Eftychis: Measuring Happiness on the Twitter Platform

BU Department of Computer Science

Anton M. Paquin, William J. Chen
{paquin,chenwill}@bu.edu

Boston University College of Engineering
Department of Electrical & Computer Engineering

Powered by:

Google Cloud Platform

elasticsearch

kibana

Data Acquisition

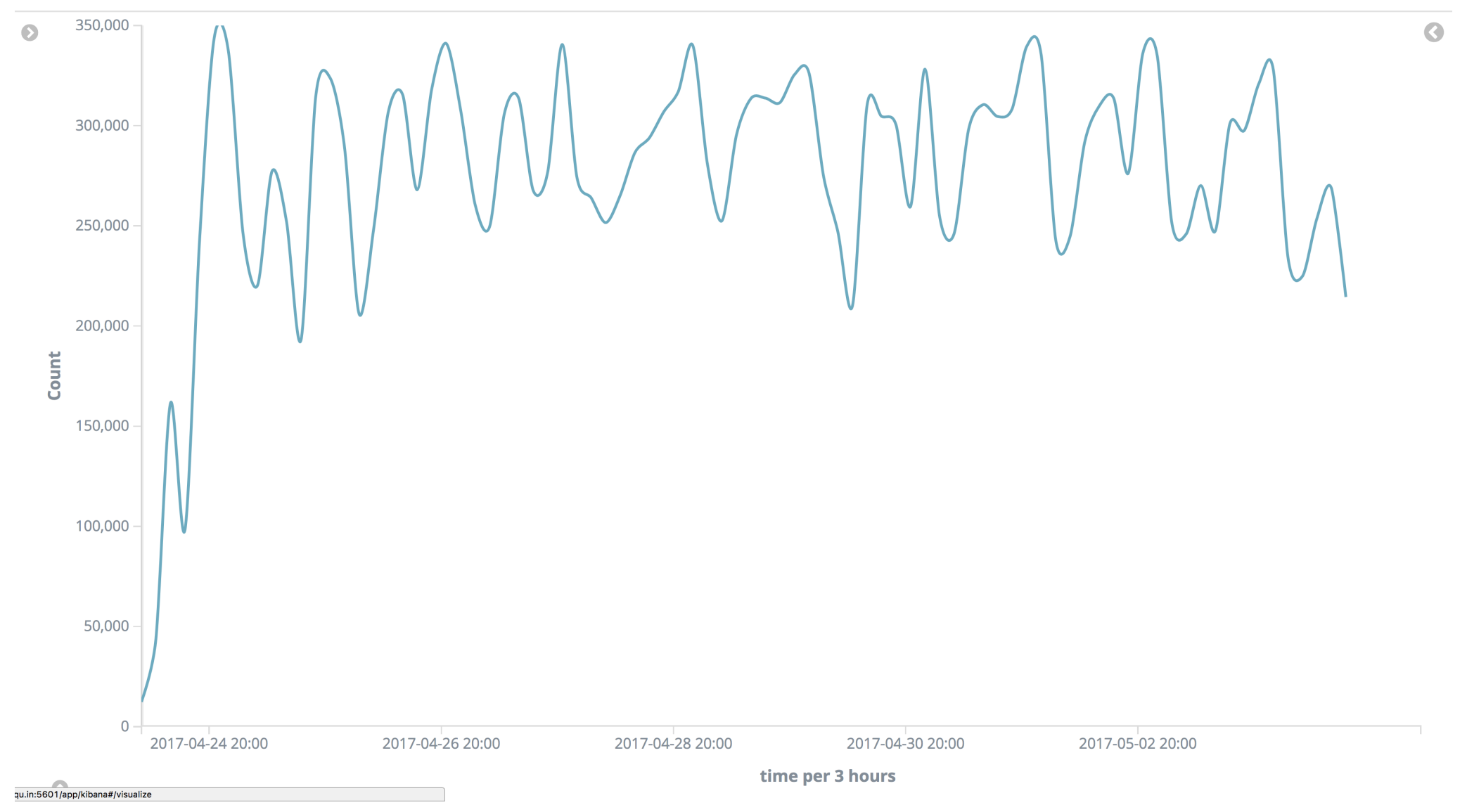

Figure 1: Scraper Acquisition Rate, bin size = 3 hours

Using a Google Cloud Platform virtual server and the Twitter API, we were able to scrape **23,585,039** tweets over the span of **10 days** from the Twitter social network. To obtain a geographically diverse set of tweets, we first list of **1054 coordinate pairs** that were equally spaced (**65km** apart) across the continental US in a grid. We then made hourly API calls that requested 100 tweets located within 65km of each coordinate pair. The collected data was then ingested into the Elasticsearch engine, which performed word tokenization and snowball filtering (word stemming).

Glossary

Clamp - Ground truth

Data Analysis

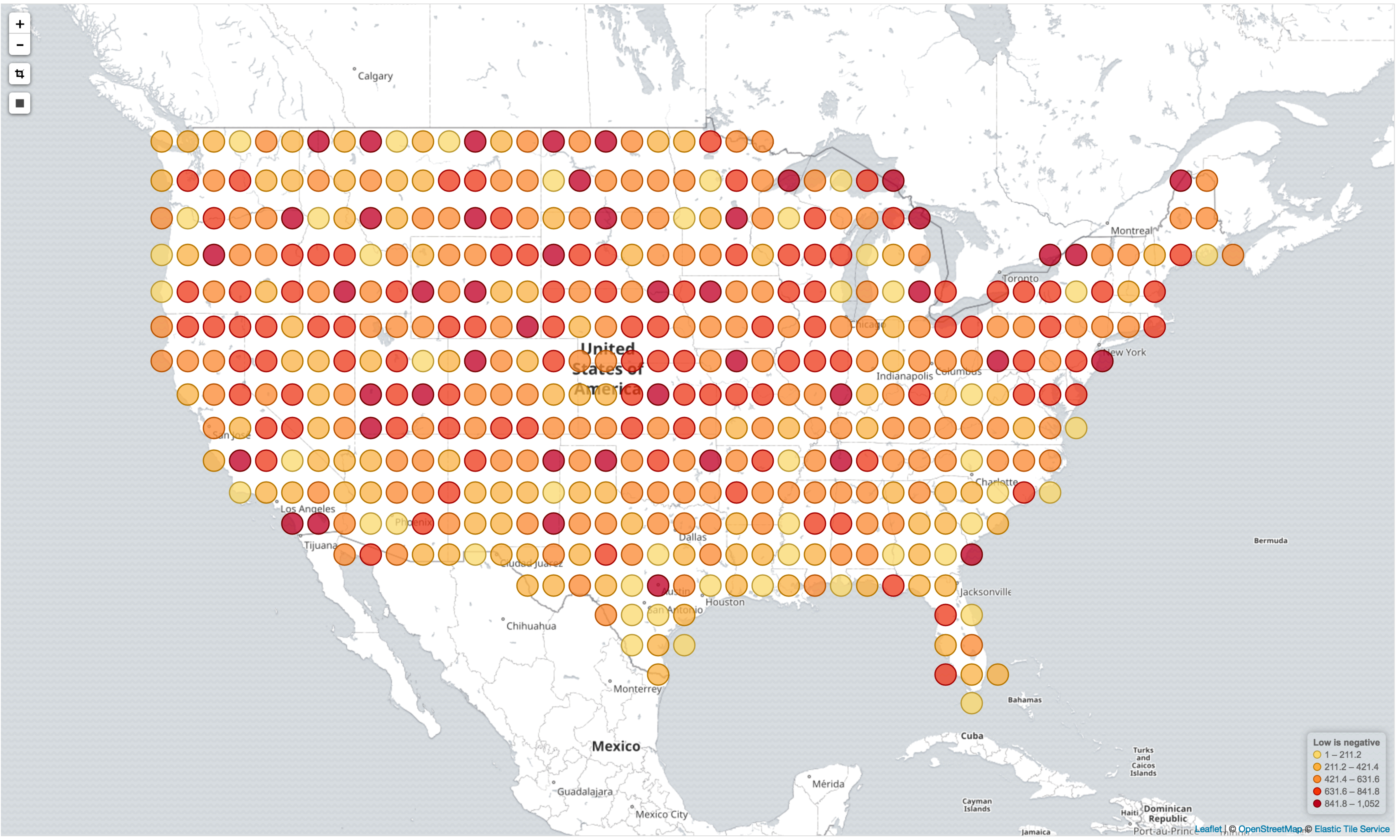

Figure 2: Happiness ranking amongst location nodes. Darker red = more happy

Label Propagation

To begin our sentiment analysis, we constructed an undirected graph with nodes representing each of the top 2000 words. Edges were created between each node with an edge weight indicating the frequency of tweets containing words A and B divided by the sum of tweets containing A and every other node. The result is a 2000-by-2000 matrix indicating probability that word i and j are related. Mathematically speaking, we calculate

$$T_{i,j} = \frac{w_{i,j}}{\sum_{k=0,i!=k}^{2000} w_{i,k}} \quad (1)$$

for all nodes i,j.

From the 2000 most popular words, we created a list of "clamp words" containing positive words, and another for negative words. We initialize a 2000-by-2 matrix y with each element set to (0,0), set positive clamps to (1,0), and set negative clamps to (0,1). Using y, T, and a constant α (= 0.3), we produce a 2000-by-2 matrix y' that relates non-clamp words with their clamped words via the equation:

$$\mathbf{y'} = \alpha\mathbf{T}\mathbf{y} + (1-\alpha)\mathbf{y} \quad (2)$$

With the y' matrix, we are able to compute a "raw happiness score" for all of the scraped tweets.

Linear Regression

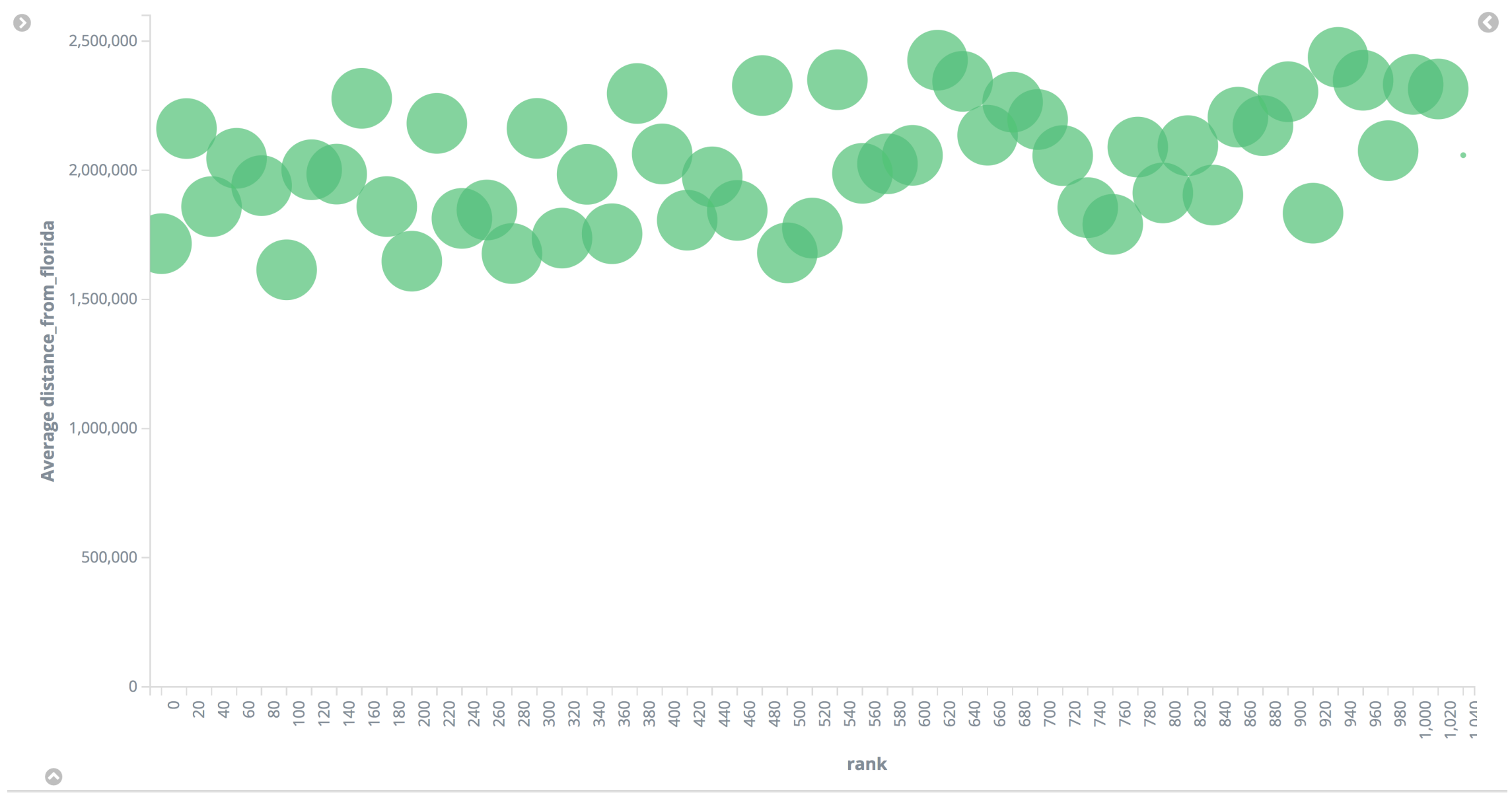

Figure 3: Happiness Rank and Distance from Florida

slope: 4.20157469466e-05
intercept: 440.693646678
r value: 0.114768473686
p value: 0.000189854300511
std err: 1.12178311817e-05

Timeseries

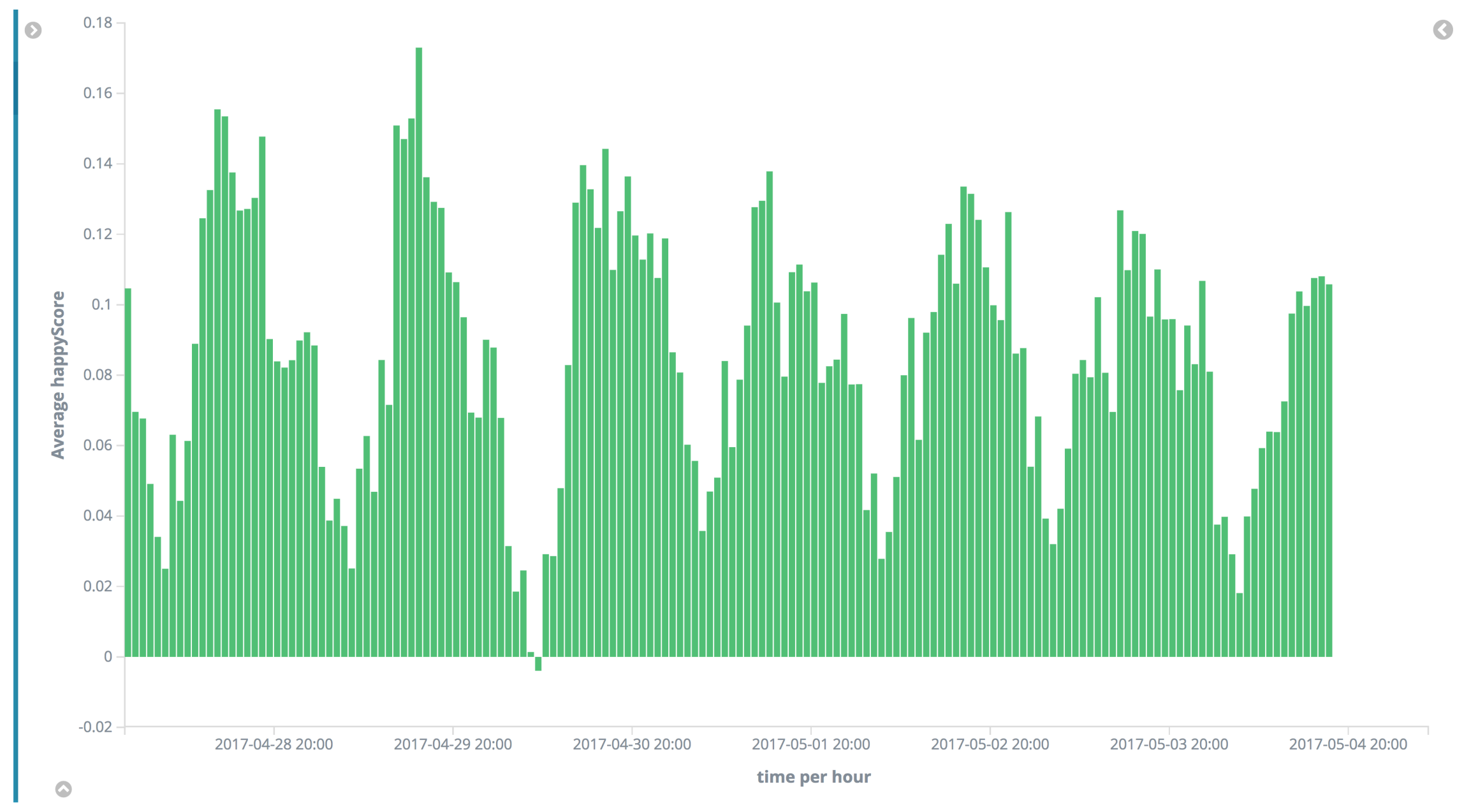

Figure 4: Average Happiness Score over 7 days, bin = 1 hour

Selected Words

korea	13828	-0.134306817929182‌84
trump	145406	0.0343466446631247‌6
putin	4894	0.00280809598530726‌03
emoji	4771	0.376287210435987
data	11407	0.212887906122215‌17
compsci	11	0.596370179544795‌7
java	625	-0.153782725250720‌98
twitter	67944	0.134914129256658‌35
anton	257	0.173716786553075‌8
will	6518	0.0578141135010575‌7
chemic	2077	-0.00417094566450365‌9
fuck	191900	-0.131366611629005‌2
mario	4620	0.288520502598543‌7
convers	14512	0.305865257178305‌1

Figure 5: Words, Frequency, Happiness Score

Happiness Score	Word
0.76879883043755948	'dog'
0.76755679586472614	'weather'
0.76078575172677931	'sure'
0.76034724795438691	'presid'
0.7483814115522347	'dope'
0.74837395464244649	'feel'
0.74769187912219326	'look'
0.74724184565477569	'smell'
....	...
0.60439786628291425	'n***a'
0.60771799519059977	'price'
0.6100769239001258	'death'
0.61080646060234023	'realdonaldtrump'

Figure 6: Selected Happiest/Unhappiest Words

References

Yen–Jen Tai, Hung–Yu Kao, Automatic Domain—Specific Sentiment Lexicon Generation with Label Propagation, Proceedings of International Conference on Information Integration and Web–based Applications & Services, December 02–04, 2013, Vienna, Austria