RESEARCH CENTER

Predicting Publication Date: a Text Analysis Exercise over 250,000 Volumes in the HTRC Secure HathiTrust Analytics Research Commons

Use case: RDA Digital Humanities Workshop, May 2015

The HathiTrust digital library has over 13 million digitized volumes (books) contributed from research libraries. The corpus dates back to before the 1500's, includes editions in 127 languages of which over 8 million of the volumes are in copyright. HathiTrust approved the creation of the HathiTrust Research Center in 2011 to provision computational investigation (e.g., text mining) over the full corpus of copyrighted and non-copyrighted content. The challenges of provisioning a research commons to support secure analytics over objects in the HT digital library is the subject of a second use case. This use case gives a brief overview of the mainstream tools and services of HTRC as they currently exist in HTRC's "SHARC", Secure HathiTrust Analytics Research Commons (SHARC)[1], before moving on to a discussion of a use case.

Digital humanities (DH) researchers frequently make use of the raw OCR text of HTRC for computational analysis. The DH researchers typically query the corpus to identify one or several subsets with which to work. These subset collections form the basis of personal "worksets", for working with in the SHARC environment. Analysis is not limited to:

1) Statistical analysis, such as word count, word tag cloud, and word use trend over time
2) Computational linguistics analysis, such as grammatical roles of words in a sentence, named entity recognition
3) Derivative analysis, such as deriving a topic modeling or language model from a workset
4) Machine learning, for example, training a model of classifying an author's work.

Researchers use either the analysis tools built into SHARC (these analysis tools currently limited to SEASR), or can run their own analysis tools using a dedicated virtual machine; the latter through the HTRC Data Capsule option (see companion piece for HTRC Data Capsule).

HTRC Architecture

HTRC SHARC provides a portal through which a research logs in. Once logged in, access is to

Workset builder. Researcher builds worksets using the Blacklight interface, submits the workset to run against one of the default analytical tools, results are displayed in the browser or downloaded.

Analysis Tools: the SEASR suite of analysis tools are available. Forthcoming is a Bookworm interface, and R.

Data Sets: Data bundle of extracted features for the 4.9 million volumes currently in the public domain.

Data subsystem: the data subsystem is served through REST APIs, most notably:

- *Data API*. Access to page level OCR and METS metadata documents.

[1] http://www.hathitrust.org/htrc

- *Solr API*. Full text index containing volume and page level OCR as well as MARC metadata fields, e.g. author, title, publish date, subject. The full text index can also carry out certain forms of analysis itself.

Data Capsule. HTRC Data Capsule is secure environment where users can create virtual machine for accessing volumes and performing customized analysis. The data capsule design has maintenance mode for software installation and secure mode for interacting with sensitive content without content leak.

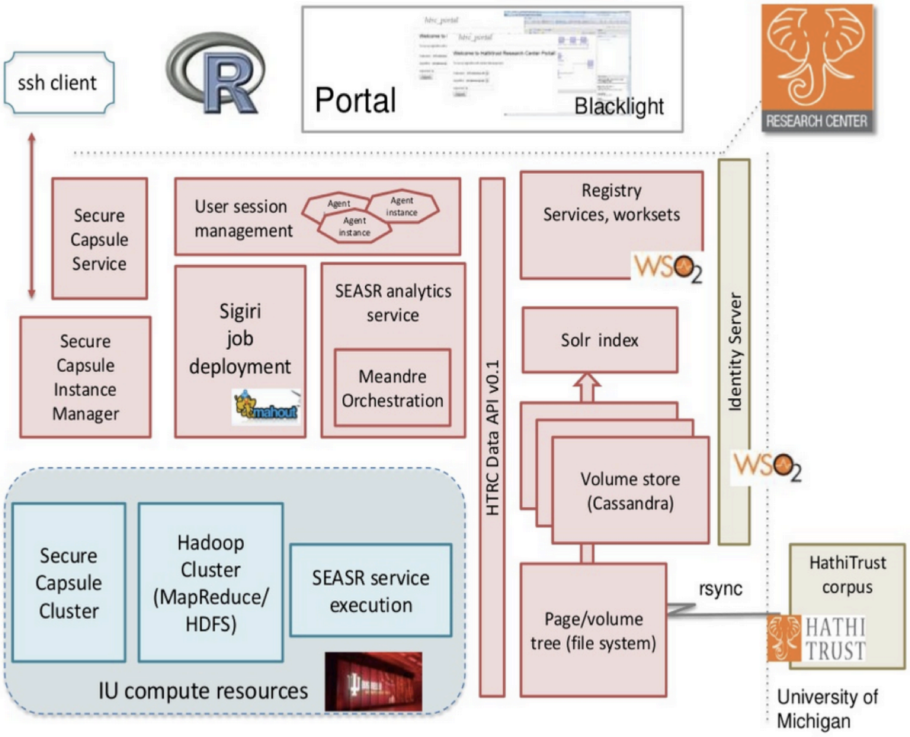

Figure 1. HTRC architecture.

Use Case: machine learning using HT corpus

The research question is this: *can the body of a text be mined to accurately predict book publication time where that information is missing in the catalog record?* The work is motivated by a non-trivial number of catalog records in HathiTrust that are incomplete and in some cases inaccurate. From the data set the authors used, a full 13% of publish date values are missing.

The approach is to extract temporal features from the early part of volumes. These features are used to train a model that can then predict a publish date. The researchers first obtained a data set from HTRC called an “open-open” corpus, called such because it contains volumes in the HT digital library that are neither under copyright nor digitized as part of the Google Books project. The open-open corpus has around 250,000 volumes with publication dates ranging from pre-1600 to 2000.

The researchers experimented with different sets of features to find the most effective ones for predicting publish time. Features include: first date in text, three text similarity metrics combining n-grams, OCR error counts. Results show that these features can effectively determine publication time given some unseen text, with the best F-score of .86 *(Guo, Edelblute, Dai, Chen, & Liu, 2015)*

To accomplish the research, the researchers used the HTRC Data API to fetch OCR content, and Solr API to obtain volume metadata including publish date. The text features extracted from the text and fed for further text analysis are (as shown in Figure 1):

- Bag-of-words, or unigram: word frequency counts
- Bi-grams: frequency counts of two-word phrases
- Tri-grams: frequency counts of three-word phrases
- OCR errors: count of manually identified OCR errors

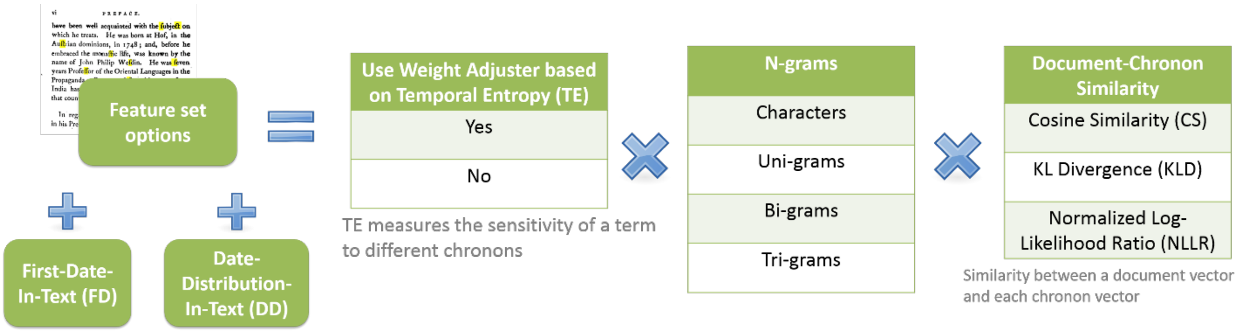
Feature set options
First-Date-In-Text (FD)
Date-Distribution-In-Text (DD)
Use Weight Adjuster based on Temporal Entropy (TE)
Yes
No
TE measures the sensitivity of a term to different chronons
N-grams
Characters
Uni-grams
Bi-grams
Tri-grams
Document-Chronon Similarity
Cosine Similarity (CS)
KL Divergence (KLD)
Normalized Log-Likelihood Ratio (NLLR)
Similarity between a document vector and each chronon vector

Figure 1. Combinations of feature sets for computing text similarity.

These features are then used to compute the three types of similarity metrics, which are cosine similarity, KL Divergence, and normalized log likelihood ratio. These similarity values are further fed to machine learning algorithms as model features to derive models for predicting publish time of texts (workflow shown in Figure 2).

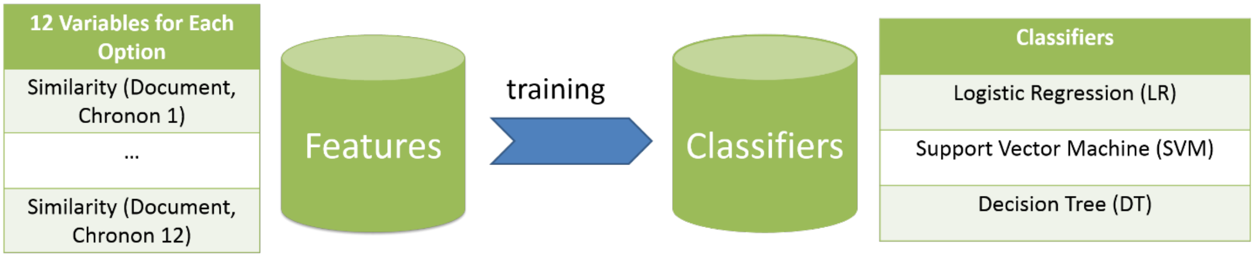
12 Variables for Each Option
Similarity (Document, Chronon 1)
...
Similarity (Document, Chronon 12)
Features
training
Classifiers
Classifiers
Logistic Regression (LR)
Support Vector Machine (SVM)
Decision Tree (DT)

Figure 2. Machine learning workflow.

Text analytics usually make use of extracted features, such as bag-of-word, n-grams, part-of-speech tagging, named entities, instead of the full text per se. That means if a study can identify what features are needed to answer its research questions, then the study can be non-consumptive by only using these derived features. For example, the research above can use only the features listed above without obtaining the full text.

Challenges

Use Case Table

ID:	
Title:	A HathiTrust Research Center (HTRC) Use Case
Description:	Architecture for non-consumptive text analysis on HathiTrust volumes
Trigger:	-
Preconditions:	Researchers want to analyze HT volumes to answer their research questions, by moving computation to data and not taking data away
Steps for Main Success Scenario:	1. Researcher propose their research questions, and identify how text analysis can ansewr the questions. 2. Identify what text analytics algorithms and what features are needed from HTRC for such analysis; also identify whether full text is needed. 3. Create workset by querying Solr search engine 4. Run text analytics on the workset, either in portal for simplicity or in Data Capsule for customized analysis 5. Obtain analysis results and interpret
Alternate scenarios:	-
Postconditions:	May need to read full text for interpreting results
Frequency of Use:	The analysis can be done as frequent as needed
Status:	Draft

Author:	Beth Plale, Miao Chen

References

Guo, S., Edelblute, T., Dai, B., Chen, M., & Liu, X. (2015). Toward Enhanced Metadata Quality of Large-Scale Digital Libraries: Estimating Volume Time Range. *iConference 2015 Proceedings*.

HathiTrust Research Center. http://www.hathitrust.org/htrc

Jockers, M. L. (2014). *Text Analysis with R for Students of Literature*. New York: Springer.

Zeng, J., Ruan, G., Crowell, A., Prakash, A., & Plale, B. (2014, June). Cloud computing data capsules for non-consumptiveuse of texts. In *Proceedings of the 5th ACM workshop on Scientific cloud computing* (pp. 9-16). ACM.