a document is an ordered set of words that (at least in part) expresses the cognitive and affective states of the author

we want a automatized method that transfers psychological scales to documents and maintain validity and reliablility

preferably, the method should be scalable both in terms of quantity and context

we can use a dictionary to extract cognitive and affective keywords from a collection of documents and apply a sentiment function

```
'Did Crooked Hillary help disgusting (check out sex tape and past) Alicia M become a U.S. citizen
so she could use her in the debate?'

Positive sex, citizen
Negative crooked, hillary, disgusting, out
Sentiment Score (2+1) + (-2-1-3-1) = -4
Sentiment Polarity Negative
Overall Score Sum of all sentence scores
```

a sentiment vector is simply a vector of keyword frequencies weighted by sentiment scores

sentiment analysis a set of methods for extracting the affective (primarily) components from unstructured data

Used in business analytics and bio-NLP to predict market behavior, consumer preferences, happiness and quality of life

Originate in psychological and sociological scale studies

Three general approaches:

- Dictionary-based methods (word counting)
- Supervised learning (machine learning)
- Unsupervised learning (machine learning)

A **dictionary** is basically a set of words with ratings

Ratings can be binary (± 1 or $0/1$) or based on continuum ($1, 2 \ldots m$ or $1 - m$)

Compute corpus frequency for each dictionary word and multiply their sentiment rating (weight)

Dictionary	# Fixed	# Stems	Total	Range	# Pos	# Neg	Construction	License
LabMT	10222	0	10222	1.3→ 8.5	7152	2977	Survey: MT, 50 ratings	CC.
ANEW	1030	0	1030	1.25 → 8.82	580	449	Survey: FSU Psych 101	Free for research.
WK	13915	0	13915	1.26 → 8.53	7761	5945	Survey: MT, >14 ratings	CC.
MPQA	5587	1605	7192	-1,0,1	2393	4342	Manual + ML	GNU GPL.
LIWC	722	644	1366	-1,0,1	406	500	Manual	Paid, commercial.
Liu	6782	0	6782	-1,1	2003	4779	Dictionary propagation	Free.
PANAS-X	60	0	60	-1,1	10	10	Manual	Copyrighted paper
Pattern 2.6	1528	0	1528	-1,0,+1	528	620	Unspecified	BSD
SentiWordNet 2.6	147701	0	147701	-1 → 1	17677	20410	Synset synonyms	CC BY-SA 3.0
AFINN	2477	0	2477	-5, -4, . . . , 4, 5	878	1598	Manual	ODbL v1.0
General Inquirer	4205	0	4205	-1,+1	1915	2290	Harvard-IV-4	Unspecified
WDAL	8743	0	8743	1 → 3	6517	1778	Survey: Columbia students	Unspecified
NRC	1220176	0	1220176	-5 → 5	575967	644209	PMI with emoticons	Free for research

across dictionaries we find inconsistencies, that is, words that seem incorrectly rated

$Negative_{MPQA}$: {*moonlight*, *cutest*, *finest*, *funniest*, *comedy*, *laugh*∗}
$Positive_{LIWC}$: {*dynamite*, *careful*, *richard*∗, *silly*, *gloria*, *securities*, *boldface*}

origin of dictionary mismatches

- reliance on specific sample of raters
- 'dirty' ratings

word ratings for dictionaries are based on more or less principled procedures

- survey-based: random samples or crowd sourcing
- manual: expert or naive (∼convenience)

rating issues

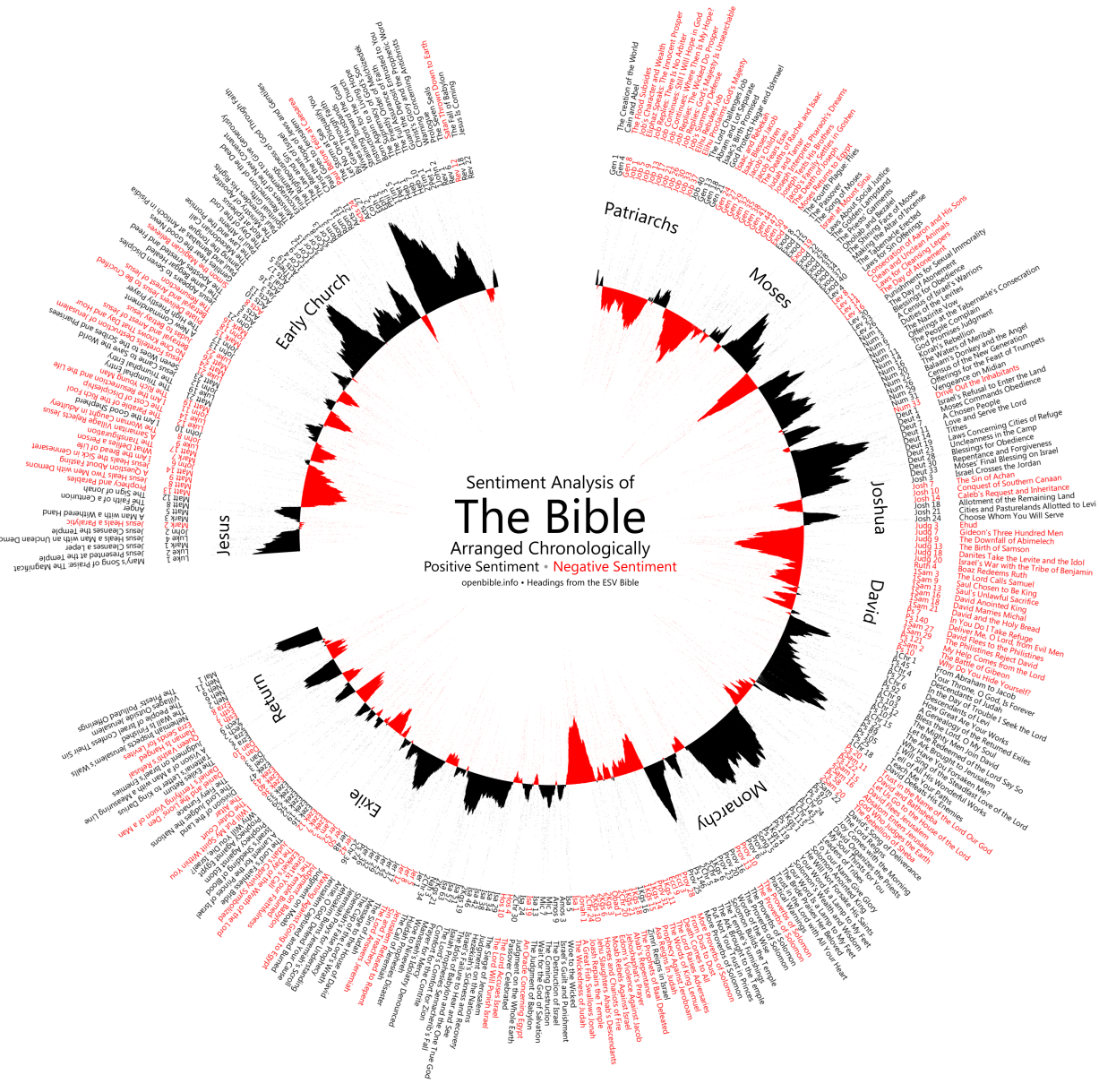
- space and time specificity (ANEW from 2000, NRC depends on Google Trans.)
- dependencies between raters
- the *Western*, *Educated*, *Industrialized*, *Rich*, *and Democratic* issue (LIWC is based on American undergraduates)
- uncorrected ratings

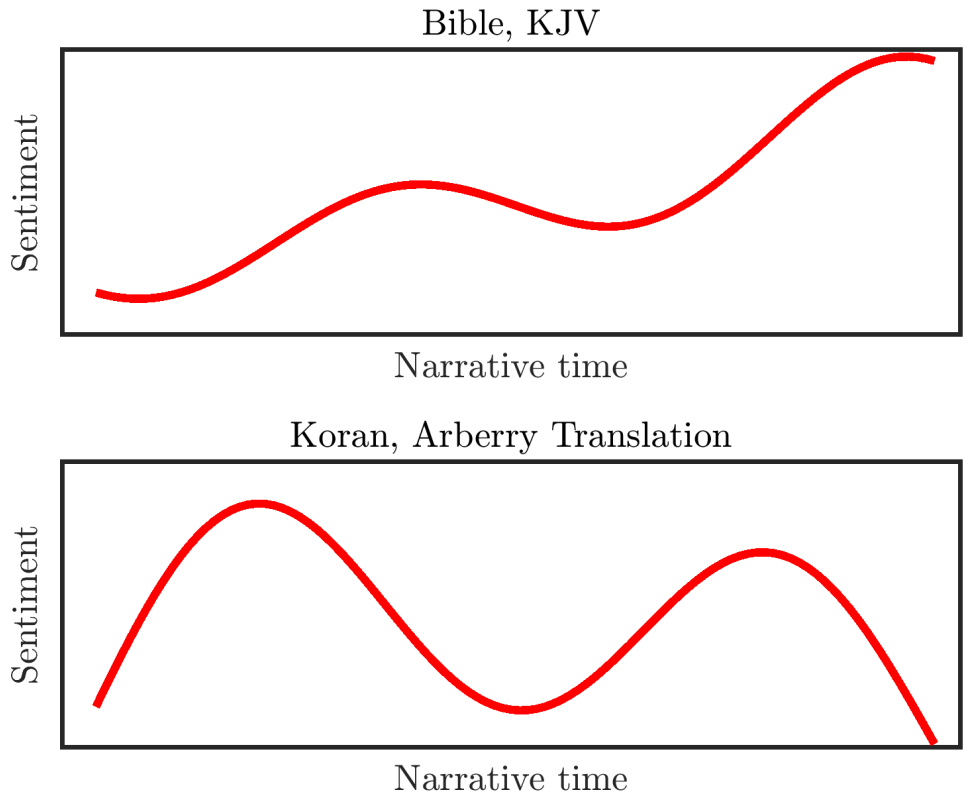
pros of dictionary-based sentiment analysis

- computationally efficient way of accessing affective components
- corpus agnostic in comparison to ML (can be applied without training)
- transparent technique that avoids black boxing the solution

cons of dictionary-based sentiment analysis

- bag-of-words assumption
- accuracy depends on large data sets (single sentence or paragraphs are useless)
- context sensitivity of word meaning ($miss_{\downarrow}$, $vice_{\downarrow}$) and negations ($\{not_{\downarrow}\ good_{\uparrow}\}_{neutral}$)
- lower accuracy than supervised learning (but supervised learning needs class information and is corpus dependent)


Sentiment Analysis of
The Bible
Arranged Chronologically
Positive Sentiment • Negative Sentiment
openbible.info • Headings from the ESV Bible
Patriarchs
Moses
Joshua
David
Monarchy
Return
Exile
Early Church
Jesus


Bible, KJV
Sentiment
Narrative time
Koran, Arberry Translation
Sentiment
Narrative time

language Assessment by Mechanical Turk (labMT)

```
1  import pandas as pd
2  url = 'http://www.plosone.org/article/fetchSingleRepresentation.action?uri=info:doi
3  10.1371/journal.pone.0026752.s001'
4  labmt = pd.read_csv(url, skiprows = 2, sep = '	', index_col = 0)
5
6  dictionary = labmt.happiness_average.to_dict()
7  labmtbar = labmt.happiness_average.mean()
8  dictionary_0 = (labmt.happiness_average - labmtbar).to_dict()
9
10 def sent_scr(string):
11     tokens = string.split()
12     return sum([dictionary.get(token.lower(), 0.0) for token in tokens]) / len(tokens)
13
14 sent = 'Did Crooked Hillary help disgusting (check out sex tape and past) Alicia M become a U.S.
15 citizen so she could use her in the debate?'
16
17 print sent_scr(sent)
```

Danish dictionary from Finn Årup Nielsen

```
1  from afinn import Afinn
2  afinn = Afinn(language='da')
3  sent = "Røvhul! Dumme svin! Du behandler mig som lort!
4          Nu kan du rende og hoppe jeg vil ikke tale med dig nogensinde igen"
5  print afinn.score(sent)
```

Matt Jockers' package for plot analysis

```
library(tm)
library(syuzhet)

text.v <- paste(scan(filepath, what = 'character', sep='\n', encoding = 'UTF-8'), collapse = " ")

text_sent <- get_sentences(text.v)

text_syuzhet <- get_sentiment(text_sent, method = 'syuzhet')

text_sent[which(text_syuzhet == max(text_syuzhet))]
text_sent[which(text_syuzhet == min(text_syuzhet))]
text_sent[which(text_syuzhet > (mean(text_syuzhet)+sd(text_syuzhet)*2))]
text_sent[which(text_syuzhet < (mean(text_syuzhet)-sd(text_syuzhet)*2))]

text_syuzhet_val <- get_percentage_values(text_syuzhet, bin = 100)
```

sentiment analysis ala Vermont (from labMTsimple/storyLab.py)

```
ignoreWords = ["nigga","nigger","niggaz","niggas"];

for word in ignore:
  ignoreWords.append(word)
newVec = copy.copy(tmpVec)
for i in range(len(score_list)):
  if abs(score_list[i]-center) < stopVal:
    newVec[i] = 0
  if word_list[i] in ignoreWords:
    newVec[i] = 0

return newVec
```