Frankenstein: a Platform Enabling Reuse of Question Answering Components

Kuldeep Singh, Andreas Both, Arun Sethupat Radhakrishna, Saeedeh Shekarpour

Fraunhofer IAIS (Germany), DATEV eG (Germany), University of Minnesota (USA), University of Dayton (USA)
Question Answering Tasks and Components

- For translating following question to SPARQL:

  “Which comic characters are painted by Bill Finger?”

- A question answering process might require:
  - NER component recognise the string Bill Finger as entity
  - NED component links string Bill Finger to dbr:Bill_Finger
  - RL component links string painted by to dbo:creator
  - CL component links string comic characters to dbo:ComicsCharacter

Note: dbr correspond to http://dbpedia.org/resource/, dbo correspond to http://dbpedia.org/ontology/
Question Answering Tasks

- For translating question “Which comic characters are painted by Bill Finger?” to SPARQL:
  - Query Builder (QB) component construct the SPARQL query:

```sparql
SELECT DISTINCT ?uri WHERE {
}
```
State of the Art QA Systems

- Number of QA systems over DBpedia has increased
- QA systems implement similar tasks to answer user’s question
QA Frameworks for promoting Reusability

- QALL- ME (Ferrández et al., J. Web Sem., 2011)
- openQA (Marx et al., Semantics, 2014)
- QANUS (Ping Ng et al., CoRR, 2015)
- OKBQA (Kim et al., SIGIR, 2017)

Observation: Several flaws particularly w.r.t. composability & data representation
Vision

To building up an infrastructure in which the state-of-the-art question answering (QA) components can be easily integrated, run, and evaluated.

→ Enable you to build your research on the shoulders of giants.

exactly 200 years ago

https://en.wikipedia.org/wiki/Frankenstein
Contributions

- Decoupled architecture of Frankenstein for two set of resources:

  - **R1**: 29 QA components (following the Qanary methodology)
    - 11 name entity recognition (NER) components
    - 9 name entity disambiguation (NED) components
    - 5 relation linking (RL) components
    - 2 class linking (CL) components
    - 2 query builder (QB) components

Andreas Both, Dennis Diefenbach, Kuldeep Singh, Saeedeh Shekarpour, Didier Cherix and Christoph Lange. “Qanary -- An Extensible Vocabulary for Open Question Answering Systems”, ESWC 2016
Contributions

- Decoupled architecture of Frankenstein for two set of resources:
  - R2: Component-wise Runner and Evaluator
    - Automatic process of running, evaluating integrated components
    - Independent components for evaluating any component
Frankenstein Platform (as decoupled architecture)
Component Selection

- Components are selected based on:
  - provided as open source?
  - research publication associated with the component?
  - provided as RESTful service?
  - used earlier in other research work?

- All components fulfil at least two of the above criteria.
Challenges for Component Integration in Frankenstein

● To deal with interoperability of components
  ○ Some components available as RESTful Web service, few as open source code
  ○ Implemented in different programming languages
● Heterogeneous output format of components
● To decouple existing systems
  ○ e.g., SINA (Shekarpour et al., WWW 2013) to extract the query builder
Reusable Components and Integration (R1)

- Used Qanary methodology for integrating components
  - populate microservice-based architecture
  - Knowledge-driven QA process, agnostic of implementation details
  - Represent knowledge about QA process in qa vocabulary

  Example: Annotation of relation:

  - PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  - PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
  - qa:AnnotationOfRelation rdf:type owl:Class ;
    rdfs:subClassOf qa:AnnotationOfQuestion .
Reusable Components and Integration (R1)

Named Entity Disambiguation (NED)
- DBpedia Spotlight
- Tag Me
- Aylien

Relation Linking (RL)
- ReMatch
- RelMatch
- RNLIWOD

Class Linking (CL)
- CL NLIWOD
- OKBQA CLS

Query Builder (QB)
- SINA
- NLIWOD QB

Qanary Wrappers

19 Components
5 Components
2 Components
2 Components
Reusable Components and Integration (R1)

- Data is stored using an annotation style (following WADM)
  - c.f. https://www.w3.org/TR/annotation-model/
- Sample output of NED component for “Which comic characters are painted by Bill Finger?”

```xml
<tag:stardog:api:0.9278702836234858>
  a <http://www.wdaqua.eu/qa#AnnotationOfInstance>;
  oa:core/hasTarget _:bnode_5daa02597cbe_4e6c_8504_3f73463e0fd8_69;
  oa:hasBody <http://dbpedia.org/resource/Bill_Finger>;
  oa:annotatedBy <https://tagme.d4science.org/>;
```

- Information is stored and can be analyzed
Component-wise Runner and Evaluator (R2)

- An automatic process of composing, running, and evaluating QA components and QA pipelines
- Simple Qanary UI for executing single questions at a time
- Scripts to run the components to execute large number of questions/text corpora at bulk, e.g.

  .. serverUpdateAndRun.sh stanfordNER

  .. serverUpdateAndRun.sh BabelflyNER AGDISTIS
Component-wise Runner and Evaluator (R2)

- QA Pipeline Executor
  - Independent component for component and pipeline execution
    - Reads from input file and pass questions to selected component
    - Generate output in “.ttl” file format: questionID_component.ttl
  - An automatic process, efficient in large number of text/questions.
  - Step-wise / individual benchmarks for QA pipeline
    - for datasets like QALD-5, LC-QuAD, ...
  - Use QA pipeline Executor and Evaluator together to benchmark components at any stage in QA pipeline
Resource Impact: Usefulness for QA community

- Scalable & developer-friendly platform for building Question Answering Systems by reusing existing components.
- Support automatic process of evaluating components
- Platform can be used for building and measuring QA systems
- Researchers can integrate their component, analyse the usability of component w.r.t. Question Answering.
- Researchers can analyse their tool w.r.t. existing components for different tasks, NED, NER, RL etc.
Resource Impact: Usefulness Beyond QA community

- Existing tools can be used to analyse large text corpora (e.g. tweets) using entity recognition and disambiguation components
- Architecture is independent of knowledge bases, component functionalities, domain.
Conclusions

Frankenstein is dedicated to extending the Qanary ecosystem

- **R1**: Reusable QA components
  - Collection of 29 reusable components for building QA systems, text analysis
  - 380 QA pipelines can be created just by configuration

- **R2**: Component wise Runner & Evaluator
  - Automatic process for running and evaluating integrated components
  - Useful for analysing a large text corpora/questions in automatically
Conclusions

- Benefits for researchers w.r.t. efficiency
  - more research in less time
- Provide knowledge-driven ecosystem of QA components
- Create foundations for rapid research process, e.g.
  - compare your research in the field of QA
  - analyze questions w.r.t. the required QA tasks / components
  - take all input for generating queries using machine learning

Future Works

- To integrate more components in Frankenstein platform using Qanary
- Focus on extending components for specific domains, e.g. biomedical
- Extend the integrated components to be agnostic of Knowledge Bases (currently all components for DBpedia)
- Replace Evaluator component with direct integration of Frankenstein into GERBIL-QA
  - GERBIL recently released benchmarking capabilities for QA tasks
  - Frankenstein can be directly integrated in GERBIL-QA
Frankenstein: a Platform Enabling Reuse of Question Answering Components

by Kuldeep Singh, Andreas Both, Arun Sethupat Radhakrishna, Saeedeh Shekarpour

- Resource 1: 29 Qanary components provided
- Resource 2: tools for composing and analyzing QA systems
- your benefits: reduced engineering, broad capabilities of QA functionalities
  ➜ faster research

  ○ Contact: kuldeep.singh@iais.fraunhofer.de or contact@andreasboth.de
  ○ WDAQUA http://wdaqua.eu/ Marie Skłodowska-Curie grant agreement No. 642795, project: Answering Questions using Web Data (WDAqua)