Demo Track

https://github.com/iai-group/pkg-api

# PKG API: A Tool for Personal Knowledge Graph Management

Nolwenn Bernard, Ivica Kostric, Weronika Łajewska, Krisztian Balog, Petra Galuščáková, Vinay Setty, Martin G. Skjæveland
*University of Stavanger*

## Background

- A Personal Knowledge Graph (PKG) centrally stores all information related to its owner in a structured representation
  - Here: fundamental unit of information is a *statement*
- The owner has control over their data, i.e., determine what data is stored and its accessibility
- Practical implementations of PKG are lacking

## Contributions

We propose a user-friendly solution to manage PKGs based on:

- A PKG vocabulary to represent statements in the PKG
- A PKG API with user-facing and service-oriented functionalities for PKG management
- A web-based PKG Client to make PKG management accessible for all users

### USER INTERFACE - PKG CLIENT

- User-friendly and intuitive web-based interface
- Interactions with the PKG through natural language and forms

### NATURAL LANGUAGE UNDERSTANDING

- Employ pre-trained LLMs with few-shot chain-of-thought reasoning prompts to perform:
  - User intent classification as addition, retrieval, deletion of statement, or unknown
  - Extraction of a subject, predicate, and object (SPO) triple
  - Identification of preference if relevant

### ENTITY LINKING

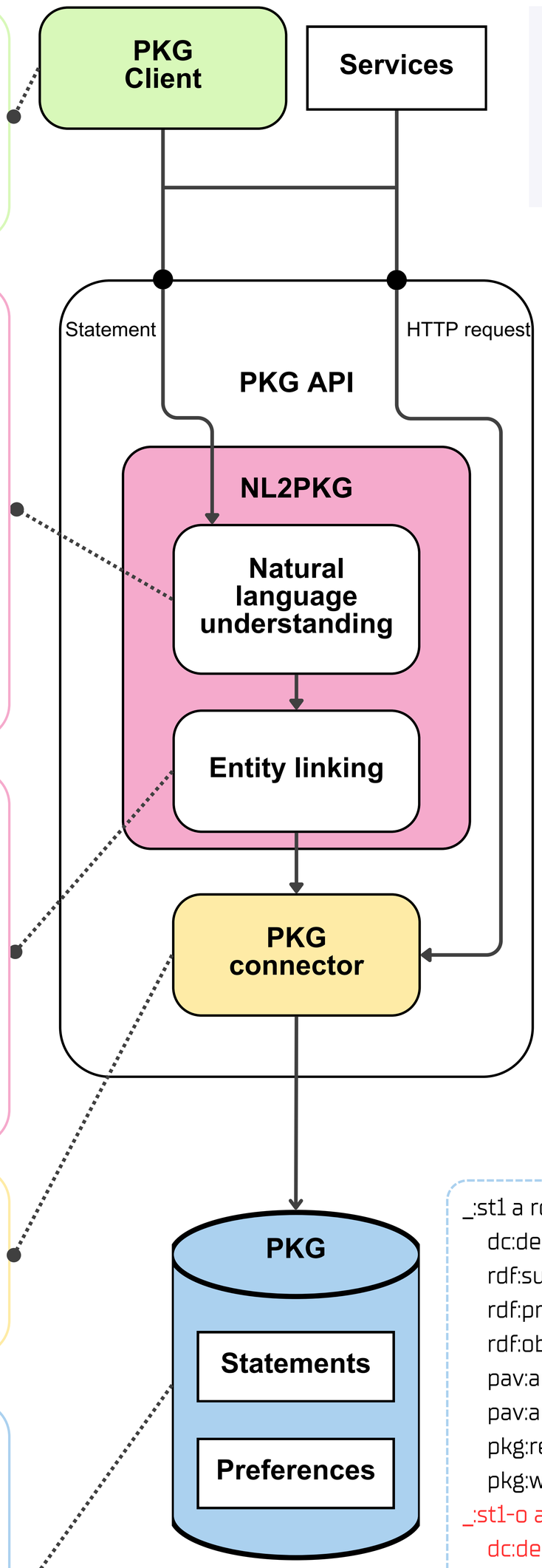
- SPO triple resolution from their surface form
- Normalization of entities and relations in line with external KGs
- It is recommended to resolve the subject with an entity linker specific to the PKG as it usually refers to the user's private circle

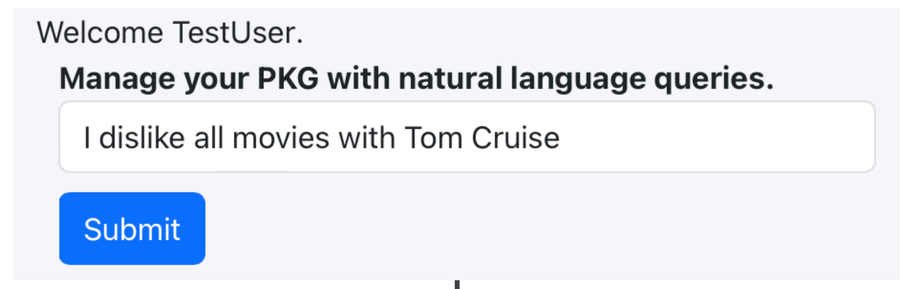
### PKG CONNECTOR

- SPARQL query generation based on the annotated statement and preference
- Execution of queries against the PKG

### PKG VOCABULARY

- Simple data model designed to support all kinds of statements and incremental semantic enrichment
- All statements represented as reified RDF triples augmented with original text and provenience data
- Additional links between statement and its components may be added in post-analysis to increase precision and quality




- Intent = ADD
- SPO triple = I | dislike | all movies with Tom Cruise
- Preference = -1

- Subject = my:I
- Predicate = Concept[dislike]
- Object = Concept[all movies with Tom Cruise, [dbp:Tom_Cruise, schema:actor], [schema:Movie]]

```
pkg.add(Triple[Subject, Predicate, Object],
Preference[object, -1])
```

```
_:st1 a rdf:Statement ;
    dc:description "I dislike all movies with the actor Tom Cruise."@en ;
    rdf:subject my:I ;
    rdf:predicate [ a skos:Concept ; dc:description "dislike" ] ;
    rdf:object _:st1-o ;
    pav:authoredBy my:I ;
    pav:authoredOn "2023-12-15T21:12:40"^^xsd:dateTime ;
    pkg:readAccessRights "MovieBot", "IMDB" ;
    pkg:writeAccessRights "MovieBot" .
_:st1-o a skos:Concept ;
    dc:description "All movies with the actor Tom Cruise" ;
    skos:related schema:actor, dbp:Tom_Cruise ;
    skos:broader schema:Movie .
my:I wi:preference [
    pav:derivedFrom _:st1 ;
    wi:topic _:st1-o ;
    wo:weight [ wo:weight_value -1.0 ; wo:scale pkg:StandardScale ] ] .
```