Personality factors and normalized (web) distance

CHRIS JENKINS
PH.D, PSYCHOLOGY, U. NEW MEXICO, 2014
JENKINSC.COM - GITHUB.COM/COEJ

Early personality research

Allport & Odbert, 1936:

4,500 English adjectives describing human personality

- Reduced to 16 independent meanings

Reserved or Warm	Trusting or Vigilant
high / low reasoning ability	Abstracted or Practical
Emotionally Stable or Reactive	Private or Forthright
Deferential or Dominant	Self-Assured or Apprehensive
Serious or Lively	Traditional or Open-to-Change
Expedient or Rule-Conscious	Self-Reliant or Group-Oriented
Shy or Bold	Perfectionistic or Tolerates-Disorder
Sensitive or Unsentimental	Relaxed or Tense

Big Five personality factors

- Dimensions of stable independent variation between individuals
- E.g.: agreeableness:
  - Trusting, forgiving
  - Undemanding
  - Altruistic, warm
  - Compliant, not stubborn
  - Modest, not showy
  - Tender-minded, sympathetic
- (Openness/IQ correlated)

Open to experience / closed
Conscientious / undisciplined
Extraverted / introverted
Agreeable / antagonistic
Neuroticism / emotional stability
(+ General intelligence)

Five personality factors

- Highly stable over time, across raters
- About 50% genetically heritable within populations
- Cross-culturally consistent structure and sex differences
- Women higher in agreeableness, conscientiousness, and neuroticism (self-rated, other-rated)

Open to experience / closed
Conscientious / undisciplined
Extraverted / introverted
Agreeable / antagonistic
Neuroticism / emotional stability
(+ General intelligence)

Trait desirability

Self-esteem correlations (self-ratings, $N = 326000$)

Openness to experience:	$r = 0.17$
Conscientiousness:	$r = 0.24$
Extraversion:	$r = 0.38$
Agreeableness:	$r = 0.13$
Neuroticism:	$\boldsymbol{r = -0.50}$

Social communication and personality

About 65% of all human conversations are social gossip (Dunbar, 1996)

Gossip shares information about others’ reputations and behaviors; e.g., warnings about dishonesty or freeriding among group members

Semantic distances and personality

Personality of writers can be predicted from word use (Neuman & Yochai, 2014)

Normalized (web) distance

$$\mathrm{NGD}(w_1, w_2) = \frac{\max(\log D(w_1), \log D(w_2)) - \log D(w_1 w_2)}{\log N - \min(\log D(w_1), \log D(w_2))}$$

Used as a quick and cheap measure of semantic relatedness

Broad Hypothesis

Individual language use on the web is influenced by mental models of a 5-factor structure:

- Within writing by the same person (about self, friends, influencers, product signaling affordances)
- Within references to the same object (across writers) - descriptions of people and behavior
- (How well would a full web index reflect this?)

Predictions, complexities

Terms related to the same personality factor should co-occur in writing about people or oneself (on the web)

To clarify and implement going forward:

- What pages contain text that mostly refers to one subject? (Tweets?)
- How can we differentiate positive and negative uses of term?
- Using search page counts: limits on search string complexity, API calls
- Crawling pages or filtering streams oneself: Twitter stream?

Simple trial: Bing

Terms strongly related to each factor

O: curious, creative

C: organized, dependable

E: assertive, outgoing

A: cooperative, helpful

N: anxious, moody

45 pair comparisons, 10 individual comparisons

Retrieving page counts with Python and the Google/Bing Custom Search APIs

Google:

- Coax Google to create a full-web custom search engine:
- Get engine ID, create API key (entered in URL string, GET, no auth.)

Bing:

- Uses HTTP basic authentication (Requests)

(Links/code to be added)

```python
def bing_count(query):
    import requests

    r = requests.get("https://api.datamarket.azure.com/Data.ashx/Bing/S
                     "?Sources=%27web%27&Query="
                     "%27" + query + "%27"
                     "&$top=1&$format=JSON",
                     auth=('user', 'jy8lW+jNxEbubtNf(....)'))

    response = r.json()
    print response
    count = response['d']['results'][0]['WebTotal']
    return int(count)
```

```python

items = ['curious', 'creative', 'organized', 'dependable', 'assertive',
         'outgoing', 'cooperative', 'helpful', 'anxious', 'moody']

from itertools import combinations
pairs = list(combinations(items, 2)) # all pairs (unordered) in lexicographic order

pair_pagecounts = {}
for p in pairs:
    query = "%s %s" % p
    pair_pagecounts[p] = bing_count(query)
    print query, pair_pagecounts[p]

single_pagecounts = {i: bing_count(i) for i in items}

distances = {pair: NGD(pair, pair_pagecounts, single_pagecounts)
             for pair in pair_pagecounts}

by_dist = sorted([(v, k) for (k, v) in distances.items()])
by_dist_rounded = [(round(a, 3), b) for (a, b) in by_dist]
```

```
single_pagecounts
{'anxious': 5900000,
 'assertive': 3090000,
 'cooperative': 7820000,
 'creative': 49300000,
 'curious': 13100000,
 'dependable': 3240000,
 'helpful': 32500000,
 'moody': 9780000,
 'organized': 18000000,
 'outgoing': 3380000}
```

```
pair_pagecounts
{('anxious', 'moody'): 1100000,
 ('assertive', 'anxious'): 1400000,
 ('assertive', 'cooperative'): 196000,
 ('assertive', 'helpful'): 3390000,
 ('assertive', 'moody'): 89400,
 ('assertive', 'outgoing'): 204000,
 ('cooperative', 'anxious'): 1420000,
 ('cooperative', 'helpful'): 16600000,
 ('cooperative', 'moody'): 403000,
 ('creative', 'anxious'): 11800000,
 ('creative', 'assertive'): 1930000,
 ('creative', 'cooperative'): 10700000,
 ('creative', 'dependable'): 61900000,
 ('creative', 'helpful'): 58400000,
 ('creative', 'moody'): 3420000,
 ('creative', 'organized'): 89900000,
 ('creative', 'outgoing'): 2930000,
```

```python
items = ['curious', 'creative', 'organized', 'dependable', 'assertive',
         'outgoing', 'cooperative', 'helpful', 'anxious', 'moody']

from itertools import combinations
pairs = list(combinations(items, 2)) # all pairs (unordered) in lexicographic order

pair_pagecounts = {}
for p in pairs:
    query = "%s %s" % p
    pair_pagecounts[p] = bing_count(query)
    print query, pair_pagecounts[p]

single_pagecounts = {i: bing_count(i) for i in items}

distances = {pair: NGD(pair, pair_pagecounts, single_pagecounts)
                for pair in pair_pagecounts}

by_dist = sorted([(v, k) for (k, v) in distances.items()])
by_dist_rounded = [(round(a, 3), b) for (a, b) in by_dist]
```

```python
def NGD(word_pair, pair_pagecounts, single_pagecounts):
    from math import log
    def log2(x): return log(x, 2)

    word1, word2 = word_pair
    #lexsort_pair = tuple(sorted([word1, word2]))

    D_w1 = single_pagecounts[word1]
    D_w2 = single_pagecounts[word2]

    try:
        D_w1w2 = pair_pagecounts[word_pair]
    except KeyError:
        D_w1w2 = tuple(reversed(pair_pagecounts[word_pair]))

    N_est = 263000000 # bing_count("the") =  263000000

    numerator = max([log2(D_w1), log2(D_w2)]) - log2(D_w1w2)
    denominator = log2(N_est) - min([log2(D_w1), log2(D_w2)])
    distance = numerator / denominator
    return distance
```

```
[(-0.224, ('creative', 'organized')),
 (-0.141, ('organized', 'helpful')),
 (-0.081, ('creative', 'helpful')),
 (-0.074, ('dependable', 'helpful')),
 (-0.052, ('creative', 'dependable')),
 (0.091, ('curious', 'helpful')),
 (0.093, ('dependable', 'outgoing')),
 (0.099, ('helpful', 'anxious')),
 (0.183, ('dependable', 'anxious')),
 (0.191, ('cooperative', 'helpful')),
 (0.243, ('helpful', 'moody')),
 (0.244, ('organized', 'dependable')),
 (0.259, ('dependable', 'cooperative')),
 (0.324, ('assertive', 'anxious')),
 (0.333, ('curious', 'dependable')),
 (0.352, ('organized', 'cooperative')),
```

```
(0.377, ('creative', 'anxious')),
(0.417, ('dependable', 'assertive')),
(0.428, ('curious', 'anxious')),
(0.435, ('creative', 'cooperative')),
(0.449, ('cooperative', 'anxious')),
(0.463, ('dependable', 'moody')),
(0.465, ('outgoing', 'anxious')),
(0.487, ('organized', 'anxious')),
(0.498, ('organized', 'outgoing')),
(0.508, ('outgoing', 'helpful')),
(0.509, ('assertive', 'helpful')),
(0.574, ('curious', 'organized')),
(0.575, ('anxious', 'moody')),
(0.584, ('curious', 'outgoing')),
```

```
(0.63, ('curious', 'cooperative')),
(0.632, ('assertive', 'outgoing')),
(0.648, ('creative', 'outgoing')),
(0.68, ('outgoing', 'cooperative')),
(0.718, ('organized', 'assertive')),
(0.729, ('creative', 'assertive')),
(0.73, ('organized', 'moody')),
(0.785, ('curious', 'moody')),
(0.811, ('creative', 'moody')),
(0.827, ('curious', 'creative')),
(0.83, ('assertive', 'cooperative')),
(0.875, ('curious', 'assertive')),
(0.904, ('outgoing', 'moody')),
(0.907, ('cooperative', 'moody')),
(1.056, ('assertive', 'moody'))]
```

Next steps?

Selecting words

- Larger, unbiased collection of personality terms

Filtering results

- Social media
- Working with negative and positive references to term
- Selecting pages where terms will apply to the same subject (tweets?)

Analyses

- Some form of factor analysis applicable to this matrix of weights?
- Network modeling (weighted graph) in NetworkX?