nora: summer 2005 plan of work for mith/hcil

The white paper that follows is the outcome of the June 9-10 nora meeting in College Park attended by Loretta Auvil, Tanya Clement, Matt Kirschenbaum, Greg Lord, Catherine Plaisant, James Rose, and Martha Nell Smith.

Summary

We plan to integrate a labeling and visualization environment with machine learning and document classification. The objective is to support the kind of scholarly investigations that have emerged through interest in such topics as erotics, ekphrasis, and sentimentality. Our prototype will integrate Fekete's InfoVis Toolkit, HCIL's Piccolo, and D2K/T2K, and be available via D2K Web services or D2K SL.

Rationale and Principles

From the standpoint of text mining, the Dickinson "hot or not" exercise is an orthodox classification problem. Several other professed areas of interest (ekphrasis, sentimentality) suggest that document classification (known popularly through "more like this"-type features) has significant appeal within the humanities research domains. After a question or hypothesis has been formulated, an initial document set can be read and labeled by an expert user. This labeled set becomes the training sample with which classification attempts of a large number of new documents can be carried out by machine learning tools. Users, then need an environment in which results can be displayed, reviewed and labeled. This is an iterative process in which training sets are revised, and multiple classification requests are generated with varying parameters.

Humanists don't like black boxes. We shouldn't ask our audience to accept one here. An important component of nora should therefore be to make the classification process comprehensible. In a paper entitled "The Visible Computer" (http://imv.au.dk/~pba/Homepagematerial/publicationfolder/VisComp.pdf) Peter Bogh Andersen develops a rationale for "technical systems that support the operators in understanding what actually happens in the system." Here we propose something like "visible classification." Our hypothesis is that powerful yet simple user interfaces and meaningful visualizations will allow users to understand the low level primitives which govern the text mining's actions, and keep a sense of control over the process of classification while developing an appropriate level of trust. Based on interaction with the visualizations, classification labels might be added or adjusted. Fekete's InfoVis toolkit includes scatter plots, time series, parallel coordinates, treemaps, icicle trees, node-link diagrams fortrees and graphs and adjacency matrices for graphs. All visualizations can use fisheye lenses and dynamic labeling. We will also use Piccolo, the Zoomable User Interface (ZUI) technology pioneered at HCIL which is likely to be critical to working with very large datasets and clarifying changes in data representations.

Process

The Maryland group currently has in hand a simple Java-based visualization tool designed for the Emily Dickinson corpus and programmed by a student in spring 2005. The prototype (named "Emily") can parse a collection of TEI encoded XML documents and provides 2 views of the collection: 1) a textual view, here showing the body of each document (Figure 1) and 2) an abstract overview of the collection (Figure 2).

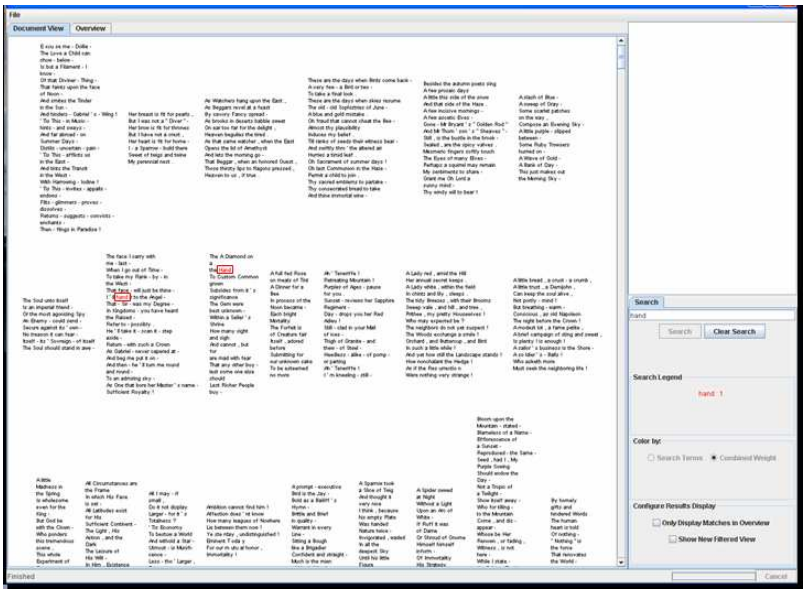

Fig. 1: In this view users can browse the text of the entire collection. Clicking on a document displays the original manuscript.

Each bar represents a document. The length of the bar is proportional to the length of the document. The documents are ordered and displayed in multiple rows. The order of the documents could be mapped to any attribute which could be extracted from the document (e.g. date). Other attributes of the documents could be used as well to layout the overview. A zooming transition from the textual view to the abstract overview could help users understand the correspondence between data representation.

A rudimentary search is included to illustrate how results can be displayed on the overview to display results of searches or other analysis. For example Figure 2 illustrate how the results of a string search can be shown on the overview (but the same view could show where repetitions occur in the documents, or where the use of punctuation is high

Obviously other visualizations of the entire collection could replace this simple view.

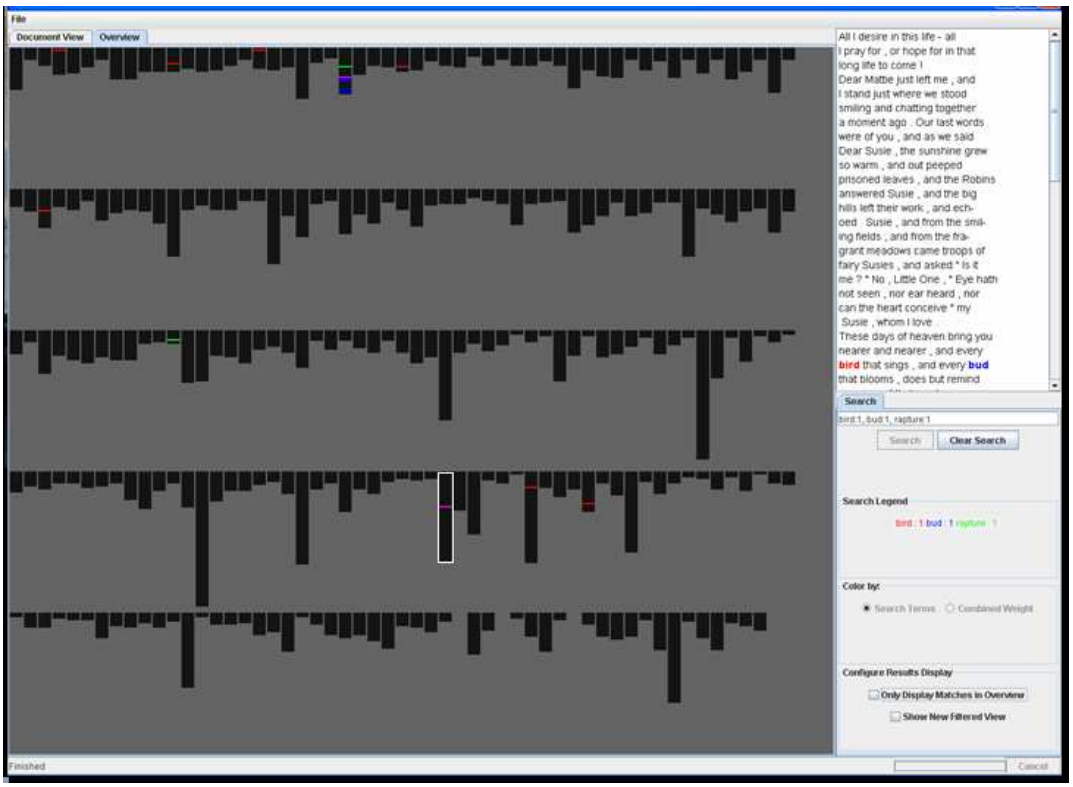

Fig. 2: An abstract view of the entire collection. Each bar is a document, the length of the bar is proportional to the number of lines in the document's manuscript.

In this current form the usefulness of the visualization is severely hampered by a weak data matrix (i.e. the unavailability of metadata about the documents) . In other words we could only used the length of the document and the order of the files in the collection's directory. In fact it is little better than a simple list..

We propose augmenting this tool to support more robust data matrices and queries in several important ways (i.e. strive to visualize a lot more metadata about the documents, coming either from the xml tags (author, type of document, publisher, author gender? etc,) or from text analysis tools we find relevant for the analysis examples we chose (e.g. rate of punctuation, parts of speech, repetitions etc.)

These are loosely grouped into two parallel development paths:

1. Integrating manual and machine learning labeling
2. Improving the visualization set and user interface

1. Integrating manual and machine learning labeling techniques

We loosely call "labeling" the user's task of determining which documents in a collection have a given characteristic of interest (e.g. documents that seem to include erotics, or sentimentality). Labeling the documents can be a tedious and expensive task. We propose to use active machine learning so that the labeling processed can be guided and

the number of documents that need to be reviewed significantly reduced. Machine learning will suggest "more documents like the ones you labeled already" but can also suggest which documents need to be manually classified in order to improve and speed the process.

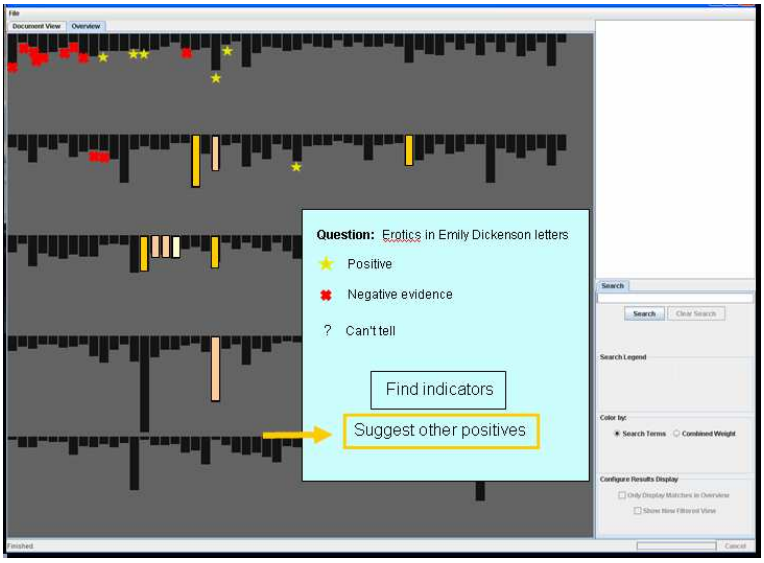

Fig. 3 The user first labeled some of the documents, for example here he marked with a yellow star a set of positive "hot" documents for the question being studied (i.e. the erotics of E.D.) Then the user asks the system to "Suggest other positives". An automatic call to D2K supplies the training set to D2K which returns a list of suggested new "hot" documents. They are shown on the overview as yellow bars. The bright yellow bars indicate high confidence, and the paler yellow indicate low confidence. Users can then review the text of the document and label the document, either confirming or rejecting the suggestion.

The "model" of the classification can also be applied to identify indicators of one class or another, e.g. a ranked list of words to be present in erotic documents , or particular tags such as the condition tag.

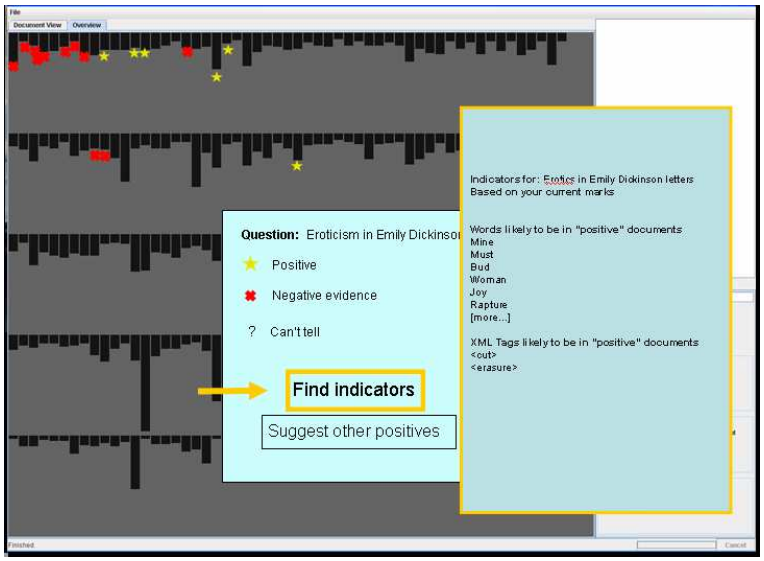

Fig. 4: an alternative is to request suggestions of possible indicators. Here the example show a list of words found to likely to be present in "hot" documents. Next, users could explore this list by reviewing the documents which use those words, and annotating the list of word/indicators as to it's correctness.

Text mining techniques will be applied to generate the attributes that can be used in the above modeling approaches. In the beginning, some assumptions will be made on how the documents should be processed (i.e. by setting reasonably good default values and options). Plans will be made for advanced modes to become available that allow users to setup the reprocessing of the documents with different parameter settings.

2. Improving the visualization set and user interface

We plan to build an interface that allows the tagging/labeling (such as positive and negative) of documents. The researcher will propose the question they are seeking to investigate and indicate the 2 classifications that they are seeking. The current visualization is inspired by SeeSoft. Each document is represented by a vertical bar (sized by the length of the document). Multiple rows of documents are shown above. Searching for particular words can be done and highlighted. The extension of this project will use Piccolo, InfoVis Toolkit and D2K/T2K. The documents will be color coded by the labeling. We want to create a way to prompt the user for the next document to label using the active learning approach. We also want to use the machine learning modeling mentioned above to suggest other documents that are in the positive class (or negative class) and to provide a ranking of the words that the modeling finds as indicators.

Data

There are several data sets in play (see Appendix A).

D2K pre-processing creates the sparse tables necessary for the classification process. At present the relationship between D2K and Tamarind in the pre-processing stage stills seems unclear, so we make no assumptions about the source of a sparse matrix for actual text mining.

Question for Steve: can you write a query that will pass a Tamarind database to D2K in the form of a sparse table?

Deliverables

In the fall (by October we hope) we will have a D2K module that can be run through SL and/or D2K Web Services interface. The module will be combine elements of the current prototype, the InfoVis toolkit and Piccolo, and allow real time access to the machine learning D2K algorithms:

Rough Plan of Work (priority ordering might change)

- Familiarization with various technologies: D2K, Piccolo and the InfoVis Toolkit. Definition of interchanged data formats.
- Integration of InfoVis Toolkit and Piccolo components in the Emily prototype
- Development of alternative overviews of the collection using additional metadata
- Development of D2K analysis modules for labeling and suggestion of indicators (i.e. refine Bei's modules).
- Connection of Emily interface to D2K analysis modules
- Provide online access thru either Web Services (similar to Phylomat) or SL

Additional Testing

We intend that the Emily prototype can be generalizable for use by the sentimentality project. Most likely this will require visualization variants, additional metadata about the documents or portions of the documents, and new analysis D2K modules. We will rely heavily on UVa for this

Final Remarks

Unlike most academic constituencies, in the humanities neither the value of text mining nor visualizations are well understood. Many scholars will be unaware that such tools and technologies even exist. Others will be mistrustful of their quantitative or perceived mechanistic nature. Of course we hope some will also respond with imagination and enthusiasm. But nora must cater to all of these audiences, and thus our work should involve a strong evangelical component, by which we seek to educate and cultivate the humanities as a user community for these technologies.

In the fullness of time we should think carefully about allow nora to evolve in the social software world of RSS and bottom-up "folksonomies" of knowledge. The humanities are driven by conversation and argumentation and these are inherently social activities. In practical terms we can easily imagine extending Emily so that it saves the labeling results and even the history of the labeling process. Users annotations can also be collected and saved in personal workspace or shared in collaboratories (possibly using TAPoR). A nora that allowed users to see how ten prominent Dickinson scholars labeled erotics in a set of poems would make nora a center of intellectual conversation and attention. This is how computational resources can become central rather than peripheral to the conversation in the humanities.

Appendix A: sample data structures

1. This was generated by Steve at the request of the Maryland group, but never actually put to use:

> From: Steve Ramsay <sramsay@uga.edu>
> Reply-To: Mellon Visualization Project <MELLON-VIZ@listserv.umd.edu>
> To: MELLON-VIZ@listserv.umd.edu
> Date: Apr 12, 2005 9:49 AM
> Subject: [MELLON-VIZ] the data
>
> What you see below is the file in which the phrase was found, followed by a "verbose" XPath expression indicating its precise location in the document. You'll notice
> that the XPath contains more information than there was in the original file (e.g. markings
> for sentences and individual tokens).
>
> hand: 13 instances
> ---
> DEAmsEDCSHDh356.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/lg[1]/Sentence[1]/l[6]/Token[2])
> DEAmsEDCSHDh325.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/lg[2]/Sentence[1]/l[8]/Token[1])
> DEAmsEDCSHDhb127.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/div1[2]/lg[2]/Sentence[1]/l[2]/Token[6])
> DEAmsEDCSHDhl15.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/div1[1]/p[1]/Sentence[2]/hi[2]/Token[1])
> DEAmsEDCSHDh376.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/lg[2]/Sentence[1]/l[2]/Token[1])
> DEAmsEDCSHDh223.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/lg[1]/Sentence[1]/l[1]/Token[2])
> DEAmsEDCSHDh301.1.xml
> (/TEI.2[1]/text[1]/body[1]/div0[1]/lg[1]/l[2]/Sentence[1]/Token[3])

DEAmsEDCSHDhl17.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/lg[3]/l[1]/Token[5])
DEAmsEDCSHDhb154.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/lg[1]/Sentence[1]/l[10]/Token[5])
DEAmsEDCSHDhb173.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/p[3]/Sentence[1]/Token[34])
DEAmsEDCSHDh316.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/lg[1]/Sentence[1]/l[3]/Token[4])
DEAmsEDCSHDa80-7.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/lg[2]/Sentence[1]/l[1]/Token[3])
DEAmsEDCSHDhb91.1.xml
(/TEI.2[1]/text[1]/body[1]/div0[1]/p[1]/Sentence[1]/Token[25])
==========

2. Steve's most recent Tamarind example, with explanations:

From: sramsay@uga.edu <sramsay@uga.edu>
To: webviz listserv <webviz@lists.prairienet.org>
Date: Jun 8, 2005 9:04 PM
Subject: Re: [Webviz] Qs on processing XML documents for nora

After Tamarind is done, you get a database with a single table containing (perhaps) in excess of a million rows. Each row contains the following:

1. a token (all lowercase)
2. a token type (word, punctuation, number, symbol, etc.)
3. a part of speech code (one of 40 from the Penn Treebank)
4. an orthographic label (lowercase, upperInitial, mixedCaps, etc.)
5. a string length (i.e. the number of characters in the token)
6. a unique XPath expression indicating where the token is in the document
7. a filename

So, for example, the row might be:

king | word | NN | lowercase | 4 |
/PLAY[1]/ACT[1]/SCENE[2]/SPEECH[33]/LINE[2]/Token[5] | j_caesar.xml

Notice the XPath. It's what I call a "verbose XPath" -- not the usual thing you see as part of a search query or in XSLT, but a (syntactically legal) string that identifies exactly where the token occurs in j_caesar.xml. In this case, it's saying that this particular instance of "king" is the fifth token of the second line of the thirty-third speech in act 1, scene ii (all of which is

enclosed in the root "PLAY" tag). It also knows that "king" is a noun (NN), that it is all lowercase in the original document, that it's a word, and that it has four characters.

That's it! That's the datastore. "Dumb as bricks," as Mark Olsen would say. ;)

But notice that that simple table contains the seeds of a vast amount of quantitative information of the sort that you need in order to do data mining, machine learning, and text analysis.

Right off the bat, we can use simple SQL queries to answer the following questions:

1. How many unique word tokens are there in Julius Caesar?
2. How many transitive verbs are there in Shakespeare?
3. How many scenes are there in act 4 of Macbeth?
4. What is the average string length for a character name?
5. How often does the word "night" occur in Shakespeare?
6. How may words are common to both Romeo and Juliet and Othello?
7. How many documents are there in the corpus?
8. How many <l> tags are there in King Lear?

These are very low-level queries, but we can use them to construct much more complicated vectors. For example, if you can compute shared vocabulary, then why not create another table that contains term weights for every token? If you already know all the parts of speech, then why not build a table that lists the proportions among all the different parts of speech within the various documents? If you have all the word frequencies, then what about matrices full of frequency vectors (for various features)?

Tamarind also inserts some tags into the XPath, so it knows about persons, date ranges, place names, and that sort of thing -- which means that you can also tabulate data about things that weren't necessarily tagged in the original document.

3. Bei's Naïve Bayesian analysis of the Dickinson corpus:

From: Bei Yu <beiyu@uiuc.edu>
To: webviz listserv <webviz@lists.prairienet.org>
Date: Apr 20, 2005 4:00 PM
Subject: [Webviz] feature ranking for Dickinson naive bayes classification

Hi,

I attached the feature ranking results for Naive Bayes classification. The third column is the probability ratio, which is a measure of the feature's discriminative power in NB classification. the positive ratio numbers mean the corresponding tokens appears more in "hot" poems, the larger the number, the higher the discriminative power the token has, and vice versa for the negative numbers.

the features I use for now is the original words appeared in the body of the poem, no stemming or upper-lower case change. The results show that NB classification does not 100% agree with our human experts.

Lists of "hot" words they agreed:

tasted
faces, face
touching, touches, touch
Lords, lord
Berries
feel
Nights
hand, Hands
Nut
Butterfly
seal
Queen
Bees

Lists of words that the NB algorithm doesn't think as "hot":

Music
warm
tune
Bee, bee
night, Night
Lightning, lighting
blood
loves
Arms
sun, Sun
nuts
berry
Face
cut
Love

Itself, itself
cold
Hand

Some words in the Maryland vocabulary never appeared in Dickinson's poem actually, like "electricity". I searched the original data set and it's true!

BTW, the NB training error for dickinson set is 99%, only 3 out of 269 poems are wrongly classified.

CLASS_PROB_Hot	CLASS_PROB_Not	RATIO	FEATURE
double	double	double	String
7.140671223094971E-4	7.871536523929471E-5	2.2051385922805053	mine

[Only one line shown as an example]