INFSCI 2950

Independent Study in Systems & Technology

INDEPENDENT STUDY REPORT

ON

RECOMMENDING RELEVANT EDUCATIONAL VIDEOS FOR E-BOOK CHAPTERS

Term: Spring 2018

Nidhi Agarwal

INFSCI 2950

Independent Study in Systems & Technology

PROJECT OVERVIEW

- Course: INFSCI 2950 Independent Study in Systems & Technology
- Term: Spring 2018
- Advisor: Dr. Peter Brusilovsky
- Topic: Recommender System: provide e-book chapters with relevant online educational videos

OBJECTIVE

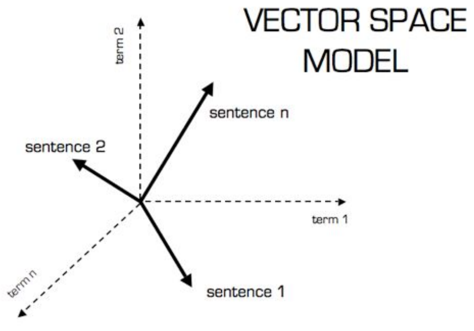
To develop a recommender system for an e-book reading portal, functioning at university server.
Here is the link: http://pawscomp2.sis.pitt.edu/ereader/home/
The online e-book reader currently helps students to annotate the important concepts, highlight them, add notes and participate in the quizzes in the end. In addition to these, and to improve the understanding of a concept, we wish to provide five to ten relevant YouTube video links for each concept at page level.

BRIEF DESCRIPTION

This tool was initiated and developed as a part of an Independent Study. The goal is to learn the utilization of documents' similarity distance calculation. Fortunately, since the tool seems to be useful for various projects involving with information retrieval area, it was used to intensively build the video recommendation provider for e-reader. It can be added, modified and upgraded to support the other systems better. This current version can be used as both standalone and a sub-module of e-reader depending on the availability of the data source. Using it as a standalone tool will be discussed more in the following topics.

TERM EXPLANATION

Document space is the representation of the set of documents. There can be different ways of representing documents, however mostly it is used to show the distance or the similarity between different documents. When documents are close each other in the document space this represents that they are similar or relevant document in the set. When documents are far apart each other in the document space this means they are not similar or relevant each other. Therefore, to represent documents in document space, the distances between documents are necessary.



Vector Space Model

Document distances can be calculated with several distance measurement methods. Mainly used methods are vector-based such as Euclidean measure and Cosine measure. In vector-based models, documents and query are represented with their weights which correspond to the importance of the term in the document. Here in our project, term weights are related to the frequency of the terms in the document. Each term in the document will have a *vector* value, by which the distance between documents will be calculated. These vectors are then used to compute the cosine similarity between documents.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

PRE-RESEARCH:

The following four research papers were read for understanding the basics, usage, implementation and how we can upgrade the e-reader. These are:

1) Assigning Educational Videos at Appropriate Locations in Textbooks (2014) Kokkodis et. al. : They introduced the problem of identifying a set of logical units in a textbook that best captures the content in a relevant educational video.

2) Study Navigator: An algorithmically generated aid for learning from electronic textbooks (2014) Agrawal et. al.: The study navigator for a section of the book consists of helpful *concept references* for understanding this section. Each concept reference is a pair consisting of a concept phrase explained elsewhere and the link to the section in which it has been explained.

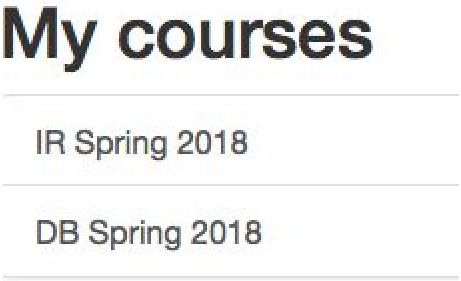
3) Mining Videos from the Web for Electronic Textbooks (2014) Agrawal et. al.: they explain the concept of formal concept analysis,which focus on combinations of concept phrases uniquely present in the section

4) Scientific Information Understanding via Open Educational Resources (OER) (2015) Liu et. al. : they elaborate how the online reading portals can be personalized as per user needs to improve the latter's learnability, but it has limitations.

TOOLS USED: Python 2.7, Django 1.9, Java 8 and MySQL, Unix

CURRENT INTERFACE

- The user goes to online e-reader, and selects the course:

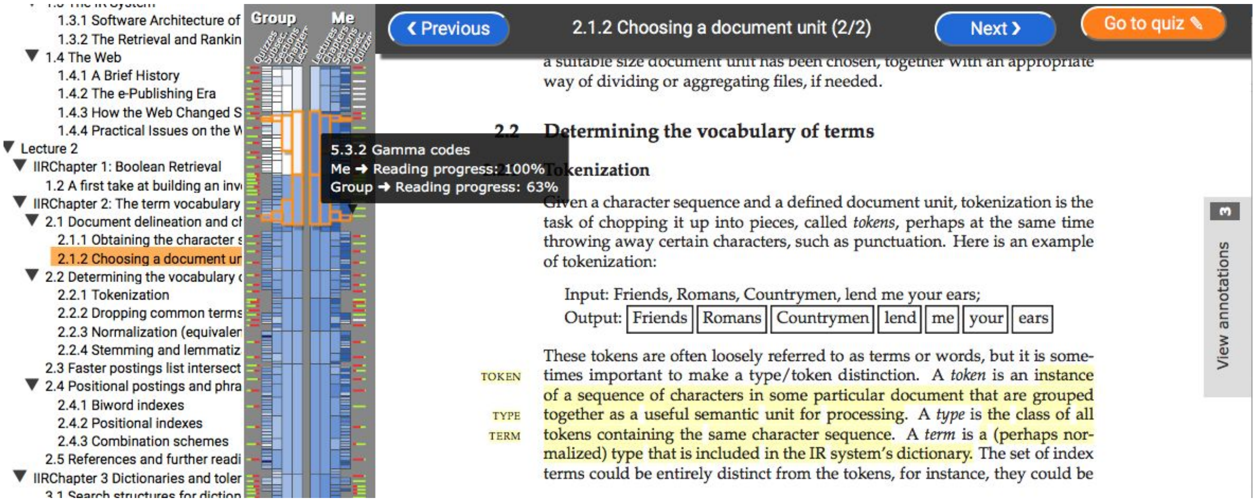


My courses

IR Spring 2018

DB Spring 2018

- The e-reader appears as follows, the reader can navigate to any section, highlight text and also annotate for future reference. The user can see their reading progress and also finish quizzes.



1.3.1 Software Architecture of
1.3.2 The Retrieval and Rankin
▼ 1.4 The Web
1.4.1 A Brief History
1.4.2 The e-Publishing Era
1.4.3 How the Web Changed S
1.4.4 Practical Issues on the W
▼ Lecture 2
▼ IIRChapter 1: Boolean Retrieval
1.2 A first take at building an inv
▼ IIRChapter 2: The term vocabulary
▼ 2.1 Document delineation and ch
2.1.1 Obtaining the character s
2.1.2 Choosing a document un
▼ 2.2 Determining the vocabulary o
2.2.1 Tokenization
2.2.2 Dropping common terms
2.2.3 Normalization (equivalen
2.2.4 Stemming and lemmatiz
2.3 Faster postings list intersect
▼ 2.4 Positional postings and phra
2.4.1 Biword indexes
2.4.2 Positional indexes
2.4.3 Combination schemes
2.5 References and further readi
▼ IIRChapter 3 Dictionaries and toler
3.1 Search structures for diction

5.3.2 Gamma codes
Me ➜ Reading progress: 100%
Group ➜ Reading progress: 63%

2.1.2 Choosing a document unit (2/2)

a suitable size document unit has been chosen, together with an appropriate way of dividing or aggregating files, if needed.

2.2 Determining the vocabulary of terms

Tokenization

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called *tokens*, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

Input: Friends, Romans, Countrymen, lend me your ears;
Output: Friends | Romans | Countrymen | lend | me | your | ears

TOKEN
TYPE
TERM

These tokens are often loosely referred to as terms or words, but it is sometimes important to make a type/token distinction. A *token* is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing. A *type* is the class of all tokens containing the same character sequence. A *term* is a (perhaps normalized) type that is included in the IR system's dictionary. The set of index terms could be entirely distinct from the tokens, for instance, they could be

- we wish to add links on these pages, so that a reader can refer them simultaneously. This is expected to improve their learning of the concepts.

TASKS ACCOMPLISHED:

1. *Video Search:* the videos are currently searched manually,ie by putting the key-words in search box and retrieving the top results. For example, to find the videos about “for-loop”, it is searched on youtube, and the links are stored. Later, using an API, those video’s title,tags,description (metadata) is retrieved and stored into a json-format file. This forms the video corpus.

2. *Concept Annotation:* manual annotation of important concepts in each chapter on the basis of certain rules is done. This helped in generating the ground truth, ie identifying whether the concepts: are prerequisite, discussed in detail, or just mentioned, so that it will be discussed in later sections. This will help in improving the performance of the recommender system by discarding the concepts not relevant to IR.

3. *Document similarity using Java*: computed the cosine similarity between IR book and video description. This was then utilized to rank the videos on the basis of similarity to each page of the book. The top ten videos are mapped to each page.

   OVERVIEW

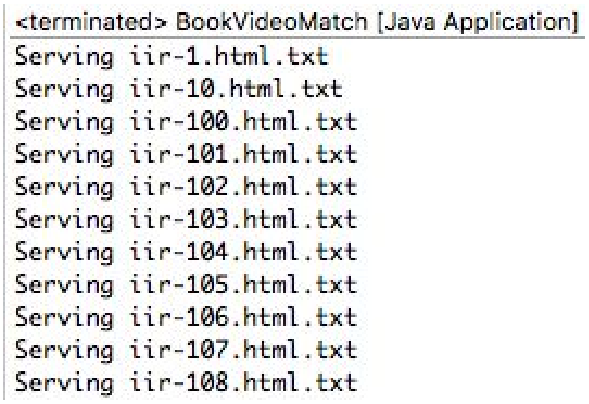
   Java libraries Indri and Lucene are used for computing the similarity between video and IR book’s text.

   DESIGN

   These documents are pre-processed. They were stemmed, stop-words were removed and then tokenized.After this, for each term: term frequency and inverse document frequencies were calculated. This led to calculation of cosine similarity video and IR book.

   IMPLEMENTATION

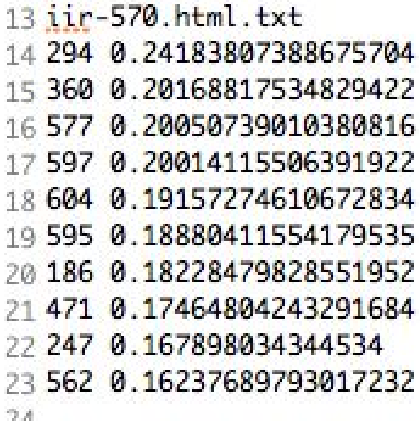
   The following screenshot shows the output of a java programme, computing the similarity of each iir book page with all the video texts. There are 578 pages and 644 videos. It takes around 1.5 minutes to compute the similarity.



```
<terminated> BookVideoMatch [Java Application]
Serving iir-1.html.txt
Serving iir-10.html.txt
Serving iir-100.html.txt
Serving iir-101.html.txt
Serving iir-102.html.txt
Serving iir-103.html.txt
Serving iir-104.html.txt
Serving iir-105.html.txt
Serving iir-106.html.txt
Serving iir-107.html.txt
Serving iir-108.html.txt
```

Screenshot- 1

The following screenshot shows the list of top ten videos mapped with each iir book page.At the top, is the name of the page. In the next line, the number of the video followed by its similarity score with the iir page.



```
iir-570.html.txt
294 0.24183807388675704
360 0.20168817534829422
577 0.20050739010380816
597 0.20014115506391922
604 0.19157274610672834
595 0.18880411554179535
186 0.18228479828551952
471 0.17464804243291684
247 0.167898034344534
562 0.16237689793017232
```

Screenshot- 2

The e-reader is implemented using python django framework, hence the programme is re-done using python.

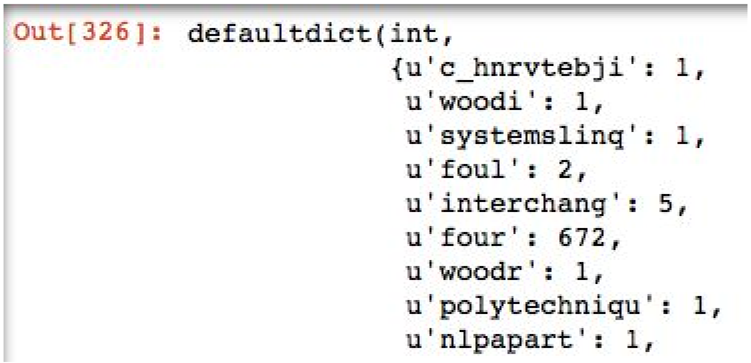
4. *Python Django:* learned Python Django for web development from scratch along with MVT framework. The mvt refers to model-view-template framework. It provides a secure connection with database, web servers and reusability of code. All the programme modules are called apps, and are like a standalone tool, which can be used with another project easily.

OVERVIEW

Python libraries nltk, numpy, scilearn, spacy, gensim are used for computing the cosine similarity between book and video text.

DESIGN

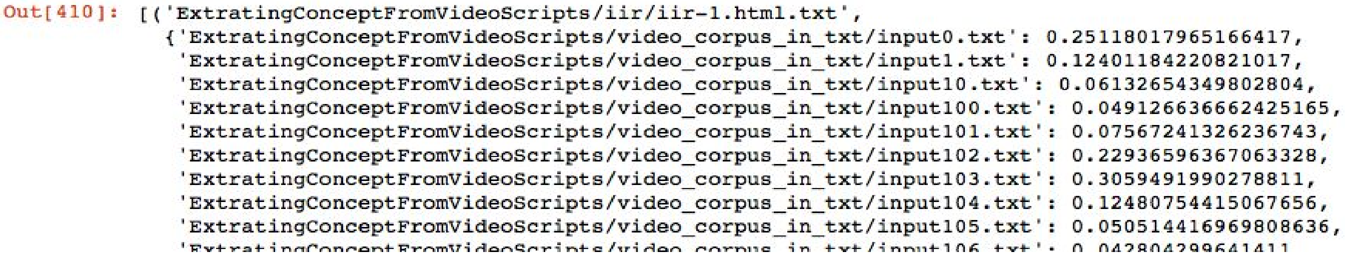
Functions are created for the following tasks: pre-process, get similarity, mapping videos to ir pages,and ranking of videos on the basis of highest similarity.The pre-processed corpus looks like :



```
Out[326]: defaultdict(int,
                      {u'c_hnrvtebji': 1,
                       u'woodi': 1,
                       u'systemslinq': 1,
                       u'foul': 2,
                       u'interchang': 5,
                       u'four': 672,
                       u'woodr': 1,
                       u'polytechniqu': 1,
                       u'nlpapart': 1,
```

IMPLEMENTATION

The following screenshot shows the similarity scores for each video with iir-1.txt file:



```
Out[410]: [('ExtratingConceptFromVideoScripts/iir/iir-1.html.txt',
            {'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input0.txt': 0.25118017965166417,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input1.txt': 0.12401184220821017,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input10.txt': 0.06132654349802804,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input100.txt': 0.04912663662425165,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input101.txt': 0.07567241326236743,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input102.txt': 0.22936596367063328,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input103.txt': 0.305949199027881 1,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input104.txt': 0.12480754415067656,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input105.txt': 0.05051441696980863 6,
             'ExtratingConceptFromVideoScripts/video_corpus_in_txt/input106.txt': 0.04280429964141 1
```

FUTURE WORK:

- The similarity scores are computed, they need to be sorted. After which the top most videos will be mapped to each IR book page.
- dynamic retrieval of relevant youtube videos for each section
- deployment of the ereader with the recommender on the university server.

REFERENCES:

1. https://youtu.be/D6esTdOLXh4
2. https://youtu.be/HW9W6EBytLg
3. https://en.wikipedia.org/wiki/Cosine_similarity
4. http://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html
5. https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.distance.cosine.html