2022 Google Summer of Code CDLI Proposal
Towards a General-Purpose Entity-Bibliography Linking System

Personal Information:

Preferred Name: Circle Chen
Legal Name: Liangyuan Chen
Email: circlecly@berkeley.edu
Phone: +1 510-710-4613
Slack: Circle Chen
Timezone: GMT-7, Pacific Daylight Time
GitHub: https://github.com/CircleCly

Project Description

Currently, the bibliography data on assyriology is only linked with ancient artifact entities, but it also contains information about other entities, such as important places, kingdoms, languages, etc. In order to facilitate better organization of the database, I have in mind 4 goals of expanding the reference system:

1. Adapt the old reference system to the new one.
2. Clean the current bibliography data, and identify the different types of entities that may be extracted.
3. Generalize the entity reference system from bibliography: given a citation entry, the user should be able to navigate to the various types of entities using the links in the citation entry. This would conform to cakePHP conventions so that connections are bidirectional: in the entity page, the user should also be able to view which citation entries refer to this single entity.
4. Enable single publication file submission in the current bibliography upload system.

Interest

My realization of the power of open source began with the Python community: here, everything is open source, the language itself, numpy, pandas, or even more advanced libraries in machine learning like sklearn and Tensorflow. With open source, so many things can be achieved, since the tool is just out there in your reach. These libraries are also of high quality, due to the variety of contributors, and they are also free to use. When I discovered that my code on GitHub has been stored in the arctic code vault that can last for 1000 years, I realized that open source can be considered as the “cultural heritage” of modern humans. However, I have not contributed to open source projects before, so I feel both curious and excited that I will be able to do so in the GSoC.

My interest in working with the CDLI in GSoC arises from my previous work with Dr. Adam Anderson in the Data Science Discovery project at UC Berkeley, which I joined in the fall of 2021. Since then I performed OCR on Assyriology-related pdfs and created a citation tagger based on deep neural networks, and I realized how powerful computing could be used to process humanities data. However, I also noticed that the recognition of the potential of computation in the humanities could be enhanced. Therefore, I hope that my contribution to the CDLI can make computational tools become more accessible in the field of humanities.

.

Education and experience

- I've taken the following relevant courses as UC Berkeley:
    - Data 100 (Principles and Techniques of Data Science)
    - Data 8 (Foundations of Data Science)
    - CS 61A (Structure and Interpretation of Computer Programs)
    - CS 61B (Data Structures)
    - CS 70 (Discrete Mathematics and Probability Theory)
    - EECS 16A, EECS 16B (Designing Information Devices and Systems)
- Data Science Discovery Program, Ancient World Citation Analysis Project with Dr. Adam Anderson
    - Utilized Tesseract to perform OCR on assyriology pdf (GitHub link: https://github.com/ancient-world-citation-analysis/awca-ocr)
    - Improved existing LSTM citation tagger (GitHub link: https://github.com/ancient-world-citation-analysis/citation-tagger)
    - Utilized Sciwing to parse Keilschrift Bibliographie data (Google Colab Notebook Link: https://tinyurl.com/sciwing-keibi)
- High School
    - Prototyped a sign language translation defined based on Machine Learning
    - Prototyped a biometric authentication system using finger snapping
    - Contributed to UC Santa Barbara Summer Research Project of creating a vital signs detection system with infrared cameras and deep learning

Methods

Here are my current thoughts on how the proposed goals can be achieved:

1. Adapt the old reference system to the new one.

For the first step, it would be necessary to adapt the old reference system to the new one: there is currently an entity table called "artifacts" and a relationship table called "artifact_publications". To merge this and our new database, we can do the following:

- iterate through every row in "artifact_publications"
- for every row in "artifact_publications", insert a new row in "entity_publications"
- The new row would have table_name = "artifacts", entity_id = entity id in artifact_publications, publication_id = "publication id in "artifact_publications"

As can be seen, our new database structure is a generalization of the old one - it allows for relationships between all kinds of entities, but we can simply add "artifacts" as a type of entity to incorporate the old system into the new one.

After modifying the SQL table, It would also be essential to change the PHP files in the cakePHP application to handle reading from the new table. I will first look into the "PublicationsTable.php" as well as the "ArtifactsPublications.php", and then write up a new file called “EntityPublications.php” that adapts the logic in the “app/cake/src/Model/Table/” directory. Furthermore, it would also be necessary to update the controllers, namely reading the “PublicationsController.php” and “ArtifactsPublicationsController.php” and then coding up the “EntityPublicationsController.php” file.

After finishing the work on the backend, I would then move on to the frontend to modify existing code to account for the database modification.

When both backend and frontend parts have been completed, I would go through integration testing to ensure all the parts are still working. The add, edit, and view functionalities for publications should still be functional.

Then I would move on to goal 2: *Clean the current bibliography data, and identify the different types of entities that may be extracted.*

The first step is data cleaning. As I have noticed, a lot of the columns, including “title”, “book_title”, and “author”, is currently empty. To resolve this, we will primarily match publications with other bibliographies and projects. The first step for this is to collect full and clean bib entries. Then, we can use a dictionary-based approach to match as many entries as possible: : for instance, we could match the author’s names with those entries in the old database, where the author's name will often appear. After the matching, it is also necessary to human-check before merging.

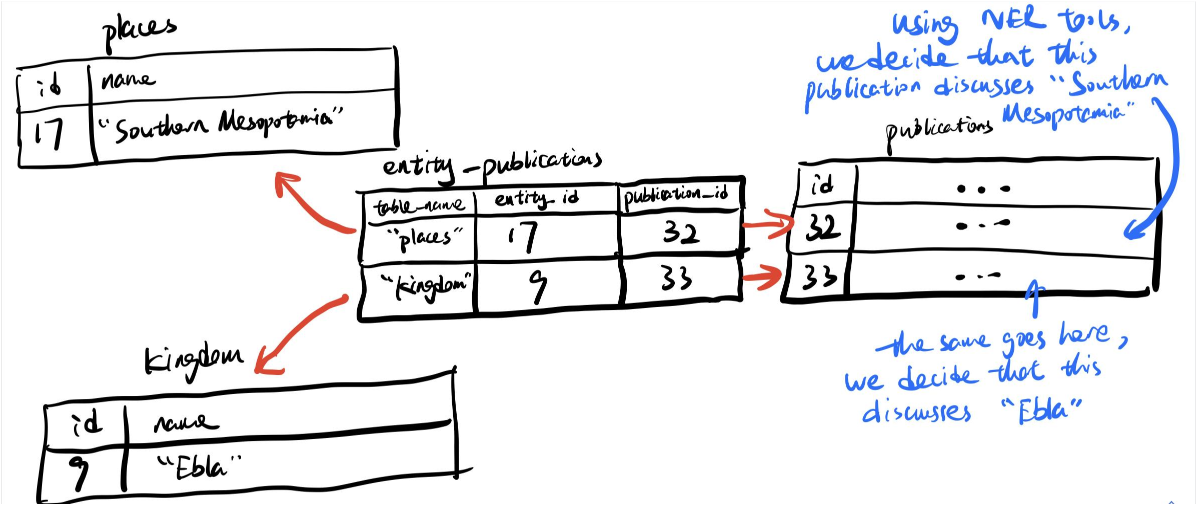
Another component of data cleaning is merging existing entries. For example, we may have ABb 13, 170 and ABb 13, 172 as separate bib entries, but these are actually referring to the same publication, only differing in exactly where to find the artifact. As a consequence, we need to store the 170 and 172 in the “exact_ref” column in the “entities_publications” relational table.

The next step after data cleaning will be to determine which entities does each citation in “publications” contain. First, we can run automatic NER tools (such as those provided in spacy or nltk) to identify the potential entities in these columns.Then, we can manually verify if the identified entities are meaningful, by comparing the output of the NER tool with the existing tables such as “dynasties”. However, one challenge inherent in this step is that when reviewing the automatically associated tags, we might not be able to remember all the details about a publication to decide if the publication discusses that entity as a main topic or not. To resolve this challenge, what I would do is to look into other bibliographies grouped by certain topics that has these publications (e.g. https://cdli.ox.ac.uk/wiki/doku.php?id=start, http://publikationen.badw.de/de/rla/index). I would also consult with domain experts on how to group the associations between publications and entities.

3. Generalize the entity reference system from bibliography: given a citation entry, the user should be able to navigate to the various types of entities using the links in the citation *entry. This would conform to cakePHP conventions so that connections are bidirectional: in the entity page, the user should also be able to view which citation entries refer to this single entity.*

Upon finishing goal 2, we will have cleaned the current bibliography data by filling in empty entries. Therefore, using the entities that are already included in our database, we can accomplish goal 3 by populating the table that describes relationships between entities and citations.

As shown in the graphic below, each type of entity would have their own table, and there will be an additional relationship table called “entity_publications”. Each row in this table would denote a mention of a specific entity in a specific citation entry. The “entity_publications” has these three columns: (1) “table_name”: where the entity table is hosted at (which also means it would describe the type of entity being mentioned), (2) “entity_id”: what the id of that entity in the entity table is, and (3) “publication_id”: what the id in the citation table is.



places

id	name
17	"Southern Mesopotamia"

entity_publications

table_name	entity_id	publication_id
"places"	17	32
"kingdom"	9	33

publications

id	...
32	...
33	...

kingdom

id	name
9	"Ebla"

Using NER tools, we decide that this publication discusses "Southern Mesopotamia"
the same goes here, we decide that this discusses "Ebla"

Once this new table is created, this database could be used to link entities mentioned in each citation entry to the corresponding entities on the website. It is also possible to view which publications refer to one single entity on the CDLI website.

After updating the database, I would also work on cakePHP web applications. I would modify the “EntityPublicationsTable.php” and models for each entity/table I would connect to in the “app/cake/src/Model/Table/” directory. I would also modify the “EntityPublicationsController.php” in the “app/cake/src/Controller” directory to ensure that it can handle representing more than one type of entity correctly. Again, these are all steps that should be done to ensure the new database can still be accessed from the frontend.

For the display of the reference data, I would aim towards at least displaying references for proveniences on the single page for proveniences. This should be done in an extensible way so future developers can adapt these to all entity types.

Following the coding, it is essential to also conduct a round of integration testing (in addition to the one after we adapted the old system to the new system), to ensure the existing functionalities are not broken.

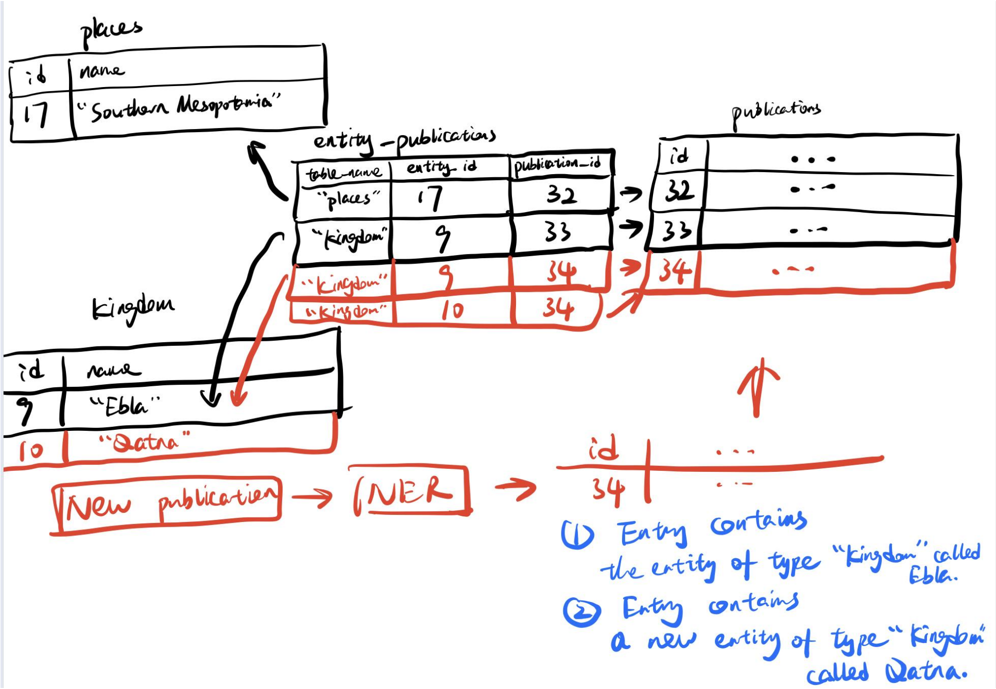
4. Enable single publication file submission in the current bibliography upload system.

For the backend side, the main challenge of this goal is how to reliably determine entities of interest in the newly uploaded bibliography. To do so, we would need to have access to the content of the article or book. One way to do this is to let the content contributors upload the contents of the actual book and article, and then perform NER on the content, extracting the entities that has more number of occurences than others. Another way is to let users provide URL of the publication, and grab the contents from there.

Once these entities of interest have been identified, they can be shown to the reviewer as potential suggestions to include in the database. The uploader can then determine whether to accept or reject the suggestion.

Since some new publications may refer to existing entities. After the NER, it is necessary to search in the corresponding entity tables to see if there are any matching entities with the new citation data, and build the connection with the existing entity. Only when there is no such entity would we append new entities in the tables.

This process is illustrated in the picture below: note that when we upload a new publication (along with the content pdf), if the NER pipeline decides it contains the entity “Ebla” and “Qatna” and our user accepts the suggestion, then our system would create a connection between this publication and “Ebla”, while adding another entity in the “kingdom” table with name “Qatna”.



places

id	name
17	"Southern Mesopotamia"

entity_publications

table_name	entity_id	publication_id
"places"	17	32
"kingdom"	9	33
"kingdom"	9	34
"kingdom"	10	34

publications

id	...
32	...
33	...
34	...

kingdom

id	name
9	"Ebla"
10	"Qatna"

① Entry contains
the entity of type "kingdom" called Ebla.
② Entry contains
a new entity of type "kingdom" called Qatna.

Frontend side:

To incorporate the upload system, it would be necessary to be verify and update controller php scripts in the “app/cake/src/Controller” directory, especially the new “EntityPublicationsController.php” to see if upload can be done correctly from there.

Bonus Task: Enable bulk-uploading in the system

For the sake of time we will focus on single file submission in this project. If there is extra time, we can work on (1) updating associations using references which are already associated with entities and (2) processing files in batches.

Documentation

During the development of this project, it is also necessary to write appropriate documentation accompanying the process of developmnt.

I plan to work on three documentation files: (1) The user documentation, for users who will look at the views. This serves to explain what they are seeing and how they can navigate from a publication to an entity, etc. (2) The editor/admin documentation, for the features of the new general purpose entity-reference linking system, such as how the files uploaded by users will be parsed, and how they can manage the new files by users. (3) The developer’s documentation, for future developers’ reference, such as how variables in the code are defined, which python libraries are used, and the testing of the program.

Timeline (350h)

Phase 0: Before bonding period (present - May 20th, 2022)

Explore how the backend and the frontend part is interconnected in this project, and read more tutorials to solidify HTML + CSS + Javascript.
Read cakePHP tutorials.
Start exploring python NER tools (spacy, nltk, pre-trained deep learning models for NER) and the CDLI database.
Exploring python tools that integrates the NER pipeline as a submodule to be queried by cakePHP.

Phase 1: Bonding period (May 20 - June 12, 2022)

Plan meeting schedules.
Discuss how frontend and backend would cooperate.

Phase 2: Coding, part 1 (June 13 - July 29, 2022)

Week 1-2 (June 13 - June 26, 2022):
Adapting the old referencing system to the new system. This would require:

- Migrating the “artifacts” and “artifacts_publications” to the new “entities” table.
- Modify the code of the website to account for database change.
- Integration testing to ensure that this does not break any existing functionalities.

Week 3-4 (June 27 - July 10, 2022):
Data cleaning

- Fill in the currently missing entries in the “publications” table, such as the author and title entries by looking e.g. at old databases.
- Filter out some entries if necessary.

Exploratory Data Analysis

- Perform NER on the current bibliography data, try different NER tools and compare the result.
- Analyze the different types of entities to be included in the entity datasets using other bibliographies and through consulting with domain experts.
- The analysis methods can be put into jupyter notebooks for reproducible cloud workflow.

Week 5-6 (July 11 - July 24, 2022):
Building the new entities and relationship SQL table.

- Construct new tables that would contain the new types of entities and relationships of the entities with the citation entries.
- Modify the website code to account for SQL table change (if necessary).

Week 7 (July 25 - July 29, 2022):
Integration testing with the frontend to make sure everything works together. This week also serves as the buffer week for coding phase 1 should anything unexpected happen.

Phase 3: Coding, part 2 (July 29, 2022 - September 12, 2022):
Week 1-2 (July 29 - August 14, 2022):
Implementation of the expanded upload function of the reference system on backend. Fine-tuning the NER pipeline to maximize performance of the reference system tagging, then deploy the pipeline as a submodule.

Week 3-4 (August 15 - August 28, 2022):
Work on the frontend to adapt the upload functionality to the current interface of the website. (Such as how to update the website based on the new dataset).

Week 5-6 (August 29, 2022 - September 12, 2022):
Buffer week reserved for integration testing and resolving existing issues.