Introduction to Intelligent User Interfaces

Text and Natural Language Processing

Overview - NLP

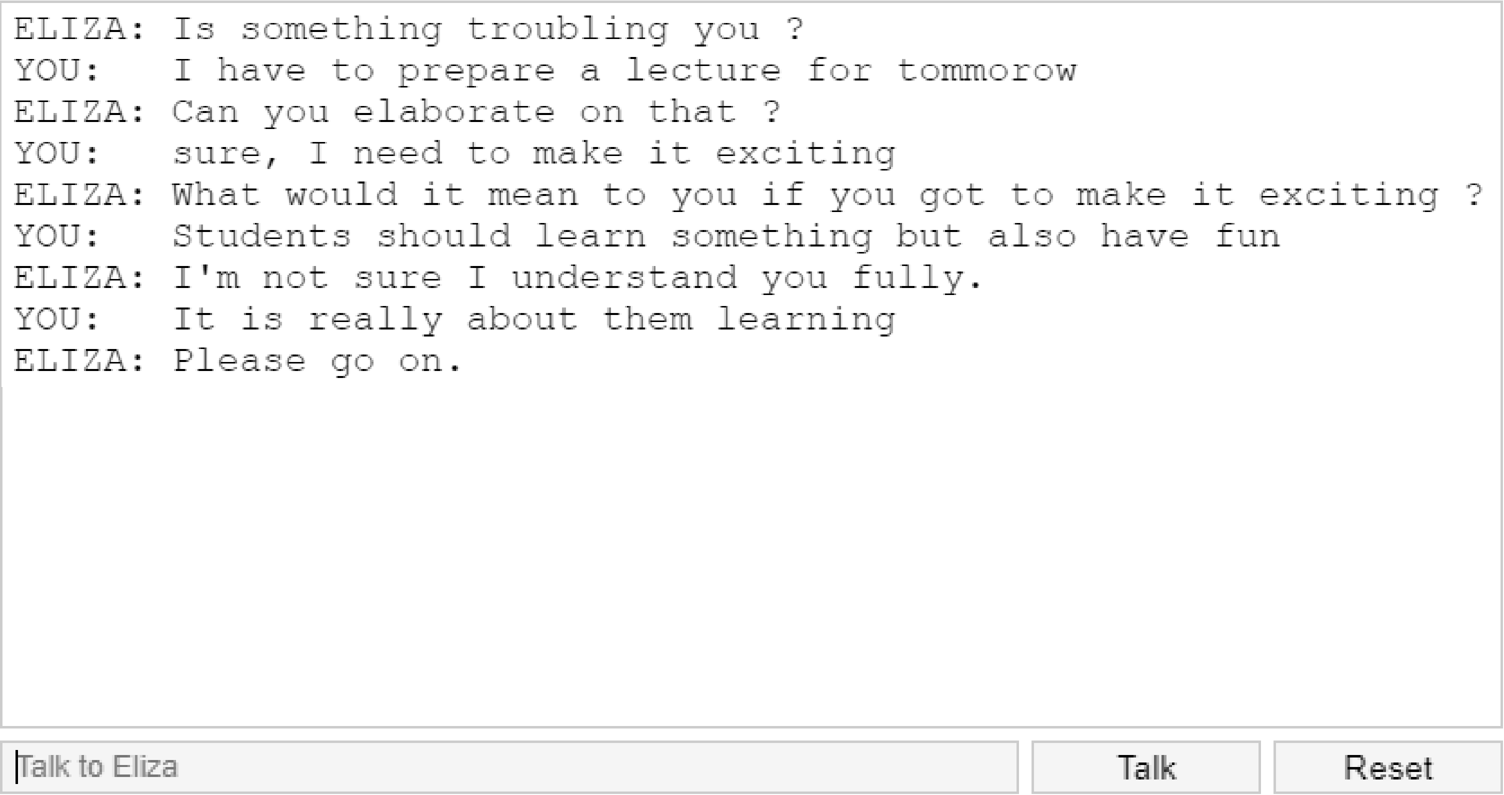
- Definition and Motivation
- Applications
- Terms, basic concepts and algorithms
- Milestones in the history of NLP
- Text Analytics

Overview - NLP

- Definition and Motivation
- Applications
- Terms, basic concepts and algorithms
- **Milestones in the history of NLP**
- Text Analytics

ELIZA by Joseph Weizenbaum

https://katzlberger.ai/2018/08/31/eliza-der-erste-chatbot/



Weizenbaum, Joseph. "ELIZA---a computer program for the study of natural language communication between man and machine." Communications of the ACM 9.1 (1966): 36-45.
https://cse.buffalo.edu/~rapaport/572/S02/weizenbaum.eliza.1966.pdf

ELIZA by Joseph Weizenbaum (1966)

- “ELIZA is a program […] which makes certain kinds of natural language conversation between man and computer possible. Input sentences are analyzed on the basis of decomposition rules which are triggered by key words appearing in the input text. Responses are generated by reassembly rules associated with selected decomposition rules. The fundamental technical problems with which ELIZA is concerned are:
- the identification of key words,
- the discovery of minimal context,
- the choice of appropriate transformations,
- generation of responses in the absence of keywords, and
- the provision of an ending capacity for ELIZA "scripts".”

Weizenbaum, Joseph. "ELIZA---a computer program for the study of natural language communication between man and machine." Communications of the ACM 9.1 (1966): 36-45. https://doi.org/10.1145/365153.365168

Eliza

- Try Eliza out at: http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm
- Source code in Python: https://github.com/wadetb/eliza

Can be run interactively:

```
$ python eliza.py
How do you do.  Please tell me your problem.
> I would like to have a chat bot.
You say you would like to have a chat bot ?
> bye
Goodbye.  Thank you for talking to me.
```

...or imported and used as a library:

```python
import eliza

eliza = eliza.Eliza()
eliza.load('doctor.txt')

print(eliza.initial())
while True:
    said = input('> ')
    response = eliza.respond(said)
    if response is None:
        break
    print(response)
print(eliza.final())
```

Breakout session

- http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm

- Explore two directions
  - Assume a conversation with a friend or therapeutic session
    - Start with: “How can I be more happy?”
  - Assume you have to answer questions about how many people live in European Capitals?

- What reactions do you see?

Task:
How to implement a chatbot for Alexa?

NLP timeline
three different types of approaches

- Since 1950s (early days of NLP):
  → **Rule-based Approaches**
- Since 1980s (statistical approaches):
  → **Machine Learning Approaches**
- Since 2010s (advances in neural networks):
  → **Deep Learning Approaches**

https://livebook.manning.com/book/essential-natural-language-processing/chapter-1/15

Overview - NLP

- Definition and Motivation
- Applications
- Terms, basic concepts and algorithms
- Milestones in the history of NLP
- **Text Analytics**

Text Analytics

- Text is a key media:
  - in personal communication (e.g. texting, email)
  - in communication media (e.g. news, web pages, social media)
  - for knowledge sharing and acquisition (e.g. books, reports)

- Most user interfaces include texts
- Text reception (reading, understanding, or skimming) is often a key factor that defines the require time for a (knowledge work) task
- Big individual differences in text reception (e.g. reading speed, understanding)

Definitions of text analytics

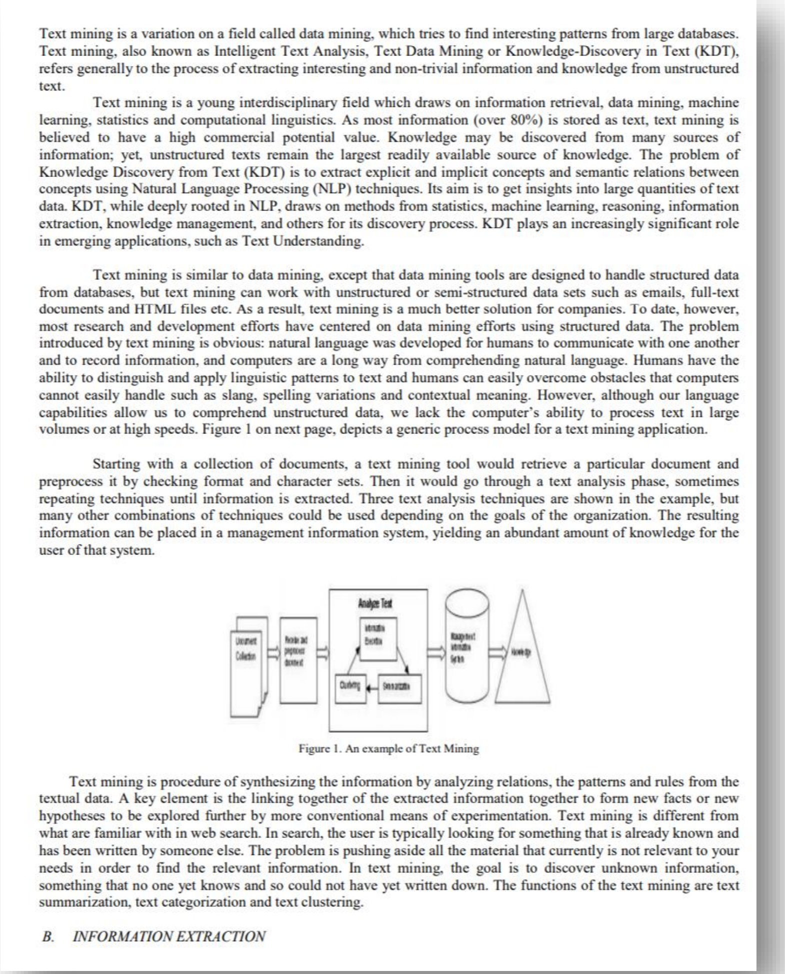
- Definition of “text data mining”: “as the application of algorithms and methods from the fields machine learning and statistics to texts with the goal of finding useful patterns” [1]
- *“Text mining* is the process of analyzing collections of textual materials in order to capture key concepts and themes and uncover hidden relationships and trends without requiring that you know the precise words or terms that authors have used to express those concepts.” [2]

[1] Hotho, Andreas, Andreas Nürnberger, and Gerhard Paaß. "A brief survey of text mining." *Ldv Forum*. Vol. 20. No. 1. 2005.
[2] https://www.ibm.com/support/knowledgecenter/en/SS3RA7_17.1.0/ta_guide_ddita/textmining/shared_entities/tm_intro_tm_defined.html

Text analytics – Why and Where?

- Answering questions like
  - What is this text about?
  - What did the person communicate?
  - What is the key information in this document?
  - What feelings are communicated?
  - Who is saying something?
  - Is this different from what was said before?

- Application areas
  - Social media analytics, e.g. twitter
  - Communication and reading interfaces
  - Customer reviews and feedback
  - Chat bots
  - Forensics



Text mining is a variation on a field called data mining, which tries to find interesting patterns from large databases. Text mining, also known as Intelligent Text Analysis, Text Data Mining or Knowledge-Discovery in Text (KDT), refers generally to the process of extracting interesting and non-trivial information and knowledge from unstructured text.

Text mining is a young interdisciplinary field which draws on information retrieval, data mining, machine learning, statistics and computational linguistics. As most information (over 80%) is stored as text, text mining is believed to have a high commercial potential value. Knowledge may be discovered from many sources of information; yet, unstructured texts remain the largest readily available source of knowledge. The problem of Knowledge Discovery from Text (KDT) is to extract explicit and implicit concepts and semantic relations between concepts using Natural Language Processing (NLP) techniques. Its aim is to get insights into large quantities of text data. KDT, while deeply rooted in NLP, draws on methods from statistics, machine learning, reasoning, information extraction, knowledge management, and others for its discovery process. KDT plays an increasingly significant role in emerging applications, such as Text Understanding.

Text mining is similar to data mining, except that data mining tools are designed to handle structured data from databases, but text mining can work with unstructured or semi-structured data sets such as emails, full-text documents and HTML files etc. As a result, text mining is a much better solution for companies. To date, however, most research and development efforts have centered on data mining efforts using structured data. The problem introduced by text mining is obvious: natural language was developed for humans to communicate with one another and to record information, and computers are a long way from comprehending natural language. Humans have the ability to distinguish and apply linguistic patterns to text and humans can easily overcome obstacles that computers cannot easily handle such as slang, spelling variations and contextual meaning. However, although our language capabilities allow us to comprehend unstructured data, we lack the computer's ability to process text in large volumes or at high speeds. Figure 1 on next page, depicts a generic process model for a text mining application.

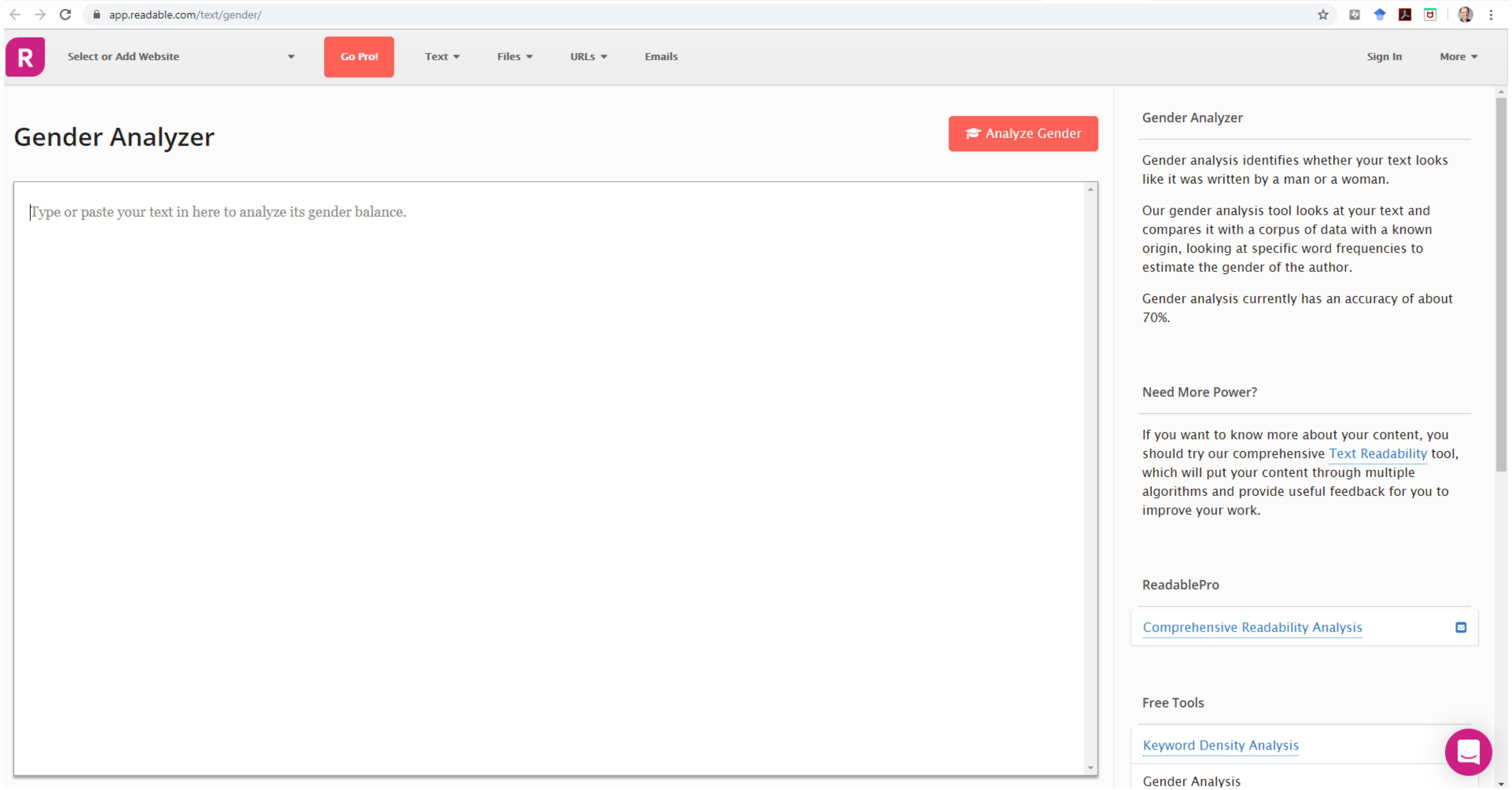
Starting with a collection of documents, a text mining tool would retrieve a particular document and preprocess it by checking format and character sets. Then it would go through a text analysis phase, sometimes repeating techniques until information is extracted. Three text analysis techniques are shown in the example, but many other combinations of techniques could be used depending on the goals of the organization. The resulting information can be placed in a management information system, yielding an abundant amount of knowledge for the user of that system.

Figure 1. An example of Text Mining

Text mining is procedure of synthesizing the information by analyzing relations, the patterns and rules from the textual data. A key element is the linking together of the extracted information together to form new facts or new hypotheses to be explored further by more conventional means of experimentation. Text mining is different from what are familiar with in web search. In search, the user is typically looking for something that is already known and has been written by someone else. The problem is pushing aside all the material that currently is not relevant to your needs in order to find the relevant information. In text mining, the goal is to discover unknown information, something that no one yet knows and so could not have yet written down. The functions of the text mining are text summarization, text categorization and text clustering.

B. INFORMATION EXTRACTION

http://ijiet.com/wp-content/uploads/2015/04/17.pdf



https://app.readable.com/text/gender/

http://www.hackerfactor.com/GenderGuesser.php

Gender Analyzer

nature International weekly journal of science

nature news home | news archive | specials | opinion | features | news blog | natu

comments on this story

Stories by subject

- Technology
- Brain and behaviour

Stories by keywords

- linguistics
- gender
- male
- female
- literature
- book
- computer
- program
- algorithm

This article elsewhere

Blogs linking to this article

Add to Digg

Add to Facebook

Add to Newsvine

Add to Del.icio.us

Add to Twitter

Published online 18 July 2003 | Nature | doi:10.1038/news030714-13

News

Computer program detects author gender

Simple algorithm suggests words and syntax bear sex and genre stamp.

Philip Ball

A.S Byatt confuses the computer; will it see through George Elliot?

A new computer program can tell whether a book was written by a man or a woman. The simple scan of key words and syntax is around 80% accurate on both fiction and non-fiction[1,2].

The program's success seems to confirm the stereotypical perception of differences in male and female language use. Crudely put, men talk more about objects, and women more about relationships.

Female writers use more pronouns (I, you, she, their, myself), say the program's developers, Moshe Koppel of Bar-Ilan University in Ramat Gan, Israel, and colleagues. Males prefer words that identify or determine nouns (a, the, that) and words that quantify them (one, two, more).

So this article would already, through sentences such as this, have probably betrayed its author as male: there is a prevalence of plural pronouns (they, them), indicating the male tendency to categorize rather than personalize.

https://www.nature.com/news/2003/030714/full/news030714-13.html

https://app.readable.com/text/gender/

Recent Trends in Digital Text Forensics and its Evaluation

Plagiarism Detection, Author Identification, and Author Profiling

Tim Gollub,[1] Martin Potthast,[1] Anna Beyer,[1] Matthias Busse,[1]
Francisco Rangel,[2,3] Paolo Rosso,[3] Efstathios Stamatatos,[4] and Benno Stein[1]

[1]Web Technology & Information Systems, Bauhaus-Universität Weimar, Germany
[2]Autoritas Consulting, S.A., Spain
[3]Natural Language Engineering Lab, ELiRF, Universitat Politècnica de València, Spain
[4]Dept. of Information & Communication Systems Engineering, University of the Aegean, Greece

pan@webis.de http://pan.webis.de

Abstract This paper outlines the concepts and achievements of our evaluation lab on digital text forensics, PAN 13, which called for original research and development on plagiarism detection, author identification, and author profiling. We present a standardized evaluation framework for each of the three tasks and discuss the evaluation results of the altogether 58 submitted contributions. For the first time, instead of accepting *the output* of software runs, we collected *the softwares themselves* and run them on a computer cluster at our site. As evaluation and experimentation platform we use TIRA, which is being developed at the Webis Group in Weimar. TIRA can handle large-scale software submissions by means of virtualization, sandboxed execution, tailored unit testing, and staged submission. In addition to the achieved evaluation results, a major achievement of our lab is that we now have the largest collection of state-of-the-art approaches with regard to the mentioned tasks for further analysis at our disposal.

1 Introduction

Nowadays, people increasingly share their work online, contribute to open projects and engage in web-based social interactions. The ease and the anonymity with which all of this can be done raises concerns about verifiability and trust: is a given text an original? Is an author the one who she claims to be? Does a piece of information originate from a

https://webis.de/downloads/publications/papers/stein_2013g.pdf

Discussion

Gollub, Tim, et al. "Recent trends in digital text forensics and its evaluation." *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, Berlin, Heidelberg, 2013.

Text analytics – typical tasks?

- Language detection
- Named entity extraction
- Detecting themes, categories, topics
- Detecting intentions
- Sentiment analysis
- Document summarization
- Basis for translation

Text analytics – Identification of the Language

- Can tell what language the text is, e.g. English, German, Spanish,…
- Relevant for understanding and translation

- Example (Online) APIs:
  - https://console.bluemix.net/apidocs/language-translator
  - https://docs.microsoft.com/en-us/azure/cognitive-services/translator/
  - https://cloud.google.com/translate/docs/detecting-language
  - https://pypi.org/project/langdetect/
- Language identification using NLTK, examples
  - https://avital.ca/notes/language-identification-using-nltk
  - http://www.algorithm.co.il/blogs/programming/python/cheap-language-detection-nltk/

Identification of the Language

https://pypi.org/project/langdetect/

```
pip install langdetect
```

```
[4] from langdetect import detect
    detect("Hallo! Wie geht es ihnen?")
```

```
'de'
```

```
[7] from langdetect import detect_langs
    detect_langs("Na was is hier los?")
```

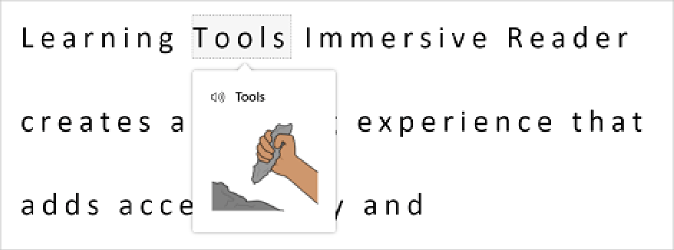
```
[en:0.5714265025185337, af:0.4285734406497653]
```

Text Analytics and Interaction
Immersive Reader

https://docs.microsoft.com/en-us/azure/cognitive-services/immersive-reader/overview

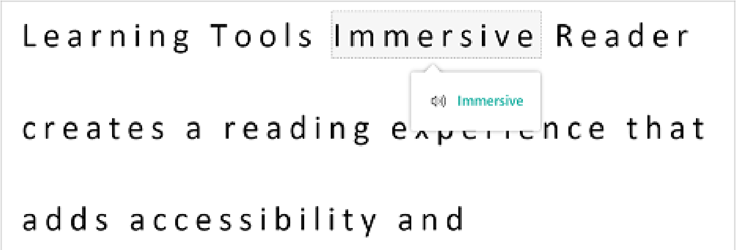
Display pictures for common words

For commonly used terms, the Immersive Reader will display a picture.



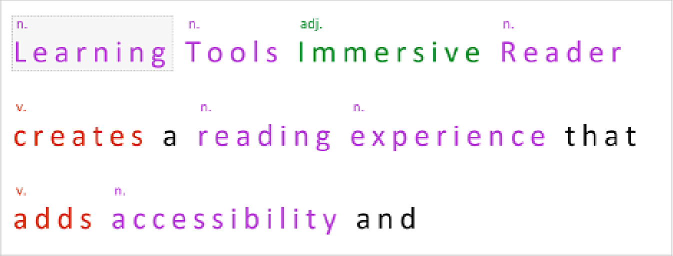
Read content aloud

Speech synthesis (or text-to-speech) is baked into the Immersive Reader service, which lets your readers select text to be read aloud.



Highlight parts of speech

Immersive Reader can be use to help learners understand parts of speech and grammar by highlighting verbs, nouns, pronouns, and more.



Text analytics – Sentiment analysis
Example – how to… classify reviews?

Klare Kaufempfehlung
29. November 2017
Format: Kindle Ausgabe | Verifizierter Kauf

Wer sich mit dem Thema künstliche Intelligenz und Neuronalen Netzen beschäftigen und auch selbst anhand von Python ein Neuronales Netz zur Schriftzeichen-Erkennung programmieren möchte, liegt hier Gold richtig.

Gute Erklärung der Grundlagen, ohne zu theoretisch zu werden. Schneller Einstieg in die Praxis mit gut verständlichen Code-Beispielen ohne unnötigen Ballast. Grundlegende Python Kenntnisse sind hilfreich aber nicht unbedingt erforderlich.

Kundenbilder

Alle Kundenbilder anzeigen

Lesen Sie Rezensionen, die folgende Stichworte enthalten

neuronalen netzen | neuronales netz | schritt schritt | neuronaler netze | tariq rashid | neuronalen netzes | zweiten teil | thema neuronale | künstliche intelligenz | grundlagen neuronaler | mathematischen grundlagen

67 Kundenrezensionen

Spitzenrezensionen

kkr

Ausgesprochen hilfreiches Einsteigerbuch
16. Januar 2018
Format: Taschenbuch | Verifizierter Kauf

Mir hat das Buch sehr dabei geholfen praktisch ohne Vorkenntnisse (ich konnte kein Python, nur etwas Pascal und C, noch aus den Neunzigern) und mit mäßigem mathematischem Hintergrund das Thema neuronale Netze zu erfassen und - und darum ging es mir in erster Linie - selbst ein neuronales Netz in Python zusammenzubasteln, um damit für ein Projekt Bilder zu erkennen. Das Ergebnis ist überraschend gut und ich kann dieses Buch dementsprechend begeistert weiter empfehlen.

Das neuronale Netz Herrn Rashids funktioniert ohne auf Frameworks wie Tensorflow oder Torch zurückzugreifen und ohne GPU-Unterstützung. Theoretisch kann man es auf einem Raspberry Pi berechnen (was allerdings quälend langsam ist), ein einfaches Notebook ohne Nvidia Graphikkarte reicht aber völlig aus.

Für den praktischen Einstieg ist das Buch perfekt und hebt sich deutlich von allem anderen ab, was der deutsch- und englischsprachige Markt so her gibt. Im Übrigen ist auch das Blog des Autors sehr zu empfehlen.

39 Personen fanden diese Informationen hilfreich

Nützlich | Kommentar | Missbrauch melden

Trifft keine Zielgruppe (nichts Halbes und nichts Ganzes)
5. März 2019
Format: Taschenbuch | Verifizierter Kauf

Das Buch beginnt als ob es für Kinder geschrieben ist. Mit sehr vielen unnötigen Wiederholungen. Scheinbar merkt der Autor aber recht schnell, dass es sich um ein komplexes Thema handelt und pendelt dann zwischen sehr leicht oder sehr knapp und anspruchsvoll. Er setzt mindestens Mathe-Leistungskurs(Abitur) voraus.
Ich habe das Buch vor allem wegen der Vielzahl an Wiederholungen als sehr anstrengend empfunden. Zeitgleich denke ich nicht, dass Laien den mathematischen Ausführungen folgen können.

Text analytics – Sentiment analysis

- In the sentiment analysis the algorithm determines if text is positive, neutral, or negative
- Used to analyze reports, social media posts, customer reviews, forums, news items, communication, etc.
- Typically a text is broken up in parts (e.g. sentences or phrases) and for each part the sentiment is estimated. The score for the parts is then combined to get an overall score
- Sentiment library and rules
  - Sentiment library (collection of adjectives and phrases that are either positive or negative, e.g. good, brilliant, great, amazing)
  - Rules are used to assign a sentiment score based on the library and rules
- Typical problems
  - “Not good”, “the cake wasn’t bad”, …

Example: https://text-processing.com/demo/sentiment/

The cake is good

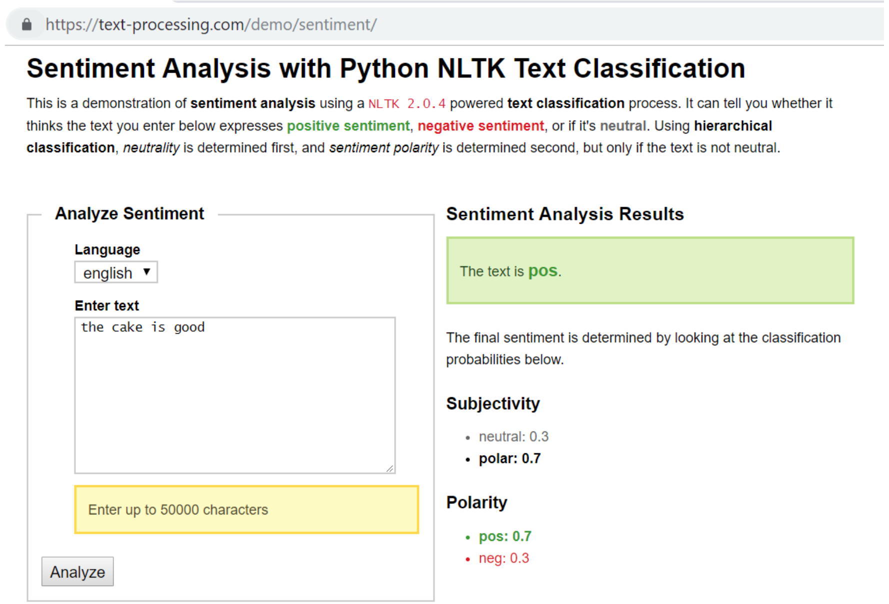
→ pos: 0.7

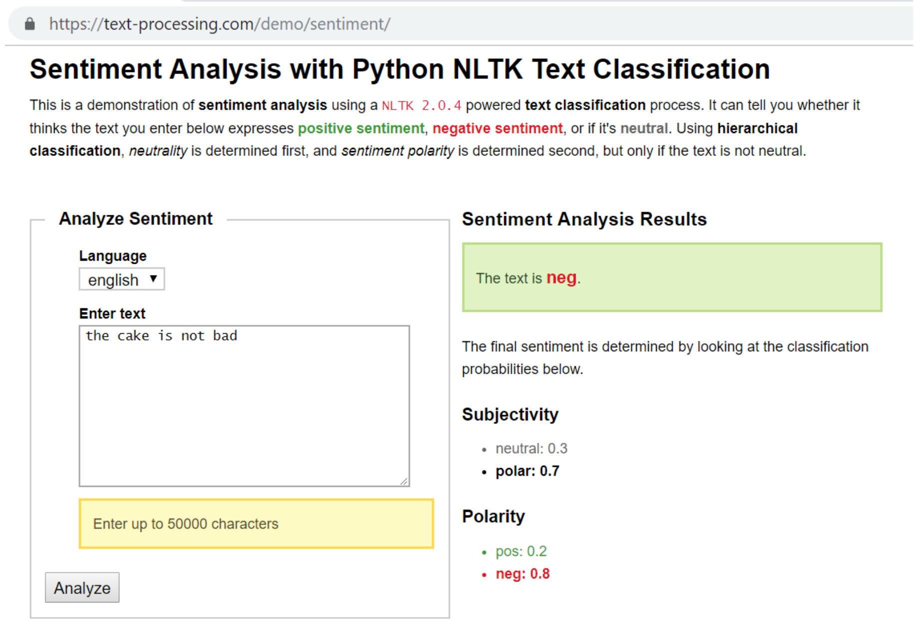
→ neg: 0.3

The cake is not bad

→ pos: 0.2

→ neg: 0.8





Sentiment analysis

```
[11] import nltk
     nltk.download('vader_lexicon')
     from nltk.sentiment.vader import SentimentIntensityAnalyzer
     sid = SentimentIntensityAnalyzer()
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

```
[16] text1 = "I feel not great today!"
     text2 = "The cake was not bad."

     sid.polarity_scores(text2)
```

```
{'compound': -0.5423, 'neg': 0.538, 'neu': 0.462, 'pos': 0.0}
```

NLTK vader

https://www.nltk.org/_modules/nltk/sentiment/vader.html

god	1.1	1.51327	[0, 0, 0, 1, 0, 3, 0, 3, 0, 4]
goddam	-2.5	1.28452	[0, -3, -3, -4, -3, -1, -4, -1, -3, -3]
goddammed	-2.4	0.91652	[-2, -3, -1, -1, -2, -2, -4, -3, -3, -3]
goddamn	-2.1	1.75784	[-3, -3, -2, -4, -4, -3, -3, -1, 1, 1]
goddamned	-1.8	2.03961	[-3, -3, -3, -4, -1, 2, -2, -3, 2, -3]
goddamns	-2.1	1.51327	[-3, -2, -4, 2, -2, -3, -3, -2, -2, -2]
goddams	-1.9	1.92094	[-3, -3, -2, -4, -4, -2, -3, -1, 2, 1]
godsend	2.8	0.87178	[2, 3, 3, 2, 4, 3, 3, 1, 4, 3]
good	1.9	0.9434	[2, 1, 1, 3, 2, 4, 2, 2, 1, 1]
goodness	2.0	1.54919	[2, 2, 2, 3, 1, 2, -2, 4, 3, 3]
gorgeous	3.0	0.63246	[3, 3, 2, 3, 3, 3, 4, 4, 3, 2]
gorgeously	2.3	0.78102	[2, 2, 2, 3, 1, 2, 4, 3, 2, 2]
gorgeousness	2.9	0.9434	[3, 4, 3, 1, 4, 4, 2, 2, 3, 3]

Sentiment ratings from 10 independent human raters […]. Over 9,000 token features were rated on a scale from "[–4] Extremely Negative" to "[4] Extremely Positive", with allowance for "[0] Neutral (or Neither, N/A)". We kept every lexical feature that had a non-zero mean rating, and whose standard deviation was less than 2.5 as determined by the aggregate of those ten independent raters. This left us with just over 7,500 lexical features […] For example, the word "okay" has a positive valence of 0.9, "good" is 1.9, and "great" is 3.1, whereas "horrible" is –2.5, the frowning emoticon :(is –2.2, and "sucks" and it's slang derivative "sux" are both –1.5.

NLTK vader

https://www.nltk.org/_modules/nltk/sentiment/vader.html

```
# booster/dampener 'intensifiers' or 'degree adverbs'
# http://en.wiktionary.org/wiki/Category:English_degree_adverbs

BOOSTER_DICT = {
    "absolutely": B_INCR,
    "amazingly": B_INCR,
    "awfully": B_INCR,
    "completely": B_INCR,
    "considerably": B_INCR,
    "decidedly": B_INCR,
    "deeply": B_INCR,
    "effing": B_INCR,
    "enormously": B_INCR,
    "entirely": B_INCR,
    "especially": B_INCR,
    "exceptionally": B_INCR,
    "extremely": B_INCR,
    "fabulously": B_INCR,
    "flipping": B_INCR,
    "flippin": B_INCR,
    "fricking": B_INCR,
    "frickin": B_INCR,
    "frigging": B_INCR,
    "friggin": B_INCR,
    "fully": B_INCR,
    "fucking": B_INCR,
```

```
SPECIAL_CASE_IDIOMS = {
    "the shit": 3,
    "the bomb": 3,
    "bad ass": 1.5,
    "yeah right": -2,
    "cut the mustard": 2,
    "kiss of death": -1.5,
    "hand to mouth": -2,
}
```

```
def negated(input_words, include_nt=True):
    """
    Determine if input contains negation words
    """
    neg_words = NEGATE
    if any(word.lower() in neg_words for word in input_words):
        return True
    if include_nt:
        if any("n't" in word.lower() for word in input_words):
            return True
    for first, second in pairwise(input_words):
        if second.lower() == "least" and first.lower() != 'at':
            return True
    return False
```

```
NEGATE = {
    "aint",
    "arent",
    "cannot",
    "cant",
    "couldnt",
    "darent",
    "didnt",
    "doesnt",
    "ain't",
    "aren't",
    "can't",
    "couldn't",
    "daren't",
    "didn't",
    "doesn't",
    "dont",
    "hadnt",
    "hasnt",
    "havent",
    "isnt",
    "mightnt",
    "mustnt",
    "neither",
    "don't",
    "hadn't",
    "hasn't",
    "haven't",
    "isn't",
    "mightn't",
    "mustn't",
    "neednt",
    "needn't",
```

http://t-redactyl.io/blog/2017/04/using-vader-to-handle-sentiment-analysis-with-social-media-text.html

Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

```python
def negated(input_words, include_nt=True):
    """
    Determine if input contains negation words
    """
    neg_words = NEGATE
    if any(word.lower() in neg_words for word in input_words):
        return True
    if include_nt:
        if any("n't" in word.lower() for word in input_words):
            return True
    for first, second in pairwise(input_words):
        if second.lower() == "least" and first.lower() != 'at':
            return True
    return False
```

```python
NEGATE = {
    "aint",
    "arent",
    "cannot",
    "cant",
    "couldnt",
    "darent",
    "didnt",
    "doesnt",
    "ain't",
    "aren't",
    "can't",
    "couldn't",
    "daren't",
    "didn't",
    "doesn't",
    "dont",
    "hadnt",
    "hasnt",
    "havent",
    "isnt",
    "mightnt",
    "mustnt",
    "neither",
    "don't",
    "hadn't",
    "hasn't",
    "haven't",
    "isn't",
    "mightn't",
    "mustn't",
    "neednt",
    "needn't",
```

NLTK vader: https://www.nltk.org/_modules/nltk/sentiment/vader.html

Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

http://t-redactyl.io/blog/2017/04/using-vader-to-handle-sentiment-analysis-with-social-media-text.html

Text analytics – Sentiment analysis

- Sentiment Analysis with Python NLTK Text Classification (online example)
  https://text-processing.com/demo/sentiment/
- Twitter Sentiment Analysis using Python
  https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/
- nltk.sentiment.sentiment_analyzer module
  facilitate Sentiment Analysis tasks using NLTK features and classifiers
  https://www.nltk.org/api/nltk.sentiment.html
- https://www.kaggle.com/ngyptr/python-nltk-sentiment-analysis

Text analytics – Summarization

- For a given text a short version is created that keep a maximum of the content and should still relay the same message
- Important, especially if dealing with a lot of text (reports, social media, communication)
- Optimum: Reduce text in a way that only the relevant information remains
- Applications:
  - Reduce reading time for human reader
  - Improve indexing of documents
  - Simplify overview of larger texts and collections
- Manual text summarization is common
  - Headings in newspapers, synopses from a book, abstracts in papers, reviews of a film or book

- See: https://machinelearningmastery.com/gentle-introduction-text-summarization/

Text analytics – Summarization Example

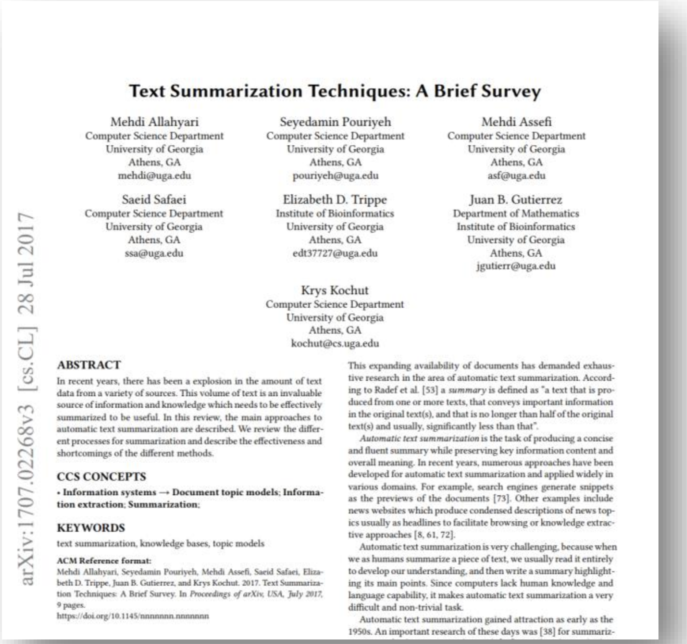
“So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.”

Task: Summarize the above paragraph in 1 sentence. The sentence must not be longer than 10 words.

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization

- Approaches:
  - **Extraction**: “identifying important sections of the text and generating them verbatim; thus, they depend only on extraction of sentences from the original text”.
  - **Abstraction**: “aim at producing important material in a new way. In other words, they interpret and examine the text using advanced natural language techniques in order to generate a new shorter text that conveys the most critical information from the original text.”
- Extraction is easier in typically better than abstraction as it requires less semantic understanding



arXiv:1707.02268v3 [cs.CL] 28 Jul 2017

Text Summarization Techniques: A Brief Survey

Mehdi Allahyari, Computer Science Department, University of Georgia, Athens, GA, mehdi@uga.edu

Seyedamin Pouriyeh, Computer Science Department, University of Georgia, Athens, GA, pouriyeh@uga.edu

Mehdi Assefi, Computer Science Department, University of Georgia, Athens, GA, asf@uga.edu

Saeid Safaei, Computer Science Department, University of Georgia, Athens, GA, ssa@uga.edu

Elizabeth D. Trippe, Institute of Bioinformatics, University of Georgia, Athens, GA, edt37727@uga.edu

Juan B. Gutierrez, Department of Mathematics, Institute of Bioinformatics, University of Georgia, Athens, GA, jgutierr@uga.edu

Krys Kochut, Computer Science Department, University of Georgia, Athens, GA, kochut@cs.uga.edu

ABSTRACT

In recent years, there has been a explosion in the amount of text data from a variety of sources. This volume of text is an invaluable source of information and knowledge which needs to be effectively summarized to be useful. In this review, the main approaches to automatic text summarization are described. We review the different processes for summarization and describe the effectiveness and shortcomings of the different methods.

CCS CONCEPTS

• **Information systems → Document topic models**; **Information extraction**; **Summarization**;

KEYWORDS

text summarization, knowledge bases, topic models

ACM Reference format:

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. Text Summarization Techniques: A Brief Survey. In *Proceedings of arXiv, USA, July 2017,* 9 pages.

https://doi.org/10.1145/nnnnnnn.nnnnnnn

This expanding availability of documents has demanded exhaustive research in the area of automatic text summarization. According to Radef et al. [53] a *summary* is defined as "a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually, significantly less than that".

Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning. In recent years, numerous approaches have been developed for automatic text summarization and applied widely in various domains. For example, search engines generate snippets as the previews of the documents [73]. Other examples include news websites which produce condensed descriptions of news topics usually as headlines to facilitate browsing or knowledge extractive approaches [8, 61, 72].

Automatic text summarization is very challenging, because when we as humans summarize a piece of text, we usually read it entirely to develop our understanding, and then write a summary highlighting its main points. Since computers lack human knowledge and language capability, it makes automatic text summarization a very difficult and non-trivial task.

Automatic text summarization gained attraction as early as the 1950s. An important research of these days was [38] for summariz-

Mehdi Allahyari et al. Text Summarization Techniques: A Brief Survey. 2017. arXiv:1707.02268 [cs.CL]

Text analytics – Summarization Example

So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.

Step 1: Convert Paragraphs to Sentences

1. So, keep working
2. Keep striving
3. Never give up
4. Fall down seven times, get up eight
5. Ease is a greater threat to progress than hardship
6. Ease is a greater threat to progress than hardship
7. So, keep moving, keep growing, keep learning
8. See you at work

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization Example

> So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.

Step 1: Convert Paragraphs to Sentences

1. So, keep working
2. Keep striving
3. Never give up
4. Fall down seven times, get up eight
5. Ease is a greater threat to progress than hardship
6. Ease is a greater threat to progress than hardship
7. So, keep moving, keep growing, keep learning
8. See you at work

Step 2: Text Preprocessing

1. keep working
2. keep striving
3. never give
4. fall seven time get eight
5. ease greater threat progress hardship
6. ease greater threat progress hardship
7. keep moving keep growing keep learning
8. see work

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization Example

> *So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.*

Step 1: Convert Paragraphs to Sentences

1. So, keep working
2. Keep striving
3. Never give up
4. Fall down seven times, get up eight
5. Ease is a greater threat to progress than hardship
6. Ease is a greater threat to progress than hardship
7. So, keep moving, keep growing, keep learning
8. See you at work

Step 2: Text Preprocessing

1. keep working
2. keep striving
3. never give
4. fall seven time get eight
5. ease greater threat progress hardship
6. ease greater threat progress hardship
7. keep moving keep growing keep learning
8. see work

Step 3. Tokens

```
['keep',
 'working',
 'keep',
 'striving',
 'never',
 'give',
 'fall',
 'seven',
 'time',
 'get',
 'eight',
 'ease',
 'greater',
 'threat',
 'progress',
 'hardship',
 'ease',
 'greater',
 'threat',
 'progress',
 'hardship',
 'keep',
 'moving',
 'keep',
 'growing',
 'keep',
 'learning',
 'see',
 'work']
```

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization Example

Step 3. Tokens

```
['keep',
 'working',
 'keep',
 'striving',
 'never',
 'give',
 'fall',
 'seven',
 'time',
 'get',
 'eight',
 'ease',
 'greater',
 'threat',
 'progress',
 'hardship',
 'ease',
 'greater',
 'threat',
 'progress',
 'hardship',
 'keep',
 'moving',
 'keep',
 'growing',
 'keep',
 'learning',
 'see',
 'work']
```

Step 4: Find weighted frequency of occurrence

Word	Frequency	Weighted Frequency
ease	2	0.40
eight	1	0.20
fall	1	0.20
get	1	0.20
give	1	0.20
greater	2	0.40
growing	1	0.20
hardship	2	0.40
keep	5	1.00
learning	1	0.20
moving	1	0.20
never	1	0.20
progress	2	0.40
see	1	0.20
seven	1	0.20
striving	1	0.20
threat	2	0.40
time	1	0.20
work	1	0.20
working	1	0.20

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization Example

Step 4: Find weighted frequency of occurrence

Word	Frequency	Weighted Frequency
ease	2	0.40
eight	1	0.20
fall	1	0.20
get	1	0.20
give	1	0.20
greater	2	0.40
growing	1	0.20
hardship	2	0.40
keep	5	1.00
learning	1	0.20
moving	1	0.20
never	1	0.20
progress	2	0.40
see	1	0.20
seven	1	0.20
striving	1	0.20
threat	2	0.40
time	1	0.20
work	1	0.20
working	1	0.20

> *So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.*

5. Replace Words by Weighted Frequency in Original Sentences

Sentence	Sum of Weighted Frequencies
So, keep working	1 + 0.20 = 1.20
Keep striving	1 + 0.20 = 1.20
Never give up	0.20 + 0.20 = 0.40
Fall down seven times, get up eight	0.20 + 0.20 + 0.20 + 0.20 + 0.20 = 1.0
Ease is a greater threat to progress than hardship	0.40 + 0.40 + 0.40 + 0.40 + 0.40 = 2.0
Ease is a greater threat to progress than hardship	0.40 + 0.40 + 0.40 + 0.40 + 0.40 = 2.0
So, keep moving, keep growing, keep learning	1 + 0.20 + 1 + 0.20 + 1 + 0.20 = 3.60
See you at work	0.20 + 0.20 = 0.40

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization Example

> So, keep working. Keep striving. Never give up. Fall down seven times, get up eight. Ease is a greater threat to progress than hardship. Ease is a greater threat to progress than hardship. So, keep moving, keep growing, keep learning. See you at work.

5. Replace Words by Weighted Frequency in Original Sentences

Sentence	Sum of Weighted Frequencies
So, keep working	1 + 0.20 = 1.20
Keep striving	1 + 0.20 = 1.20
Never give up	0.20 + 0.20 = 0.40
Fall down seven times, get up eight	0.20 + 0.20 + 0.20 + 0.20 + 0.20 = 1.0
Ease is a greater threat to progress than hardship	0.40 + 0.40 + 0.40 + 0.40 + 0.40 = 2.0
Ease is a greater threat to progress than hardship	0.40 + 0.40 + 0.40 + 0.40 + 0.40 = 2.0
So, keep moving, keep growing, keep learning	1 + 0.20 + 1 + 0.20 + 1 + 0.20 = 3.60
See you at work	0.20 + 0.20 = 0.40

6. Results

> So, keep moving, keep growing, keep learning

> So, keep moving, keep growing, keep learning. Ease is a greater threat to progress than hardship.

Example from https://stackabuse.com/text-summarization-with-nltk-in-python/

Text analytics – Summarization

- Explanation: https://rare-technologies.com/text-summarization-in-python-extractive-vs-abstractive-techniques-revisited/
- Example: https://stackabuse.com/text-summarization-with-nltk-in-python/
  https://medium.com/jatana/unsupervised-text-summarization-using-sentence-embeddings-adb15ce83db1
- Libraries and Tools
  - “PyTeaser takes any news article and extract a brief summary from it. It's based on the original Scala project.” https://github.com/xiaoxu193/PyTeaser

Breakout Group
Responsive Design for Text

- Example: Email client that summarizes text when resized

Recap
… do you remember what we did last time?

- Tokenization
- Stop Words Removal
- Text normalization
- Stemming / Porter Stemmer
- Lemmatization
- Part-Of-Speech Tagging
- Named Entity Disambiguation
- Named Entity Extraction
- Bag of Words
- Corpus, Corpora