Little Bird: The Twitter Rating Engine

Gabriel Levine, Maria Smith, & Graham Northrup

Goal

- Create an engine that will collect tweets about a query
- Analyze the content of each tweet to determine a “rating” associated with that tweet
- Aggregate these ratings into a single score returned to the user

- Sample input: “Star Wars 7: The Force Awakens” and “Movie”
  - Sample output: “Twitter users rated this movie ?/100” (? is what we are finding)

Level 1: Constrained Search

Input Prompt: Search for a movie currently playing.

Input: Iron Man

Output: A Little Bird Says, “Iron Man has a rating of 80%”

Level 2: Discerning, Less-Constrained Search

Input Prompt: Search for a movie, restaurant, or weather location.

Input: [Iron Man] or [Noodles] or [Chicago]

Output: A Little Bird Says, “<Input> Man has a rating of 80%”

Requires: Additional data streams. Ability to distinguish domain of query.

Level 3: Natural Language, Open Search

Input Prompt: How good is_____?

Input: Kanye West

Output: A Little Bird Says, “Kanye West has a rating of 80%”

Requires: Generalization to sentiment prediction for any query.

Data Sources

1. Twitter API
    a. We will need to work with the twitter API in order to do a project about tweets
    b. Instagram API
        i. Perhaps if we decide to work on food specifically we can use Instagram as an alternative data stream
2. APIs used to check the effectiveness of Little Bird
    a. Restaurants
        i. Yelp API
    b. Weather
        i. Open Weather or Forecast APIs
    c. Movies
        i. Rotten Tomatoes API

New Technologies

- Django Interface
  - Takes in the query and sends back the Little Bird rating
- Tweet to Rating algorithm
  - Will analyze the text to determine a positive or negative sentiment for each text
  - Likely use Natural Language Toolkit (NLTK)
- Some sort of data structure to keep this all together
  - Perhaps some sort of class that has multiple parts to keep track of tweets and ratings
- Testing algorithm to compare our results with Yelp, Forecast, etc.

Timeline

1. Checkpoint 1 (Week 6): Be able to collect all tweets relating to some query and collect them in a data structure locally
    a. Maria will lead this section
2. Checkpoint 2 (Week 8): Have a polished algorithm that can convert a tweet to a rating for at least one type of query (movie, restaurant, etc.)
    a. Gabe will lead this section
3. Final Checkpoint (Week 10): Put together the Django interface and be able to send and receive queries.
    a. Also during this time we will attempt to generalize the engine to accept as many types of queries as we can
    b. Graham will lead this section