New Updates

Having started on the technical bits, all progress made so far has revolved around testing Jaccard's metric as a relevant significance score as discussed with my advisor and implementing a dynamic container system in react to facilitate the first list of resources Amalgam should support, Images. The following Slides and Colab illustrate the implementations on a few LOs.

Find Colab Showing Jaccard's Metric in action: Jaccard's Score on LOs.

Find Slides Showing my base version of the React Container System: See updates past slide 7

Current proposal version:

Capstone Proposal

Abstract

The paper is an overview of the project Amalgam, software in development. The system aims to provide alternative formatting to study material such that distractions are minimized. Ease of use is a key feature thus a lot of the resource compilation is intended to be drag and drop. The style of development, use of modules and even the technologies used are all a reflection of the design philosophy that Amalgam is built on. I aim to make use of modules that already exist such as pdf.js and react.js to facilitate the breakdown of pdf documents to HTML and to automatically render the added resources without reloading the whole document. The end product will be a software that has a minimalistic design with no need for sessions to add to the convenience and to protect user data. It is a tool for the public and as such, all the code and documentation will be publicized. Options to publish compiled resources will be developed as add-ons once the system is refined and it satisfies the basic requirements set for it. For now, the main types of resources targetted are videos, audio, images, web pages and pdfs. Other formats like docx can be converted to these formats for addition to the system but support for them might be facilitated if the system catches on.

Basic Discussion

The project I am working on is tech-heavy. I intend to build a resource processing server that will allow for the standardization of study resources of various formats. I am exploring various techniques for output. At the moment I am fixed on producing JSON output containing links to the resources from a standardized repository. The system is supposed to strip the web of any tagged resources, eliminate any unnecessary code be it ad divs and such and present the final format in a manner suitable for a single, seamless webpage. The format of the page would be like jupyter notebooks to allow for extending the document.

From the description above, I would rely a lot on feedback from relevant professors. I have proposed my Capstone Advisor list based on the project scope. I will ask for a critique of prototypes at every stage for various aspects be it optimizations, resource formatting, database design and overall presentation. Atop this, I have peers who are adept with Software Engineering so I will use them to test out different backend modules I make use of and take their suggestions for systems to use. I will also keep my code open source to allow for public suggestion in case anyone gets interested in the system. In case the project ends up spawning a vibrant community, then the capstone will end up being an open-source system I manage as well as contribute to but for now, the basic project trajectory entails feedback from professors and peers.

New Additions

The proposal of the significance metric by the professor is something I am considering. I consider this a more feasible way of processing pdfs and larger files in lieu of opening them in an iframe. Having a widget that shows the significance of a text file relative to a subject paragraph is indeed a better approach with links to the full resource. Other potential features are highlights to regions wherewith more significance. Students are other academics can then auto-scroll to these preprocessed sections.

Timeline

Task	**Dates**
Research and module list: This matches the various resources to a javascript, python or java module for Android support, or I could use react.js for native cross platform development and this will facilitate the processing of these resources to normal xml. I will also find ways to convert these to JSON and other data token formats.	**April 20th -COMPLETE, CONTINUOUS**
Boilerplate Set up: Node.js package loading: Having identified my packages, I will set up a coding boilerplate with all these packages loaded up on Node.js. Some identified modules such as react work with Node.	**April 28th - COMPLETE, CURRENTLY DOING ITERATIVE BUILDS**
Front end design specification: I will create demos for the	**September 30th-WILL BE DYNAMIC**

expected front end design be it the page transitions, button layout, text layouts among others. These will be in the GitHub project repository as the md text	**DEPENDING ON DESIGN UPDATES**
Significance metric Based on the pagerank idea, I will incorporate Machine Learning and NLP by identifying indices I can judge the relevance of resources with beit video transcripts or more sensibly, pdf text. This is a new feature and will be implemented iteratively but the first phase is research.	**October 30th - 90% Completed. Dynamic and Subject to Change**
PDF breakdown and loading: The first line of development is for pdf. This will entail the extraction of information from pdf files and standardizing them in html format that is easy to work with. I will make use of existing modules, tweaking them to satisfy my format requirements.	**June 30**
Other resources support	**TBD**

Relevant LOs

LO	Relevance
#summarystatistics	These are statistics that will be shown on the live board. They are retrieved from distributions of time spent on the system and through Bayesian inference, we can evaluate the expected times if it were on alternative study media like the web. We can create such distributions from already existing data.
#communication	This is all about the documentation of code and the outlining of the development criteria. Code needs to be standardized and formatted such that the public contribute and the design specifications both for the front end will be documented int he front and back end.
#agile	The style of development being used is feature based thus the system undergoes continuous revision and upgrading to keep it up to date with technologies and requirements.
#sql	Given that we will need to save a lot of data, we need to define database architectures to serve as temporary storage for the system data. This will entail making structured queries hence the relevance of #sql

#abstraction	The user needs not to know how the system does the processing. They don't need to know how drag and drop work to show their data. All they need is to see the rendered page once it is added to the system and the functionality will be like a black box to them. In this regard, abstraction will be implemented as a pillar to usability and convenience.

Any other relevant HCs and LOs will be added as needed.

Sample Code

Jaccard Similarity

```
hc_context = """
the ratio of the number of outcomes in an exhaustive set of equally likely outcomes that
produce a given event to the total number of possible outcomes
When drawing and analyzing inferences about any expressive work—for example, a piece of
writing, a sculpture, a symphony, a song, a speech, or even a
scientific work—it is important to understand its context. To whom is the work addressed?
What historical events shaped it? How is it responding to other
works in its genre? What contributions does it make to its discipline? And how is it shaped
by its culture? All of these questions are important to answer
when developing an interpretation of a work.
"""
hc_probabilities = """
A probability specifies how likely it is that a specific event will occur. There are
different interpretations of probability,
which provide different frameworks for understanding claims about the probability of
various events. In addition, a conditional
probability is the probability of an event occurring given the occurrence of another event.
One important type of conditional probability
 is the probability based on a prior probability
```

```
 (the baseline or the degree of belief in the hypothesis prior to the new data). The
posterior probability is updated as new data are obtained.
 One must take care to identify the appropriate method to calculate the probability of an
event and to properly interpret probabilities.
 """
```

```
#Source:
https://www.mytutor.co.uk/answers/14587/GCSE/English-Literature/What-is-context-and-why-is-
it-important-when-studying-a-text-for-example-Romeo-and-Juliet/
context_example_resource = """
The simple definition of context is the background information surrounding a subject.\
When studying a literary text, context can apply to either historical context: what was\
taking place around the time a text was written (and in the case of Romeo and Juliet,
performed)\
, and how does this impact our reading of the text? You can specifically look at
historical, cultural,\
social or political contexts,
"""

# Source: https://whatis.techtarget.com/definition/probability
probabilities_example_resource = """
Probability is a branch of mathematics that deals with calculating the likelihood of a
given event's\
occurrence, which is expressed as a number between 1 and 0. An event with a probability of
1 can be considered a certainty: for example,\
the probability of a coin toss resulting in either heads or tails is 1, because there are
no other options, assuming the coin lands flat.\
An event with a probability of .5 can be considered to have equal odds of occurring or not
occurring: for example, the probability of a coin \
toss resulting in heads is .5, because the toss is equally as likely to result in tails. An
event with a probability of 0 can be considered\
an impossibility: for example, the probability that the coin will land (flat) without
either side facing up is 0, because either heads.
"""
```

```
import nltk

nltk.download('wordnet')

from nltk.stem import WordNetLemmatizer

import re



def preprocess(text):

    documents = []



    stemmer = WordNetLemmatizer()

    X = [text]

    for sen in range(0, len(X)):

        # Remove all the special characters

        document = re.sub(r'\W', ' ', str(X[sen]))



        # remove all single characters

        document = re.sub(r'\s+[a-zA-Z]\s+', ' ', document)



        # Remove single characters from the start

        document = re.sub(r'\^[a-zA-Z]\s+', ' ', document)



        # Substituting multiple spaces with single space
```

```
        document = re.sub(r'\s+', ' ', document, flags=re.I)



        # Removing prefixed 'b'

        document = re.sub(r'^b\s+', '', document)



        # Converting to Lowercase

        document = document.lower()



        # Lemmatization

        document = document.split()



        document = [stemmer.lemmatize(word) for word in document]

        document = ' '.join(document)



        documents.append(document)



    return document




def jaccard(text1, text2):

    intersection = len(list(set(text1).intersection(text2)))

    union = (len(text1) + len(text2)) - intersection
```

```
    return float(intersection) / union



def JaccardScore(base_text, resource_text):

    #Tokenize, remove punctuations, stem and standardize cases

    preprocessed_base = preprocess(base_text)

    pre_processed_resource = preprocess(resource_text)

    jaccard_score = jaccard(preprocessed_base, pre_processed_resource)

    print("JACCARD SCORE:",jaccard_score)

    return jaccard_score




print("\n Context v Context")

JaccardScore(hc_context, context_example_resource)

print("\n Probabilities v Probabilities")

JaccardScore(hc_probabilities, probabilities_example_resource)

print("\n Context vs Probabilities")

JaccardScore(hc_context, probabilities_example_resource)
```

Output
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Package wordnet is already up-to-date!

Context v Context
JACCARD SCORE: 0.02368692070030896

Probabilities v Probabilities
JACCARD SCORE: 0.015509103169251517

Context vs Probabilities
JACCARD SCORE: 0.018782870022539443
0.018782870022539443