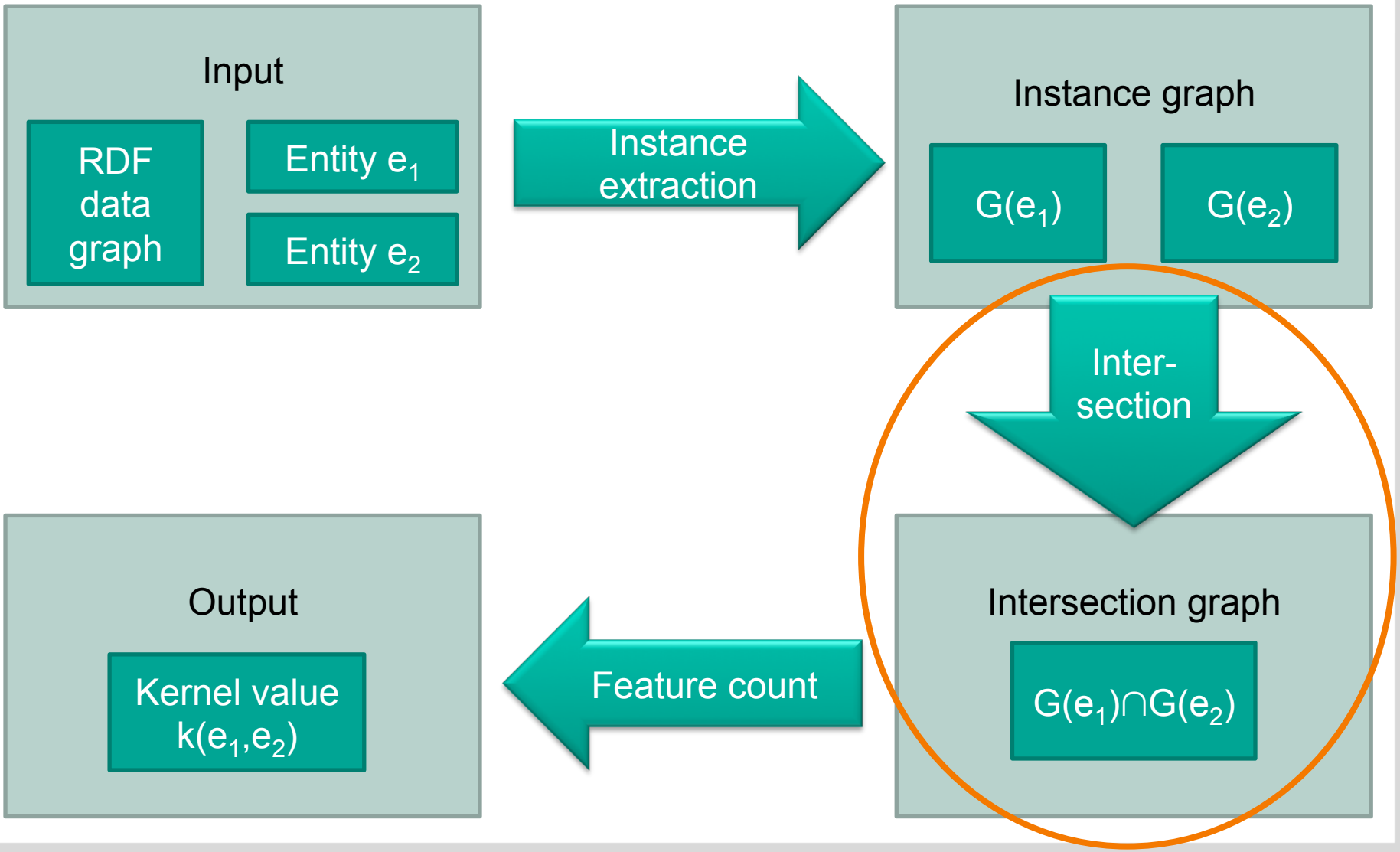


- Instance graph: *k*-hop-neighbourhood of entity *e*
  - Explore graph starting from entity *e* up to a depth *k*

Instance graph - Example



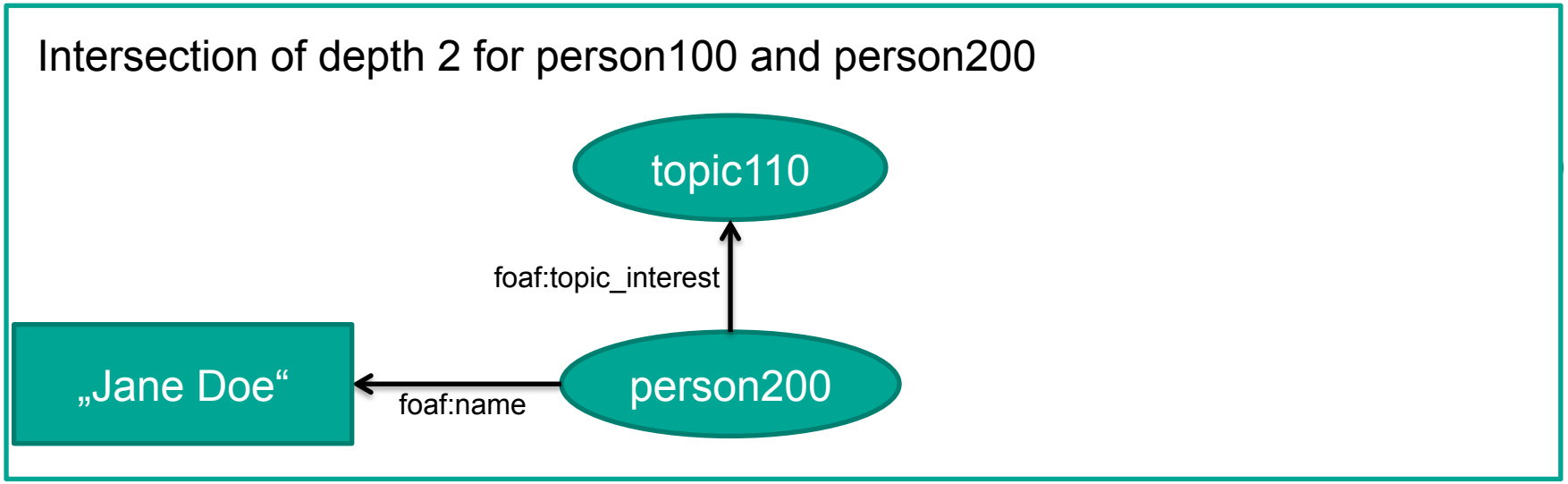
Intersection Graph



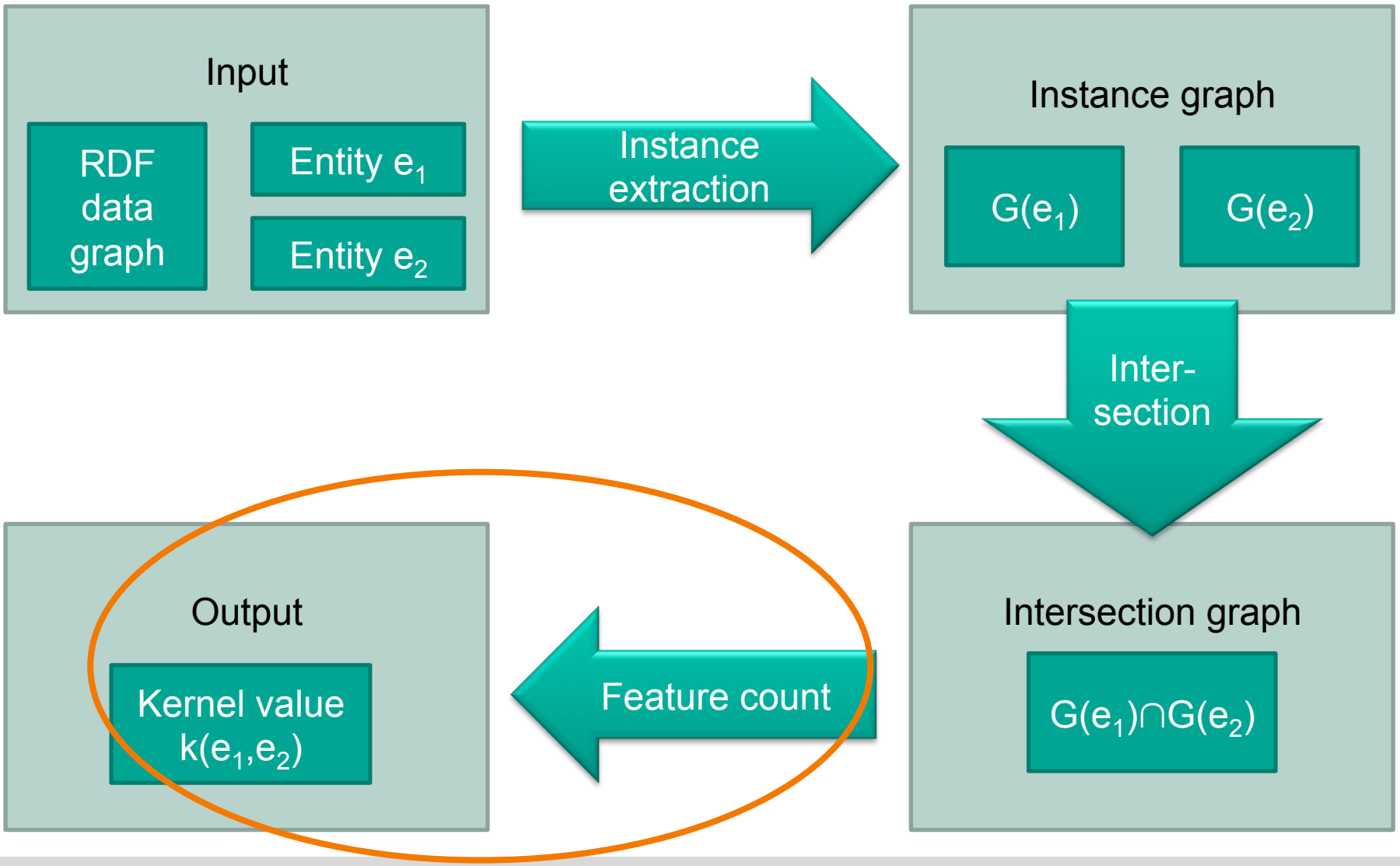
Intersection graph

- Intersection graph of graphs $G(e_1)$ and $G(e_2)$:

$$V(G_1 \cap G_2) = V_1 \cap V_2$$
$$E(G_1 \cap G_2) = \{(v_1, p, v_2) | (v_1, p, v_2) \in E_1 \wedge (v_1, p, v_2) \in E_2\}$$



Intersection Graph



Feature count

- Kernel function: Count specific substructures of the intersection graph.
- Any set of *edge-induced subgraphs…*

$$E' \subseteq E$$
$$V' = \{v \mid \exists u, p : (u, p, v) \in E' \vee (v, p, u) \in E'\}$$

- …qualifies as a candidate feature set
  - Edges
  - Walks/Paths up to a length of an arbitrary *l*
  - Connected edge-induced subgraphs

Intersection Trees

- Problem:
  - Intersection graph needs explicit calculation of instance graphs and the intersection.
  - Computationally expensive
- Intersection Tree:
  - Alternative representation of common structures
  - Can be extracted directly from the RDF graph
  - Tree structure
    - Synchronized exploration starting from both entities
    - Common elements are part of the intersection tree
    - *e1* and *e2* need special treatment

Overview

Motivation

Related Work

Proposed family of RDF kernel functions based on

- Intersection Graphs
- Intersection Trees

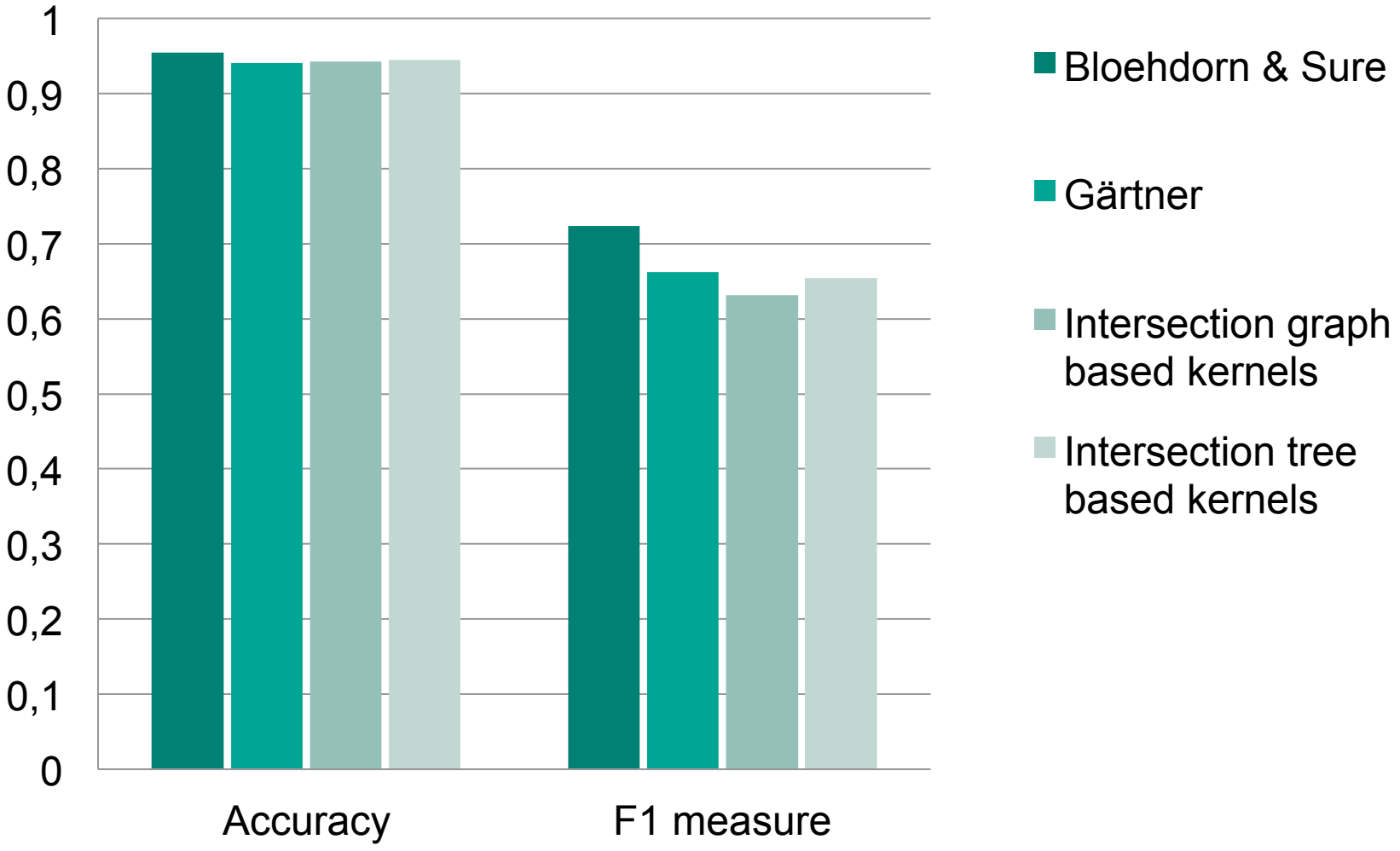
Empirical evaluation on

- Property Value Prediction
- Link Prediction

Property Value Prediction

- Multilabel learning problem (SVM)
- Data set: SWRC
  - Contains persons, publications, research topics, research groups, and projects.
  - 2547 entities, 1058 persons
  - Task: Predict if a person is member of a research group
- Classification model:
  - 1 classifier per class (research group)
  - Evaluation measure is averaged over classifiers
  - Leave-one-out-Cross-Validation

Property Value Prediction Results



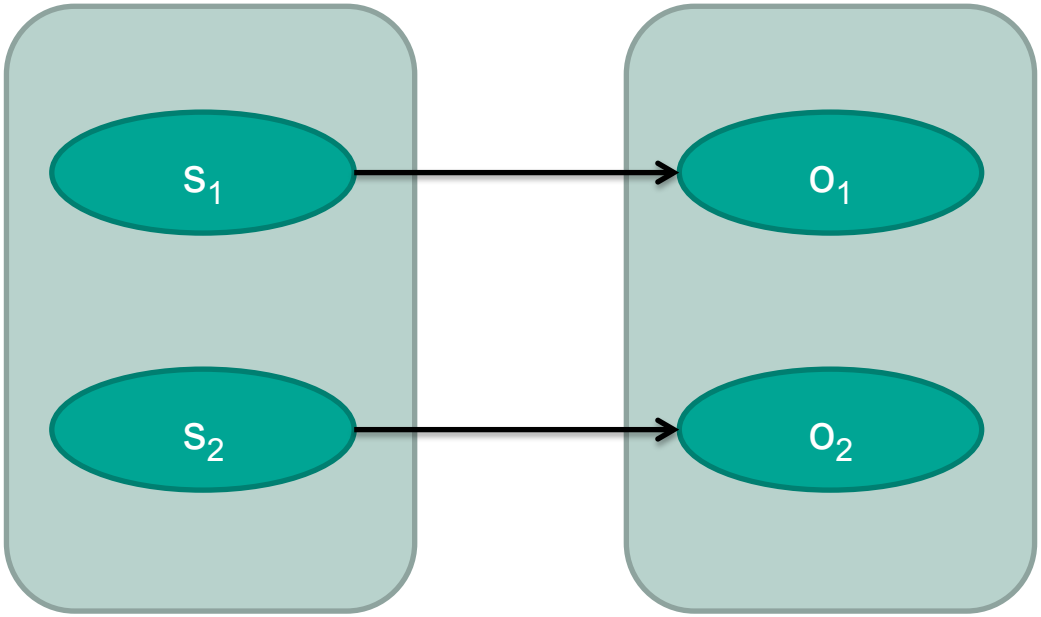
Link Prediction

- Predict links between entities (SVM)
  - Data set: Friend-of-a-friend (FOAF)
    - Gathered from Livejournal.com
    - 3040 entities, description of 638 persons, 8069 instances of the foaf:knows relation
  - Task: Predict unknown foaf:knows-relations

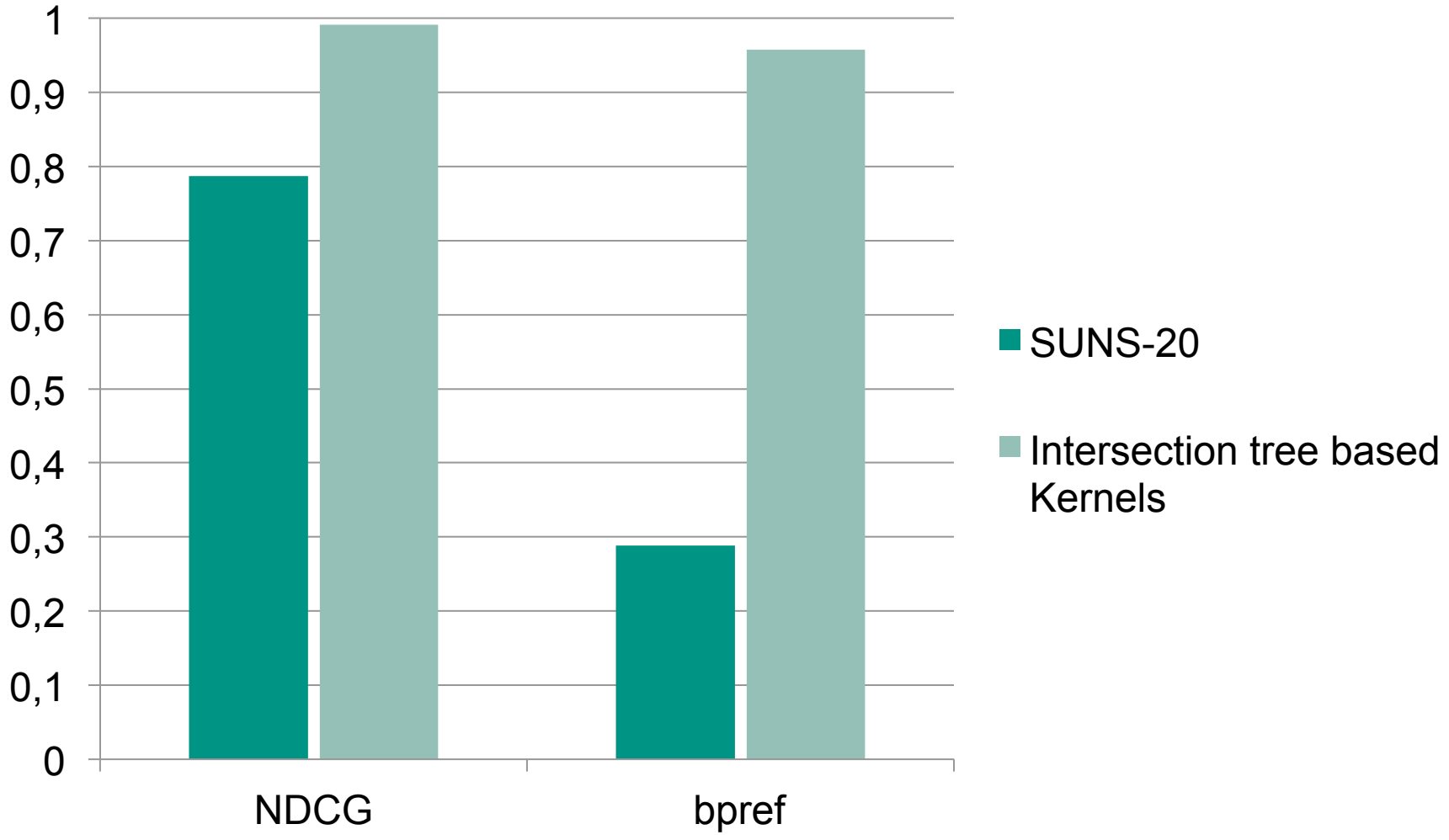
- Classification model
  - Predict likelihood that the relation foaf:knows exists between a pair of entities

Classification model

$$\kappa((s_1, o_1),(s_2, o_2)) = \alpha\kappa_s(s_1, s_2) + \beta\kappa_o(o_1, o_2)$$



Link Prediction Results



Summary

We introduced two families of kernel functions for RDF graphs, which

- can be used with ANY kernel machine
- and might solve ANY associated learning task.

- They can be applied to ANY RDF graph
- while exploiting the specifics of RDF.

- They show comparable performance to more specific and more general approaches.

Future Work

- Test with other kernel machines
- Investigate dependence of graph characteristics on performance of various graph kernels

Thanks!