CS341 Journal

Jo, Dilli and Howon

April 9, 2014

Jo: 4-2-2014

This is a lil journal system I like using. Basically, you put a new file whose name is the date (this one is April 2nd, that is, `4-2.tex`) with your journal entry. Then a composite pdf is generated from the python script `GenerateJournal.py`. Its simple and offline, so we still have a collaborative thing but without the annoyances of sharing a google doc. The pdf file could be generated during the pre-commit hook, if you like that sort of thing.

Howon: 4-6-2014

More resources, I found. More thoughts about the data, I had.

https://dev.twitter.com/docs/platform-objects is probably most important for not having to go through the json fields themselves. We will have the problem of nulled fields, empty sets, and this field guide (get it? birds?) has the defaults.

https://github.com/kevinweil/elephant-bird/ is a good resource for dealing with json in hadoop

The dataset is about 400 gb. It seems like a 1% sample would be good, as it allows us to fit it on main memory even on a crappy box like mine. This I approximated a little bit by taking a 1-file sample, which is actually just about a gigabyte: ideally, we would use something like TABLESAMPLE in Hive or SAMPLE in Pig Latin, because then it would be a good pseudorandom sample.

One troubling thing about taking from the *firehose*, at least at first glance, is that the tweets are taken at the moment of conception. So it's much more annoying to get the favorites data on the tweets.

Histograms about the facts about the users I found are also forthcoming.

Howon: 4-8-2014

Todos

Howon

1. Email the guy about getting the graph data, so we can actually sample the users and we need that graph for other stuff

2. Maybe do an actual sample to think about distributions myself to unbias the tweets (maybe we're ok with just getting people who tweet more?)

Dilli

1. Keywords for one file
2. Think about reconstructing the graph

Jo
Work on implementation of the TwitterRank and the topic distributions
Brainstorming.
Some ideas may be worse than others.

1. We could have more users
2. We could construe each tweet as a separate document
3. Think of a 3-layer bayes net, where user has distribution over topics and the tweet has distribution over topics where prior is over user's distribution over topics.
4. Find a user that does poorly in twitterrank even though they do well in real life or in our algo (someone strong in 2 different fields)
5. What if we pick our own topics, and do actual labelling?
6. Syntax? Stop words? Use NLTK
7. Can we reconstruct the retweeting graph and do everything in the proposal first?

Jo: 4-9-2014

Topic Distillation

The module `Topic.py` has a function `topic_model` which takes a number of topics and a list of word count histograms for some documents and performs LDA on it to assign each document to a topic and to topic a distribution of words. The TwitterRank paper recommends running this where each document is all words of all a user's tweets. We can try this with users picked in various ways and words chosen in various ways. Does reasonably on the generative model in `TopicTest.py`.

Everything runs in-memory and not distributed. We should be able to fix this so that the histograms actually live on disk and are streamed when necessary, but I don't know if we'll need to yet. It depends on the size of the Twittersphere we wish to rank and the number of words considered.

This analysis is required to compute the edge weights for the GraphRank algorithm.

GraphRank

Given as input is a JSON graph where each node has its weighted outgoing edges and teleportation probability. Outputs the weighted graph rank. The TwitterRank paper recommends running this once per topic where the edges are weighted by topic similarity and number of tweets and the teleportation probabilities are the topic ratings for each user. We should try to use the explosive retweetiness for edge probabilities.

TODO

Link these together (even on a fake graph), and make this happen on a fraction of our data. Probably will require a map reduce run to get histograms.