CSE 4/535-Introduction to Information Retrieval - Fall 2022
Project 1 Requirements
Due Date: 21st September 2022, 10:00 AM EST

Introduction

This project aims to introduce students to online conversation analysis (Reddit). An *online conversation* is defined as any exchange in the form of written text between multiple users (for example, submissions and comments). Before mining and analyzing such content, collecting, cleaning, and storing such data is a crucial step. This project aims to familiarize students with collecting online conversations and efficiently storing them, making data retrieval and analytics easier for downstream applications. This project will also introduce students to the different technical aspects involved in this course and subsequent projects.

[**Caution**]: Make sure you SAVE all the data you have collected/ indexed in a secure location; you will need this in Project 4.

Part 0: Setup

GCP: For this project(and the forthcoming projects), we will use Google Cloud Platform(GCP) as our default cloud platform. Requirements to get a GCP account with $300 credits:

- A valid Gmail account. Please DO NOT use your buffalo.edu email address.
- A credit/debit card.

[**Caution**]: Make sure you are not crossing your $300 limit. Otherwise, you may be charged. Please do not keep your instances running unless specifically asked to do so.

How to open a GCP account?
https://www.youtube.com/watch?v=W5mPX1-015o or watch any other youtube videos.

How to create a VM instance?
https://cloud.google.com/compute/docs/instances/create-start-instance
https://www.youtube.com/watch?v=goIKNBMqQhE or watch any other youtube videos.

How to open a port?
https://www.youtube.com/watch?v=-RjDWwTZUnc or watch any other youtube videos.

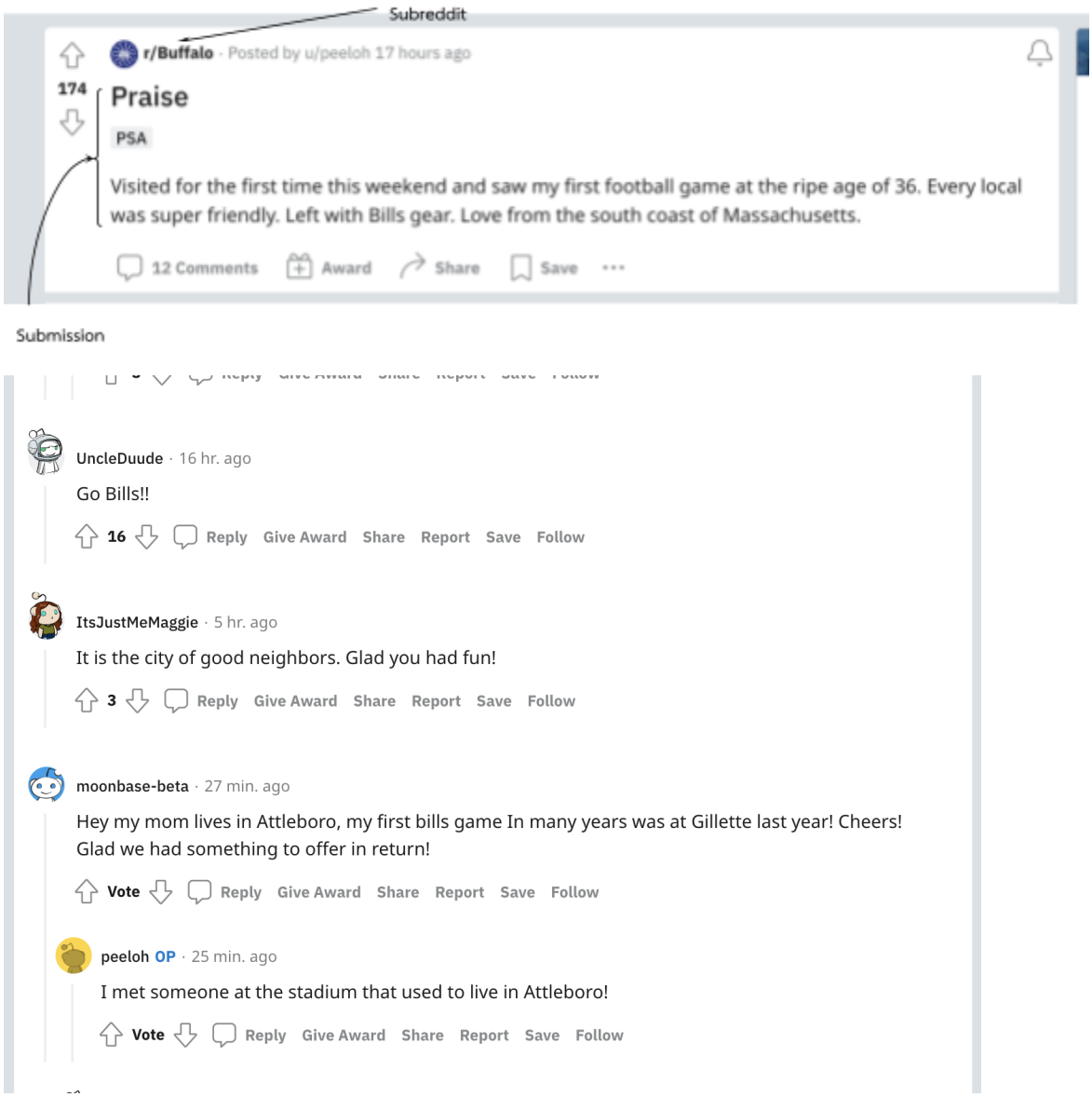
Apache Solr:
Installation:
https://tecadmin.net/install-apache-solr-on-ubuntu/

If wget is not available in VM, use this to install: sudo apt install wget
In GCP VMs, Solr runs in localhost, to run Solr which is accessible to public IP follow this link:
https://tecadmin.net/configuring-apache-solr-to-accessible-on-public-ip/

Tutorials:
https://www.youtube.com/watch?v=y2TqbjTc1GE&list=PLBrWqg4Ny6vXPalbTc_QqPiW1_G01AE2a

Part 1: Reddit Data Ingestion

We will ingest data from reddit.com using *pushshift* API (https://github.com/pushshift/api). Before that, let's understand what Reddit's data structure looks like:



https://www.reddit.com/r/Buffalo/comments/wubqcg/praise/

Data Ingestion Requirements:

1. Get a minimum of 100 submissions from each of these subreddits:
    - ExplainLikeImFive
    - FoodForThought
    - ChangeMyView
    - TodayILearned
2. Get a minimum of 200 submissions for each of these topics:

    - Politics, Environment, Technology, Healthcare, and Education.
    - You are required to come up with custom keywords to get submissions related to these topics.
3. Get a minimum of 2,000 submissions.
4. Get a minimum of 20,000 comments.
5. Not more than 5% of submissions containing [removed] or [deleted] in selftext
6. Not more than 5% of comments containing [removed] or [deleted] in body

API usage:

Endpoints:

- For fetching submissions: GET https://api.pushshift.io/reddit/search/submission
- For fetching comment_ids: GET https://api.pushshift.io/reddit/submission/comment_ids/<submission_id>
- For fetching comments: GET https://api.pushshift.io/reddit/search/comment

Follow the documentation for parameters: https://reddit-api.readthedocs.io/en/latest/#
You can also use psaw, follow this tutorial: https://alphascientist.com/reddit_part1.html

Part 2: Indexing

Before we describe the indexing process, we introduce some terminologies.

Solr Terminologies

- Solr indexes every document subject to an underlying schema.
- A schema, much akin to a database schema, defines how a document must be interpreted.
- Every document is just a collection of fields.
- Each field has an assigned primitive (data) type int, long, String, etc.
- Every field undergoes one of three possible operations: analysis, index, or query.
- The analysis defines how the field is broken down into tokens, which tokens are retained and which ones are dropped, how tokens are transformed, etc.
- Both indexing and querying at a low level are determined by how the field is analyzed.

Thus, the crucial element is configuring the schema to correctly index the collected tweets per the project requirements. Every field is mapped to a type, and each type is bound to a specific tokenizer, analyzer, and filters. The *schema.xml* is responsible for defining the full schema, including all fields, and their types, and analyzing indexing directives.

Although a full description of each analyzer, tokenizer, and filter is out of the scope of this document, a great starting point is at the following page: https://cwiki.apache.org/confluence/display/SOLR/AnalyzersTokenizersTokenFilters where you can find tips and tricks for all important elements you might want to use for this project. You are encouraged to start either in a schemaless mode or with the default schema, experiment with different filters, and work your way from there.

Prepocessing strategies

This is where students need to figure out the appropriate way to preprocess their collected tweets. Overall, there are two overarching strategies that you must consider:

- Using out-of-the-box components and configuring them correctly. For example, the StopFilter can filter out stopwords as specified by a file listed in the schema. Thus, at the very minimum, you would be required to find language-specific stopword lists and configure the filters for corresponding type fields to omit these stopwords.
- Preprocessing tweets before indexing to extract the needed fields. For example, you could preprocess the tweets to introduce new fields in the JSON response as per project requirements. Here again, it is left to your choice of programming language and/or libraries to perform this task. Solr supports a variety of data formats for importing data (XML, JSON, CSV, etc.). You would thus need to transform your queried tweets into one of the supported formats and POST this data to Solr to index.

Solr Schema API and PySolr

We will use Solr programmatically using Solr Schema API and PySolr, some documentation:

https://solr.apache.org/guide/6_6/schema-api.html
https://github.com/django-haystack/pysolr

More details will be provided in the upcoming classes.

Indexing Requirements

Create a core named `IRF22P1`. We will use the same core to index both the submissions and comments.

- For submissions

Field name	Description	Field Type	Is Custom?
id	Id of the submission	string	
subreddit	Name of the subreddit	string	
full_link	Full link of the submission	string	
title	Title of the submission	string	
selftext	The body of the submission	text_en	
author	The original poster of the submission	string	

is_submission	A flag denoted is this record a submission or not	boolean	Y
topic	Topic of the submission(applicable wherever needed)	string	Y
created_at	Convert the created_utc to datetime format (YYYY-MM-DDThh:mm:ssZ)	pdate	Y

- For comments

Field name	Description	Field Type	Is Custom?
id	Id of the comment	string	
subreddit	Name of the subreddit	string	
full_link	Full link of the comment	string	
body	The body of the comment	text_en	
author	The original poster of the comment	string	
parent_id	Id of the comment/submission for which this comment is made.	string	
parent_body	The body of the parent comment/submission	text_en	Y
is_submission	A flag denoted is this record a submission or not	boolean	Y
created_at	Convert the created_utc to datetime format (YYYY-MM-DDThh:mm:ssZ)	pdate	Y

Submission

You are required to submit a .json file containing the IP address of your EC2 instance, and the final set of keywords used by you. The file should be named project1_index_details.json .

Example file contents:

```
{
        "ip": "172.222.111.000",
        "port": "8983",
        "core": "IRF22P1"
}
```

[Caution] : Please check the validity of the JSON file(Tool to check validity: https://jsonlint.com/) before you submit. If you submit an invalid JSON file, a ZERO score is guaranteed (We mean it).

Steps for submission:

1. Create the JSON file, write details, and check validity.
2. Transfer your JSON file to any CSE servers. If you don't know how to transfer files to a server, please write an effective query and search google OR get your peer's help.
3. SSH to any CSE servers(Follow the link for more details:https://ubisec.cse.buffalo.edu/timberlake/)
4. Submit your file using the following command (You need to fire this command from the same directory where your JSON file is located in the server):
    a. For CSE 435 students: **submit_cse435 project1_index_details.json**
    b. For CSE 535 students: **submit_cse535 project1_index_details.json**
    c. On successful submission, the server will respond with a success message.
5. If we don't find your JSON file after the submission deadline in the server, a **ZERO** score is guaranteed (**We mean it**).

Grading

This project is worth **10 points**, and these are distributed as follows:

Criterion	**Description**	**Points**
Submissions Volume	Get a minimum of 2,000 submissions.	2
Min submission- by subreddit	minimum of 100 submissions from 4 subreddits = 400 min. submissions.	1
Min submission- by topic	Minimum 200 submissions from each of 5 topics = 1,000 min. submissions.	1
Comment Volume	Minimum of 20,000 comments.	2
Removed/Deleted	Not more than 5% of submissions/comments containing **[removed]** or **[deleted]** in **selftext/ body**	1
Solr sanity	Validate Solr instance runs + can run some queries	1
Solr schema validation	All fields are named as required, contain values as required, etc.	2

This project will be graded by an automatic grader. Not earlier than a week before the final submissions, an automatic grader may be made available as an API, which can be used to sanity check your submission.

Academic Integrity

Academic integrity policies will be taken very seriously. We will check the pattern of your indexed data, and if similarities are found more than a threshold value(which will not be disclosed) with other submissions, you will get a **ZERO** score. If you were found continuously arguing with us regarding your grades(unless there is a fault on our side), your grade would be reduced by up to 50% of your original score, and you will be reported to the department.