* The Whale

Visualizing Twitter with local groups

Erik Turnquist, Rob Hanlon, Eric Kimbrel, Zack Allred

Overview

All features we could possibly think of:
- Visualization of Twitter relationships
- Friend and follower similarity
- Determine trends among a group of users
- Follower recommendations
- Identify groups of similar users

Core functionality achieved:
- visualization
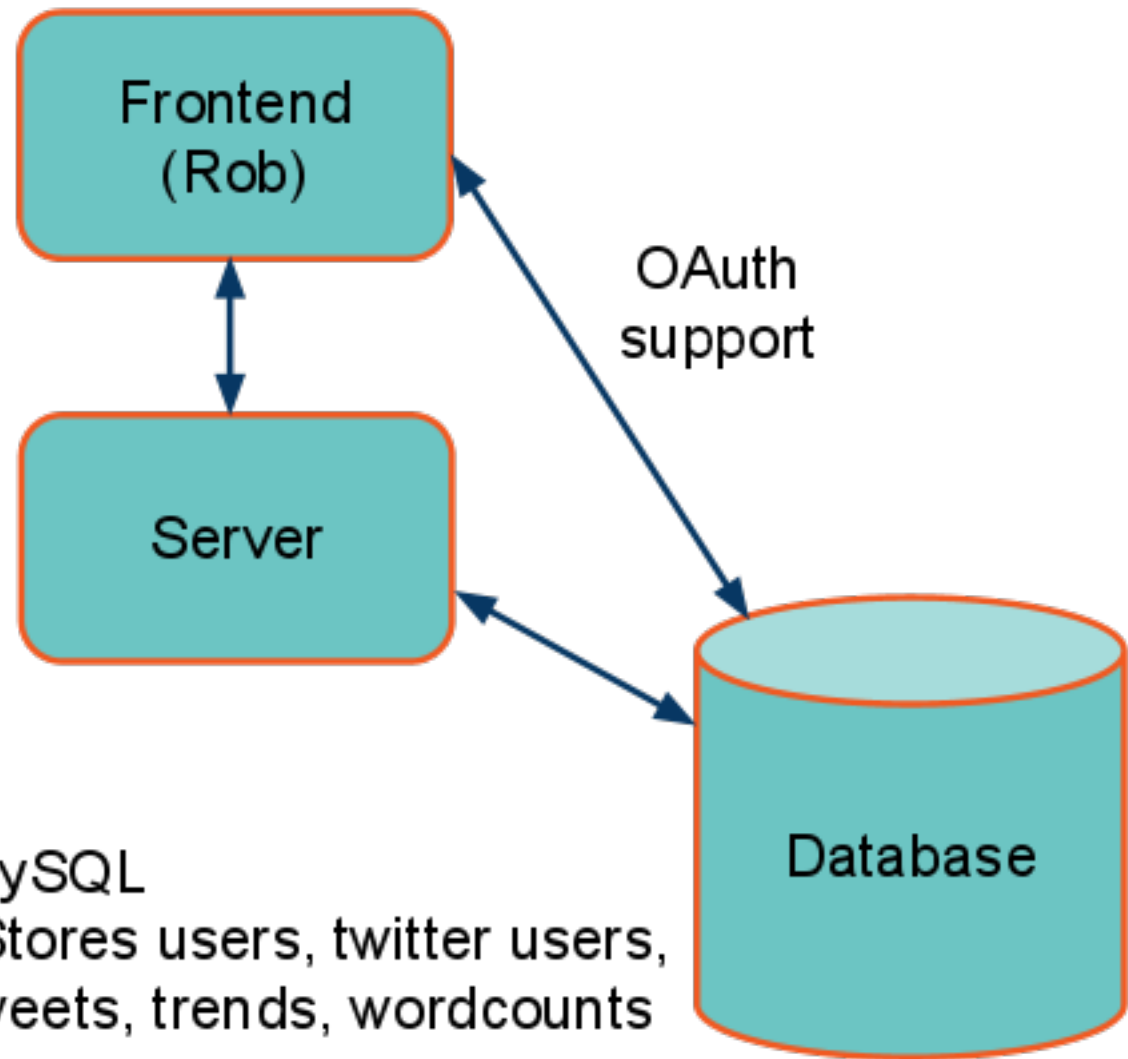- similarity
- trends

Architecture

Ruby
- jruby for requesting data
- OAuth for login and authorization
- <canvas> for UI

Java
- Auto-update twitter data (Erik)
- twitter4j
- Similarity ranking (Zack)
- Group trends (Eric)

Frontend
(Rob)

OAuth
support

Server

MySQL
-Stores users, twitter users,
tweets, trends, wordcounts

Database

Gathering Data

- Background service for automatically updating data
  - Lists of friends and followers
  - Newest tweets
- Optimized for performance
  - Threadpools for API calls
  - Only receives new information from API
- Uses minimal API calls
- Twitter API Issues

Similarity

- Treat each user as a concatenation of their tweets.
- We tried two different algorithms to rate the similarity scores of the result.

- First Attempt:
  - Euclidean distance of the tf vectors (as proportions).
    - Ignored words whose frequency were too large or small
    - Ignored words used by only one of the two Twitterers
  - Similarity = 1 - the Euclidean distance

- Second Attempt:
  - Cosine similarity of the tf-idf vectors

We chose to use the cosine similarity of the tf-idf vectors since the first method did not take into account the document frequency of the terms.

Finding Trends on Selected Users.

1. Word Usage: Base trends off the most common words that occur between all selected users.

   Group of 3 users:
    hcr, house, healthcare, politics, base64, facebook, red, twitter, glenn, songoftheday

   Group of 254:
   db09, splatter, fb, hc09, sandvoxtip, unix@40, gomachine, dinner, unixtour, l4d

2. Term Extraction: Base trends on topics pulled from text using Yahoo's term extractor

   Group of 3:
   iphone, google, bandstand, sounders, storytelling, curio, dingell, yesss, bystander, skerik

   Group of 254:
   iphone, google, twitter, rt, tweets, tweet, tinyurl, ly, facebook, followers

Finding trends by word frequency vs. topic extraction.

Process Time Vs Doc Length
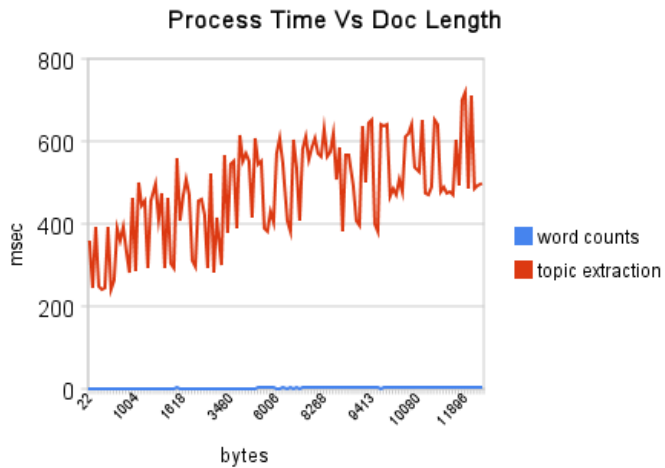
Phase 1: Consider all of a users tweets as a single document and translate the document into word counts or a list of topics.

Time to return trends

Rows Stored in Database
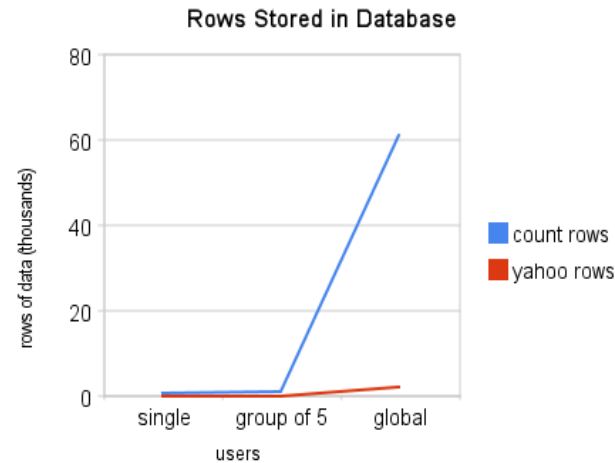
Phase 2: When trends are requested query our database and return the most frequent terms from the selected users.

Frontend structure

Non-GUI components:
- JRuby on Rails
  - Why? Direct construction and reference to Java objects from Ruby code, so Rails can do what it does best, and Java can do the number crunching
- twitter-auth RubyGem
  - 2 minute Twitter OAuth integration
- Resque
  - Used to asynchronously hit the backend and check for updated data
- Distributed Ruby
  - Conveniently shuttles around object references from Rails to Resque in order to make sure we're all working with the same objects

```
14 Dec 09:03:08 . DB 0: 9 keys (0 volatile) in 16 slots HT.
14 Dec 09:03:08 . 1 clients connected (0 slaves), 420713 bytes in use, 0 shared objects
14 Dec 09:03:13 . DB 0: 9 keys (0 volatile) in 16 slots HT.
14 Dec 09:03:13 . 1 clients connected (0 slaves), 420713 bytes in use, 0 shared objects
14 Dec 09:03:18 . DB 0: 9 keys (0 volatile) in 16 slots HT.
14 Dec 09:03:18 . 1 clients connected (0 slaves), 420713 bytes in use, 0 shared objects
14 Dec 09:03:23 . DB 0: 9 keys (0 volatile) in 16 slots HT.
14 Dec 09:03:23 . 1 clients connected (0 slaves), 420713 bytes in use, 0 shared objects
14 Dec 09:03:25 - 10 changes in 300 seconds. Saving...
14 Dec 09:03:25 - Background saving started by pid 29420
```

GUI components:
- <canvas>
  - HTML5
  - Open technology, new standard
  - Flash sucks, but canvas isn't really ready for primetime. Learned that the hard way
- Processing.js
  - Ported from Java by John Resig (of jQuery fame), a really intuitive framework for building animations and graphics on top of canvas
- traer.physics.js
  - Also ported from Java, but by me. :) Great for building force-directed layouts

```
23              :sim => db.get_similarity_score(cur_id, id),
24              :follow => !follow[id].nil?,
25              :friend => !friends[id].nil?
26
27      render :json => hash
28      set.remove(id)
29    end
30
31    def get_center
32      render :json => { :login => @current_user.login,
33                        :id => @current_user.twitter_id.to_i,
All|~/repo/frontend/app/controllers/graph_controller.rb [Rails-controller]
53    that.mouseDragged = function () {
54    };
55
56    that.mousePressed = function () {
57      for (var i = 0; i < proc.circles.length; i++) {
58        var position = proc.circles[i].scaledPosition(),
59            distanceX = position.x() - proc.mouseX,
60            distanceY = position.y() - proc.mouseY;
61
62        if (proc.sqrt(proc.sq(distanceX) + proc.sq(distanceY)) < (proc.NODE_SIZ
63          proc.circles[i].toggleHighlight();
```

Some eye candy


```
23358945        0 new tweets       71 db tweets
  SQL (0.0ms)    SET SQL_AUTO_IS_NULL=0


Processing GraphController#get_center (for 32.158.102.13 at 2009-12-14 08:57:34) [GET]
  User Load (1.0ms)    SELECT * FROM `users` WHERE (`users`.`id` = 3) LIMIT 1
Completed in 96ms (View: 9, DB: 1) | 200 OK [http://attu1.cs.washington.edu/graph/get_center]
3647681 0 new tweets       9 db tweets
69591866        0 new tweets     100 db tweets
4794501 0 new tweets       100 db tweets
```

Trending topics:

- chunk from the goonies
- dynamic memory allocation
- pointer arithmetic
- new years resolution
- c pointer
- essay version
- magnificent idea
- dad work
- i don t care
- bayes theorem
- cool place
- beta release
- tweeters
- arianna
- oooh
- zombies
- academics
- lawyer
- innovation
- trees

Trending topics:

- iphone
- tweet
- aya
- follower
- account settings
- notifications
- sync
- timeline
- rt
- api
- fm support