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

Building a Pipeline for Biomedical Question Answering

Important dates

- **Hand out: October 24.**[a]
- **Milestone 0 (M0): Project Wiki and proposal draft. Turn in: October 27.**
  - You are required to fill out the project proposal for your assigned team on a newly created GitHub repository. Please 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 expect to find a filled out draft of the proposal template on your project Wiki. Please note that this is only a draft of your proposal, the deadline is just to get your project wiki and your teams together. You will continue working on this proposal until Milestone 1. Please see the proposal outline at the end of this document.
- **Milestone 1 (M1) : Concepts, documents, and triples retrieval. Turn in: November 10.**
  - As in previous homeworks, you are required to submit all the sources for your components via a Maven release, and you DON'T need to submit any other documents.
  - In addition, please 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.
- **Milestone 2 (M2): Snippets retrieval. Turn in: November 19.**
  - You are required to submit all the sources for your components via a Maven release, and you DON'T need to submit any other documents.
  - We will again look into your Issues page and Wiki page, and expect you properly solved your previous created milestones and issues, and probably created new milestones and issues, and reported the evaluation results of your M2 on your own, your team meeting minutes and probably your revised project goals.

[a]This version was built on October 24, 2014

- **Milestone 3 (M3): Exact answer generation. Turn in: December 1.**
  - You are required to submit all the sources for your components via a Maven release.
  - We expect you properly solved all the issues and accomplish all milestones, and reported the evaluation results of your final system, your team meeting minutes and a final summary.
  - Name your presentation slides as `project-teamID.ppt`[a] and put it in the `src/main/resources/docs`.

[a] Or other formats, e.g., pptx, odp, pdf, etc.

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/issues`

Task 1

Getting Ready for a Biomedical Question Answering System

In this task, you won't need to write any code (except that you need to edit your Wiki page in Github flavored markdown language), instead you will learn a new biomedical question answering task. Plus, we will also mention how to create Milestones and Issues in the GitHub ISSUES system.

Task 1.1 Learning biomedical question answering task

Question Answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP) which is concerned with building systems that automatically answer questions posed by humans in a natural language.[1] The system you are going to build will handle biomedical questions (e.g. "Is Rheumatoid Arthritis more common in men or women?"). As you can see, different from a general question answering, a biomedical QA system requires domain specific knowledge bases, NLP tools, literature collections, etc., in order to produce meaningful answers.

One of the biomedical information retrieval challenges is the BioASQ task[2]. BioASQ provides a platform for testing and experiments of retrieval methods for different tasks. Specifically, you will be focusing on Task 2b of the BioASQ challenge on the retrieval of "relevant concepts (from designated terminologies and ontologies), relevant articles (in English, from designated article repositories), relevant snippets (from the relevant articles), relevant RDF triples (from designated ontologies), exact answers (e.g., named entities in the case of factoid questions) and 'ideal' answers (paragraph-sized summaries), both in English on retrieval of relevant passages within an article collection from biomedical journals as answers to the questions in biomedical domain[3]".

[1] `https://en.wikipedia.org/wiki/Question_answering`
[2] `http://www.bioasq.org/`
[3] From BioASQ task description

BioASQ provides 29 annotated sample questions created by domain experts in biology. There are four types of questions: factoid, yes/no, list questions (provide one or more answers), summary questions. All of the questions are listed in Table 1.1:

Table 1.1: 29 questions from BioASQ

Type	Question
factoid	Is Rheumatoid Arthritis more common in men or women?
yesno	Are there any DNMT3 proteins present in plants?
list	What is the most prominent sequence consensus for the polyadenylation site?
summary	What is the function of the mammalian gene Irg1?
yesno	Is thrombophilia related to increased risk of miscarriage?
list	Which extra thyroid tissues have thyrotropin (TSH) receptors?
factoid	What is the methyl donor of DNA (cytosine-5)-methyltransferases?
summary	Super-SILAC is a method used in quantitative proteomics. What is the super-SILAC mix? (SILAC: Stable Isotopic labelling by aminoacids in cell culture)
list	Tumors of which three organs are classically associated with the multiple endocrine neoplasia type 1 syndrome?
summary	What is the mechanism of action of anticoagulant medication Dabigatran?
yesno	Is Mammaprint approved by the United States Food and Drug Administration?
list	Which acetylcholinesterase inhibitors are used for treatment of myasthenia gravis?
factoid	Which medication should be administered when managing patients with suspected acute opioid overdose?
yesno	Does the Oncotype DX test work with paraffin embedded tissues?
yesno	Is depression associated with poor prognosis of brain tumor patients?
summary	What is known about the reimbursement of Viagra?
yesno	Does HER2 under-expression lead to favorable response to trastuzumab?
yesno	Can Alzheimer's disease related miRNAs be detected in patients' blood?
factoid	Which antiepileptic drug is most strongly associated with spina bifida?
factoid	Where in the cell do we find the protein Cep135?
list	In which isochores are Alu elements enriched?
list	Which forms of cancer is the Tpl2 gene associated with?
summary	What is the INSURE procedure in premature babies?
list	Which species may be used for the biotechnological production of itaconic acid?
summary	What is the role of TRH in hypertension?
list	Which histone marks are deposited by Set7?
yesno	Are there any desmins present in plants?
summary	Is it possible to detect survivin protein expression in normal human adult tissues?
summary	How do histone methyltransferases cause histone modification?

Concepts, documents, and triples will be wrapped in services described in Milestone 1. Snippet retrieval will be described in Milestone 2. Finally, exact answer generation will be described in Milestone 3.

Task 1.2 Team coding with GitHub

You will again use GitHub to host your project code. But different from previous homeworks, all the team members need to collaborate on the project, and commit the

code changes to the same repository, which means it is good to learn how to more about git and GitHub in this subtask.

Creating a repository

1. The team leader needs to create a repository with his/her GitHub acccount, and name your project as `COURSENUM-teamID`, where `ID` is your team number, which is a two-digit number ranging from 01 to 18. If you are not sure how to create a GitHub repository, please refer to Homework 0.

   At this point, all the team members are able to checkout the empty project and start working on their own. But this is NOT recommended! We recommend the team leader could checkout the project from GitHub repository, go through the entire Maven project building process (we will come to this in the next subtask) until you can run a simple pipeline. Then, the team leader again, from his/her laptop, commits and pushes everything to the repository before the team members clone the project.

2. Now, everyone has the read permission to your project repository (since it is a public repository). To help your team members gain read/write permissions, you, the team leader, should add them as collaborators. You need to click the **Admin** tab on your project homepage, and click the **Collaborators** menu on the left. Put in your teammates' GitHub IDs one by one, and click **Add** button.

Creating milestones, issues, and Wiki pages

To create milestones for your development will help better organize your team members, know your collaborators' progress, and help users/customers know what they could expect from a future release. Issues can be detailed action items team members would like to contribute to each milestone. Action can be bug fix, feature enhancement, etc. If you are still unclear what a milestone is or what an issue is, or you want to understand how milestone and issue tracking system can help software development, we recommended that you read about the features of GitHub issue tracking system at `https://github.com/blog/831-issues-2-0-the-next-generation` and `https://github.com/features/projects/issues`.

We recommend all the team members have a discussion on how to set the milestones, and what your first several initial issues will be (i.e., what they first couple of things you want to do at the beginning.) And the team leader does the following steps. **Remember that you are unlikely to submit all the issues to the tracking system at once, you may create new issues for bugs to fix or new features to implement, change the owner (assignee) of the issue, close implemented issues, or mark some as wontfix. All your team members are responsible to maintain your milestones and issues.**

1. To create a milestone, you can navigate to the **Issues** tab of your project homepage, then click the **Milestones** below the main menu bar. Now you are able to see the button **Create a new milestone**, and click it.

2. Type your title for the milestone, e.g., "M1" for the first milestone (you can also make it more meaningful and specific). Write a few descriptions for this milestone, like the goal and the brief descriptions of features will be enhanced in this milestone. Finally, select a "Due Data" for the milestone. Click the "Create Milestone" to finalize creating your milestone.

3. Do the previous step once again until all your milestones are created.

4. Now, you need to submit your first several issues. Go back to the **Issues** tab on your project homepage, and click **New Issue**. Type a title, assign the task to a particular team member, link this issue to a previously created milestone. Finally write down detailed comments to the issue.

   You may find when you type "@" followed by your collaborator's ID in the comment box, you are able to mention your collaborator as you mention your friend in a tweet on Twitter.

   You can learn how to write in GitHub Flavored Markdown language, by clicking the link above the textbox.

   You can also attch labels to each issue by selecting them from the right panel. As we mentioned earlier, you can change the milestone assignment, in particular, if you find you couldn't finish it by M1, then you can change it to M2 later, or you can also relabel it as **wontfix**.

Now, you can create a Wiki page for your team meeting minutes and other important items.

1. Click the "Wiki" tab at the top of your project homepage, and click **Edit Page** to start editing it.

Once you reach this point, remember to send us an email to report the URL of your project repository page (e.g., `https://github.com/COURSENUM-ID/project-teamID`).

Team coding with git-branch, git-merge, git-rebase

Collaboration is important for this homework. All the team members start their individual development after the team leader gets the framework ready, and pushes all his/her local commits to the repository. Once a team member finalizes his/her development, he/she might take responsibility on another task, or want to check the integrity or compatibility with features implemented by collaborators. After all the individual developments are done, team leader is responsible to merge all the newly developed components into the same codebase and test the integrity.

Git branching is a good tool to help the team manage parallel development and distributed codebase. In fact, the branching mechanism is widely adopted by not only git, but many other version control systems, e.g., SVN or Mercurial (Hg). But one of the most important reasons that people love git branching over SVN or Mercurial

(Hg) is its light-weight nature, which allow switching between development branches within the same clone of a repository.

Normally, the team leader is in charge of the `master` branch (the one you probably used for homework 0 and 1), which usually corresponds to a codebase for the most recent stable release, while team members should create branches for each individual task assignment, e.g., bug fixes, feature enhancement with `git branch` command. You can also do it within Eclipse by right-clicking the project name, and select **Team** → **Switch To** → **New Branch...**.

To merge multiple development branches back to `master` branch (or in Eclipse **Team** → **Switch To** → **master**), you should switch back to master, and then `git merge` the branch you want to be merged (or **Team** → **Merge...**). Sometimes, you may also need `git rebase` when you later realize your development should depend on another feature that was also being developed by your collaborator, which hadn't beed integrated in the `master` branch by the time you created your development branch.

To better understand git branching, you should read the "Git Branching" chapter of Pro Git Book (`http://git-scm.com/book/en/Git-Branching`). You can also find a more sophisticated branching model at `http://nvie.com/posts/a-successful-git-branching-model/`, which may be too complicated for this homework, but it can inspire how you want to manage your branches.

If you are looking for a Eclipse plug-in that can bring the GitHub issue tracking system to your workspace, e.g., create/close/comment issues, label them, assign to a person within eclipse, or automatically get notified by a message bubble if an issue is assigned to you, then you can try to investigate Mylyn plug-in. In the "Tips and Tricks using Eclipse with Github" post[4] , you will find the basic idea how Mylyn GitHub connector works. By default, Eclipse Juno for Java developers comes with EGit, Mylyn, and Mylyn Github connector, and the Task View is on the top-right corner of the Java perspective by default.

[4] `http://eclipsesource.com/blogs/2012/08/28/tips-and-tricks-using-eclipse-with-github/`

Proposal Outline

1. What questions will you focus on for your final system (exact answer generation part)?
    - Yes/No questions
    - Factoid questions
    - List questions
    - Summary questions
2. What technologies/ 3rd party tools do you plan to use?
3. Brief overview of the main components of the system. What is the Intelligent component in your system?
4. Critical aspects of the system/ Possible pitfalls in your system.
5. Evaluation/Error analysis of your system: a brief statement explaining how you will evaluate your method(s).
6. Team members and their andrew ids.