11-791 Design and Engineering of Intelligent Information System Fall 2014 Project

Building a Pipeline for Biomedical Question Answering

Important dates

- **Hand out: November 3.**[a]
- **Milestone 1 (M1) : Concepts, documents, and triples retrieval. Turn in: November 10 (tentative).**
  - For this milestone, you only need to send us the URL of your project repository page (i.e.,`https://github.com/COURSENUM-ID/project-teamID` where 'COURSENUM' is the course designation number 11791/11693, and ID is your assigned team number). We will look into your Issues page and Wiki page, and expect you have created milestones and issues, reported the results of your components, and completed proposal.

[a] This version was built on November 3, 2014

Useful information

1. Please visit Piazza regularly to check if a newer version is published. We may have new versions or revised instruction at the begining of each milestone.

   We expect that most of the general communication between the instructor team and students will take place on Piazza `https://piazza.com/class/hyvsubeilei6dd`. For private questions, e.g., regarding grades, you may contact instructors by e-mail. Your friendly TAs are: Avner Maiberg (`amaiberg@andrew.cmu.edu`), Parag Argawal (`paraga@andrew.cmu.edu`), Leonid (Leo) Boytsov (`srchvrs@cmu.edu`), and Xuezi (Manfred) Zhang (`xueziz@andrew.cmu.edu`).

2. Again, both source files and pdf file of this assignment are publicly available on my GitHub

   `http://github.com/amaiberg/software-engineering-preliminary`

   Please feel free to fork the project and send a pull request back to me as some of you did for Homework 0 for any error. Or you can just report an issue at

   `http://github.com/amaiberg/software-engineering-preliminary/isues`

Milestone 1

Creating your first BioASQ pipeline

In this task you will implement components to retreive concepts, documents, and RDF triples for a biomedical question answering pipeline (based on BioASQ's Task1b Phase A). You will create your own pipeline based on a provided archetype framework called `DEIIS-project-archetype`. Note: the **only requirement** is that you use the typesystem provided in the archetype. You are otherwise permitted and encouraged to use any additional types and sources you may wish.

M 1.1 Types and evaluation metrics

The following are the output types described in the official BioASQ evaluation documentation [1]:

Concepts

> A list of relevant concepts $c_{i,1},\ c_{i,2},\ c_{i,3},\ \ldots$ from the designated terminologies and ontologies. The list should be ordered by decreasing confidence, i.e., $c_{i,1}$ should be the concept that the system considers most relevant to the question q_i, $c_{i,2}$ should be the concept that the system considers to be the second most relevant etc. A single concept list will be returned per question and participant, and the list may contain concepts from multiple designated terminologies and ontologies. The returned concept list will actually contain unique concept identifiers (obtained from the terminologies and ontologies), rather than terms (words or phrases).

Example: Here the JSON array `concepts` corresponds to designated terminologies and ontologies

```
"body": "What is the role of PrnP in mad cow disease?",
      "type": "factoid",
      "id": "160",
       "concepts": [
              "http://www.disease-ontology.org/api/metadata/DOID:162",
              "http://www.uniprot.org/uniprot/M3K8_RAT"
               ]
```

Documents

> A list of relevant articles (documents) $d_{i,1}, d_{i,2}, d_{i,3}, \ldots$. from the designated article repositories. Again, the list should be ordered by decreasing confidence, i.e., $d_{i,1}$ should be the article that the system considers most relevant to the question, $d_{i,2}$ should be the article that the system considers to be the second most relevant etc. A single article list will be returned per question and participant, and the list may contain articles from multiple designated repositories. The returned article list will actually contain unique article identifiers (obtained from the repositories).

Example: Here the JSON array `documents` corresponds to the document PMIDs.

```
"body": "What is the role of PrnP in mad cow disease?",
      "type": "factoid",
      "id": "160",
       "documents": [
            "http://www.ncbi.nlm.nih.gov/pubmed/23420787",
            "http://www.ncbi.nlm.nih.gov/pubmed/23397482",
            "http://www.ncbi.nlm.nih.gov/pubmed/23298766",
             ...
            "http://www.ncbi.nlm.nih.gov/pubmed/17451943",
            "http://www.ncbi.nlm.nih.gov/pubmed/12146646",
            "http://www.ncbi.nlm.nih.gov/pubmed/1366363"
        ],
```

Triples

> A list of relevant RDF triples $t_{i,1}, t_{i,2}, t_{i,3}, \ldots$ from the designated ontologies. Again, the list should be ordered by decreasing confidence. A single triple list will be returned per question and participant, and the list may contain any triples from multiple designated ontologies.

Example: Here `triples` refers to `o` (object), `p` (predicate), and `s` (subject).

```
"body": "What is the role of PrnP in mad cow disease?",
      "type": "factoid",
      "id": "160",
      "triples": [
       {
           "o": "http://linkedlifedata.com/resource/umls/label/A17680439",
           "p": "http://www.w3.org/2008/05/skos-xl#prefLabel",
           "s": "http://linkedlifedata.com/resource/umls/id/C2827401"
       },
       {
           "o": "fda",
           "p": "http://www.w3.org/2004/02/skos/core#altLabel",
           "s": "http://linkedlifedata.com/resource/#_4434464B59390011"
       },
       {
           "o": "fda",
           "p": "http://www.w3.org/2004/02/skos/core#altLabel",
           "s": "http://linkedlifedata.com/resource/#_51395844503300D"
       },
           ...
     ]
```

Note that for each phase you will return a list of k items in order of decreasing confidence (where $k \leq 100$). See the evaluation section below for how these will be evaluated.

Evaluation

Please refer to table 1.1 to see the evaluation used for each type.

Retrieved items	Unordered retrieval measures	Ordered retrieval measures
concepts	mean percision, recall, F-measure	MAP,**GMAP**
articles	mean percision, recall, F-measure	MAP,**GMAP**
triples	mean percision, recall, F-measure	MAP,**GMAP**

Figure 1.1: Evaluation metrics

Also, please review the definitions of the evaluation metrics:

- Precision and Recall:

$$P = \frac{TP}{TP + FP} \qquad \text{(where TP are true positives, and FP are false positives)}$$

$$R = \frac{TP}{TP + FN} \qquad \text{(where TP are true positives, and FN are false negatives)}$$

- F-measure:

$$F = 2 \cdot \frac{P \cdot R}{P + R} \qquad \text{(Harmonic mean of Precision and Recall)}$$

- Average Precision (AP):

$$AP = \frac{\sum_{r=1}^{|L|} P(r) \cdot rel(r)}{L_R} \qquad \text{(see below)}$$

  Where, for any given query q_i and a golden set of items:

  - $|L|$ is the total number of items.
  - $|L_R|$ is the number of relevant items.

  - $P(r)$ is the precision for a list containing the first r items.
  - $rel(r)$ is an indicator function for the existence of item r in the golden set (i.e., it returns 1 if the rth item is relevant, and 0 otherwise).

- Mean Average Precision (MAP):

$$MAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$$

(To get the average precision for list of queries $q_1, q_2, \ldots, q_n$.)

- Geometric Mean Average Precision (GMAP):

$$GMAP = \sqrt[n]{\prod_{i=1}^{n}(AP_i + \epsilon)} \quad \text{(with some small } \epsilon \text{ for cases where } AP_i = 0\text{)}$$

GMAP is similar to MAP, only used to further penalize low performing queries.

M 1.2 Data Sources

As described in [1], you are provided with the following data sources:

GoPubMed

GoPubMed is a knowledge-based search engine for biomedical texts. The Gene Ontology (GO) and Medical Subject Headings (MeSH) serve as Table of contents in order to structure the millions of articles of the MEDLINE database. The search engine allows its users to find relevant search results significantly faster than PubMed.

MEDLINE

MEDLINE (Medical Literature Analysis and Retrieval System Online) is a bibliographic database of life sciences and biomedical information. It includes bibliographic information for articles from academic journals covering medicine, nursing, pharmacy, dentistry, veterinary medicine, and health care. MED- LINE also covers much of the literature in biology and biochemistry, as well as fields such as molecular evolution.

PubMed

PubMed is a free database accessing primarily the MEDLINE database of references and abstracts on life sciences and biomedical topics. It comprises more than 22 million citations for biomedical literature through MEDLINE, life science journals, and on-line books. Citations may include links to full-text content from PubMed Central and publisher web sites. The United States National Library of Medicine (NLM) at the National Institutes of Health maintains the database as part of the Entrez information retrieval system. PubMed has an publically available web interface for accessing its contents.

MeSH

MeSH (Medical Subject Headings) is the NLM controlled vocabulary thesaurus used for indexing arti- cles for PubMed. It consists of approximately 26.000 terms and new terms are added in a yearly basis. The terms are organised hierarchically in 12 trees. MEDLINE and PubMed use Medical Subject Head- ings (MeSH) for information retrieval. In addition, many engines (e.g. GoPubMed) are designed to access and search the MEDLINE content using MeSH terms.

Note: we also include a Java API client to help you access these services in the archetype 1.3.‘

M 1.3 Archetype

Typesystem

For this milestone, you are required to use the typesystem `OAQATypes.xml` included in the archetype. Note however, you are only required to use the types defined in 1.3 and their supertype ancestry. You are of course welcome to extend the typesystem and/or use any other preexisting types. Most importantly. it is imperative that you use the inherited `uri` and `rank` attributes, as we will use them for evaluation.

Web Services

To access the data sources in 1.2, we provide `bioasq-gopubmed-client`, a packaged API wrapping the following web services (as described in [2]):

- The Medical Subject Headings (MeSH) Hierarchy:a service for acessing the MeSH ontology, with input parameter "*findEntity*", and output parameter "*findings*", which contains the list of related concepts (a list of "*concept*" entries with "*label*" entries), given the query submitted with the input parameter. Additional information is provided inside each "label" entry in the JSON object, such as "termId" and "uri" of the concept. In addition, inside each "concept" entry, the offsets in which the query keywords matched each returned concept are provided.
- Disease Ontology: a service for accessing the Disease Ontology, with the same input and output parameters as aforementioned.
- Gene Ontology: a service for accessing the GO ontology, with the same input and output parameters as aforementioned.

BioASQ Type	**UIMA Type**
Result (supertype)	• `edu.cmu.lti.oaqa.type.retrieval.SearchResult` – `uri` – `rank`
Concept	• `edu.cmu.lti.oaqa.type.retrieval.ConceptSearchResult` – `uri` (inherited) – `rank` (inherited) • `edu.cmu.lti.oaqa.type.kb.Concept`
Document	• `edu.cmu.lti.oaqa.type.retrieval.Document` – `uri` (inherited) – `rank` (inherited)
Triple	• `edu.cmu.lti.oaqa.type.retrieval.TripleSearchResult` – `uri` (inherited) – `rank` (inherited) • `edu.cmu.lti.oaqa.type.kb.Triple` – `subject` – `predicate` – `object`

Figure 1.2: Typesystem Mapping

- Jochem: a service accessing the Jochem ontology, with the same input and output parameters as aforementioned.
- Uniprot: accessing the UniProt database, with the same input and output parameters as aforementioned.
- Linked Data: a service for accessing the LinkedLifeData platform triples. The input parameter is “findTriples”, and accepts any keywords as query. The output

Service	API call
MeSH	`findMeshEntitiesPaged`
DO	`findMeshDiseaseOntologyPaged`
GO	`findMeshGeneOntologuPaged`
Jochem	`findJochemEntitiesPaged`
Uniprot	`findUniprotEntitiesPaged`
Linked Data	`findLinkedLifeDataEntitiesPaged`
Pubmed	`findPubMedCitations`

Figure 1.3: Client methods for calling web services

parameter contains a list of "*triples*" entries. Each entry has in turn a "*subj*", "*pred*", "*obj*" and "*score*" field, representing the subject, the predicate and the object of the triple, and the matching score given the input query.

- Indexed Document Sources (Pubmed): service for accessing the PubMed indexed documents (titles and abstracts), with the same input parameters as aforementioned, and the output parameter containing "document" entries in the return JSON object. Each entry has a "*pmid*" element, which is the PubMed id of the indexed citation, a "*documentAbstract*" entry, and a "*title*" entry. In addition, the *MeSH*annotations are provided when available.

See 1.3 for how to use the client to call each web service. Note that all of these methods have the signature (`String keywords`, `page`, `conceptsPerPage`). Please also inspect `GoPubMedServiceExample.java` included in the archetype for a concrete example of how to use this client. For more detail on web services, see Appendix A.

Data

In the archetype you will also find `BioASQ-SampleData1B.json` containing an annotated version of the 29 sample questions shown in M0. We also provide you with a convenience class `JsonCollectionReaderHelper.java` to read from JSON format.

M 1.4 Creating Maven project from the archetype

For this homework, we create another archetype called `DEIIS-project-archetype` to help you quickly get your development started. We briefly show you the process you've gone through for your Homework 1.

1. Open your Eclipse's **Preferences** window, and navigate to **Maven** → **Archetypes**, and click **Add Remote Catalog...**.
2. Type the following URL into the **Catalog File** field. `https://raw.githubusercontent.com/oaqa/DEIIS-project-archetype/master/archetype-catalog.xml`

Optionally, you can add a **Description** for this catalog, for example “BioQA Catalog”. Then click **OK** on the **Remote Archetype Catalog** window and another **OK** on the **Preferences** window.

3. Add the following to your `settings.xml` from Listing 1.1.

```
<server>
  <id>oaqa</id>
  <username>ID</username>
  <password>PASSWORD</password>
</server>
```

Listing 1.1: Configuring settings.xml

4. Now you can follow almost follow the same steps to import to Eclipse as you did for Homework 1. Since we have created the archetype for you, remember to unselect **Create a simple project (skip archetype selection)**. Then click **Next**.

5. Here you can select “BioQA Catalog” (or other names you specified in the previous step) or “All Catalogs” in the drop-down menu for **Catalog**. Then, type in “project-archetype” (without quotes) in the **Filter** field, and in order to get the latest snapshot archetypes, you need to check **Include snapshot archetypes** as well. Select the archetype listed below, and click next to continue.

6. In the next window, you are asked to specify the **Group Id** and **Artifact Id**. Similar to Homework 0 and 1, the Group Id is

   edu.cmu.lti.11791.f14.project

   and Artifact Id is

   project-teamXX

   with XX being your team number. Remember to specify `Package` as

   edu.cmu.lti.11791.f14.project

   Then click **Finish**.

7. You need to edit the `pom.xml` file to type in the SCM information of your GitHub repository for project as you did in Homework 0.

8. Same as before, you probably need to right-click the project name, and click **Maven** → **Update Project** to download the dependencies.

You can see that we have included:

- the `pom.xml`. To see the dependencies you can click the **Dependencies** tab after you double-click the pom file. There you will see that in addition to the core UIMA components, the project depends only on the `bioasq-gopubmed-client` project. This client will simplify calling all the ontology services defined for this milestone.

  If you want to take a look at what is inside `bioasq-gopubmed-client` project and other indirect dependencies, you can unfold the `Maven Dependencies` folder under your project name in the Package Explorer View.

Before you commit and push all the initial code changes to GitHub repository, we suggest you to first test if you can successfully run the pipeline.

Appendix A

Web Services

Document Sources

The primary corpora for text-based QA in the biomedical domain are accessible through PubMed and PubMed Central. PubMed, a service provided by the National Library of Medicine (NLM), under the U.S. National Institutes of Health (NIH), contains over 23 million citations from Medline, a bibliographic database (DB) of biomedical literature, and other biomedical and life science journals dating back to the 1950s. It is accessible through the National Center for Biotechnology Information (NCBI). PubMed Central (PMC) is a digital archive of full-text biomedical and life science articles. The full text of all PubMed Central articles is freely available. As of July 2011, the archive contains approximately 2.2 million items, including articles, editorials and letters.

Linked Data

The BioASQ tasks 1b and 2b require the usage of biomedical data expressed as triples, e.g., *subject-predicate-object* structured facts, extracted from biomedical resources or bibliography. In this direction,the Linked Life Data project provides the LinkedLifeData platform. LinkedLifeData is a data warehouse that syndicates large volumes of heterogeneous biomedical knowledge in a common data model. The platform uses an extension of the RDF model that is able totrack the provenance of each individual fact in the repository and thus update the information. It contains currently more than 8 billion statements, with almost 2 billion entities involved. The statements areextracted from 26 biomedical resources, such as PubMed, UMLS, DrugBank, Diseasome, and Gene Ontology. The statements are publicly available, and the project provides also a wide list of instance mappings.

Disease Ontology

The Disease Ontology (*DO*) contains data associating genes with human diseases, using established disease codes and terminologies. Approximately 8, 000 inherited, developmental and acquired human diseases are included in the resource. The *DO* semantically integrates disease and medical vocabulary through extensive cross-mapping and integration of MeSH, ICD, NCIs thesaurus, SNOMED CT and OMIM disease-specific terms and identifiers. The *DO* is utilized for disease annotation by major biomedical databases (e.g., Array Express, NIF, IEDB), as a standard representation of human disease in biomedical ontologies (e.g., IDO, Cell line ontology, NIFSTD ontology, Experimental Factor Ontology, Influenza Ontology), and as an ontological cross-mappings resource between DO, MeSH and OMIM (e.g., GeneWiki). DO has been incorporated into open source tools (e.g., Gene Answers, FunDO) to connect gene and disease biomedical data through the lens of human disease.

Gene Ontology

The Gene Ontology (GO) is currently the most successful case of ontology use in bioinformatics and provides a controlled vocabulary to describe functional aspects of gene products. The ontology covers three domains: cellular component, the parts of a cell or its eextracellular environment; molecular function, the elemental activities of a gene product at the molecular level, such as binding or catalysis; and biological process, operations or sets of molecular events with a defined beginning and end, pertinent to the functioning of integrated living units: cells, tissues, organs, and organisms.

Jochem

Jochem (Hettne et al. (2009)), the Joint Chemical Dictionary,, is a dictionary for the identification of small molecules and drugs in text, combining information from UMLS, MeSH, ChEBI, DrugBank, KEGG, HMDB, and ChemIDplus. The resources were chosen on the basis of free availability. They are downloadable terminology databases containing small molecules from human studies. Given the variety and the population of the different resources merged in Jochem, it is currently one of the largest biomedical resources for drugs and chemicals.

The Medical Subject Headings Hierarchy

Medical Subject Headings (MeSH) is a hierarchy of terms maintained by the United States National Library of Medicine (NLM) and its purpose is to provide headings (terms) which can be used to index scientific publications in the life sciences, e.g., journal articles, books, and articles in conference proceedings. The indexed publications may be then searched through popular search engines, such as PubMed or GoPubMed, using the MeSH headings to filter semantically the results. This retrieval methodology seems to be in some cases beneficial, especially when precision of the retrieved results is important (Doms and Schroeder (2005)).

MeSH includes three types of data: (i) descriptors, also known as subject headings, (ii) qualifiers, and, (iii) supplementary concept records. Descriptors are the main terms that are used to index scientific publications. The descriptors are organized into 16 trees, and as of 2013 they are $26,8531$. They include a short description or definition of the term, and they frequently have synonyms, known as entry terms. Qualifiers, also known as subheadings, may be used additionally to narrow down the topic of each of the descriptors. In total there are approximately 80 qualifiers in MeSH. Supplementary concept records, approximately $214,000$ in the most recent MeSH release, describe mainly chemical substances and are linked to respective descriptors in order to enlarge the thesaurus with information for specific substances. MeSH is the main resource used by PubMed to index the biomedical scientific bibliography in Medline

Uniprot

The Universal Protein Resource (UniProt) provides the scientific community with a comprehensive, high-quality and freely accessible resource of protein sequence and functional information. Its protein knowledge base consists of two sections: Swiss-Prot, which is manually annotated and reviewed, and contains approximately 500 thousand sequences, and TrEMBL, which is automatically annotated and is not reviewed, and contains approximately 23 million sequences. The primary mission of the Universal Protein Resource (UniProt) is to support biological research by maintaining a stable, comprehensive, fully classified, richly and accurately annotated protein sequence knowledge base, with extensive cross-references and querying interfaces freely accessible to the scientific community. In particular, the Swiss Prot component of UniProt, it is a high-quality, manually annotated, non-redundant protein sequence database which combines information extracted from scientific literature and biocurator-evaluated computational analysis. The aim of Swiss-Prot is to provide all known relevant information about a particular protein. Annotation is regularly reviewed to keep up with current scientific findings.

Bibliography

[1] Georgios Balikas, Ioannis Partalas, Aris Kosmopoulos, Sergios Petridis, Prodromos Malakasiotis, Ioannis Pavlopoulos, Ion Androutsopoulos, Nicolas Baskiotis, Eric Gaussier, Thierry Artieres, and Patrick Gallinari. Evaluation framework specifications. Project deliverable D4.1, 05/2013 2013.

[2] George Tsatsaronis, Matthias Zschunke, Michael R. Alvers, and Christian Plonka. Report on existing and selected datasets. Project deliverable D3.2, 01/2013 2013.