NATURAL LANGUAGE PROCESSING

Lecture Topics
Basic outline

OTH Amberg-Weiden

- Information retrieval
- Basic text processing (regular expressions, string operations, …)
- Advanced text processing (segmentation, tokenization, …)
- Analytics of text: classification, unsupervised methods
- Typical text applications: Question Answering, Summarization, Translation, …
- Transformers and GPT models (basics of Chat-GPT and other popular models)

Organization of Lectures

OTH Amberg-Weiden

- The material is prepared by
  - Prof. Dr. Christian Bergler
  - Prof. Dr. Patrick Levi

- Lectures are a mixture of
  - theory (presentation) and
  - practical work and
  - journal club for advanced topics.

Organization of Lectures
Practical Parts

OTH Amberg-Weiden

- You will work on tasks, studies, … in every lecture
- You present your results and we talk about it in the group.
- You get a certain time frame in the lecture to do the tasks, I will be available for questions and support.
- You complete unfinished tasks at home.
- You get a rough (!) solution for (most) practical works. But with a certain delay.
- **If you skip these practical parts or just “ChatGPT” yourself through the solution without understanding it deeply, you are likely to miss proficiency in NLP and, consequently, the required skills to pass the final exam.**
- **You need a computer with a running Python environment in every lecture.**

Prerequisites

OTH Amberg-Weiden

- Mathematical skills
  - Calculus, Linear Algebra, Probability Theory & Statistics
- Machine Learning Skills
  - ML Algorithms and methods, data preparation, Deep Learning (!)
- Programming Skills
  - Fluency in Python absolutely required.
- This is neither a basic ML course nor a programming course. If you lack the required knowledge, you are likely to fail this course.

Final Exam
Project Work

OTH Amberg-Weiden

- Release date and deadline will be announced in the next weeks.
- You will realize a project related to NLP.
- You will need to apply the methods from the lecture but also develop new skills and apply them during the task.
- Individual or group work will be announced later.
  - Group work will increase the requirements.
- Grading criteria will be specified more in the task description.
- You will be graded for
  - Efficiency and effectiveness of your solution (think economically!)
  - Creativity in the solution approach.
  - Code quality & documentation quality (scientific standards!)
  - Value above AI tool solution (e.g. ChatGPT, perplexity.ai or similar)

Moodle
Please sign in to the Moodle space

- **https://moodle.oth-aw.de/course/view.php?id=3825**

- Via Moodle you will
  - Get relevant information on the lecture during the semester.
  - Get relevant organisational information.
  - Get materials for the lecture.
  - Get announcements.
  - Get the final exam, submit your solution for the exam, and schedule a review meeting.

QUESTIONS REGARDING ORGANIZATION?

PYTHON SETUP

Python Setup
Basics

OTH Amberg-Weiden

- Please create a virtual environment for Python
  - E.g. using the venv package: https://docs.python.org/3/library/venv.html
  - You can at anytime create a new env for new tasks or re-create your environment if it is broken
- Python version should be at least 3.8
- We will also work with Jupyter Notebooks since they integrate coding and presentation
  - https://jupyter.org
  - To work with your virtual env in Jupyter you need to create a kernel first
    python -m ipykernel install --user --name=<your venv name>
- Get access to the GPU lab computers (DC 1.07) → see description in Moodle.

Quellen:

Python Setup
Packages

OTH Amberg-Weiden

- You can work on the OTH computers or on your own laptop or even work on both.
  - In the latter case a version control like git is recommended, we will come to that point later.
- In your venv, for the beginning, please install the following packages
  - Numpy, pandas, matplotlib
  - Spacy, nltk
  - We will continuously add packages with new tasks
- You can at any time create a snapshot of your environment to reproduce it using
  - Pip freeze > requirements.txt
  - Reinstall it with pip install –r requirements.txt

Quellen:

Python Setup
Requirements.txt

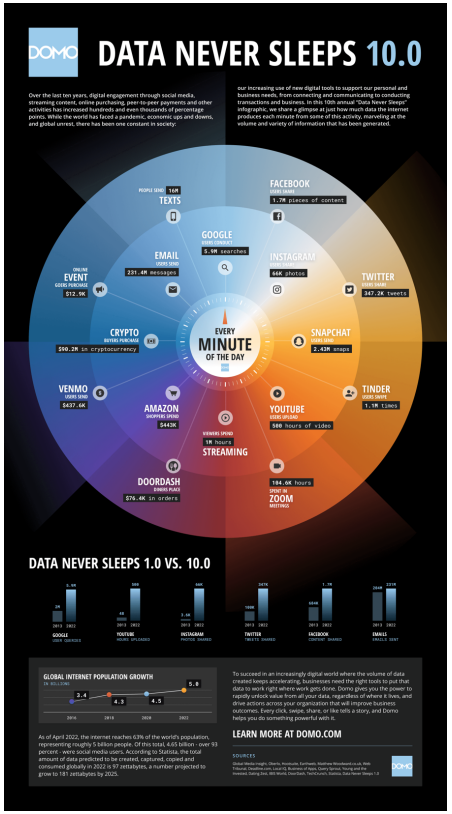
OTH Amberg-Weiden

- You can at any time create a snapshot of your environment to reproduce it using
  - Pip freeze > requirements.txt
  - Reinstall it with pip install –r requirements.txt
- If you work with version control
  - Regularly commit the requirements.txt file
  - Especailly before changes, it enables you to roll back to the last (working) environment

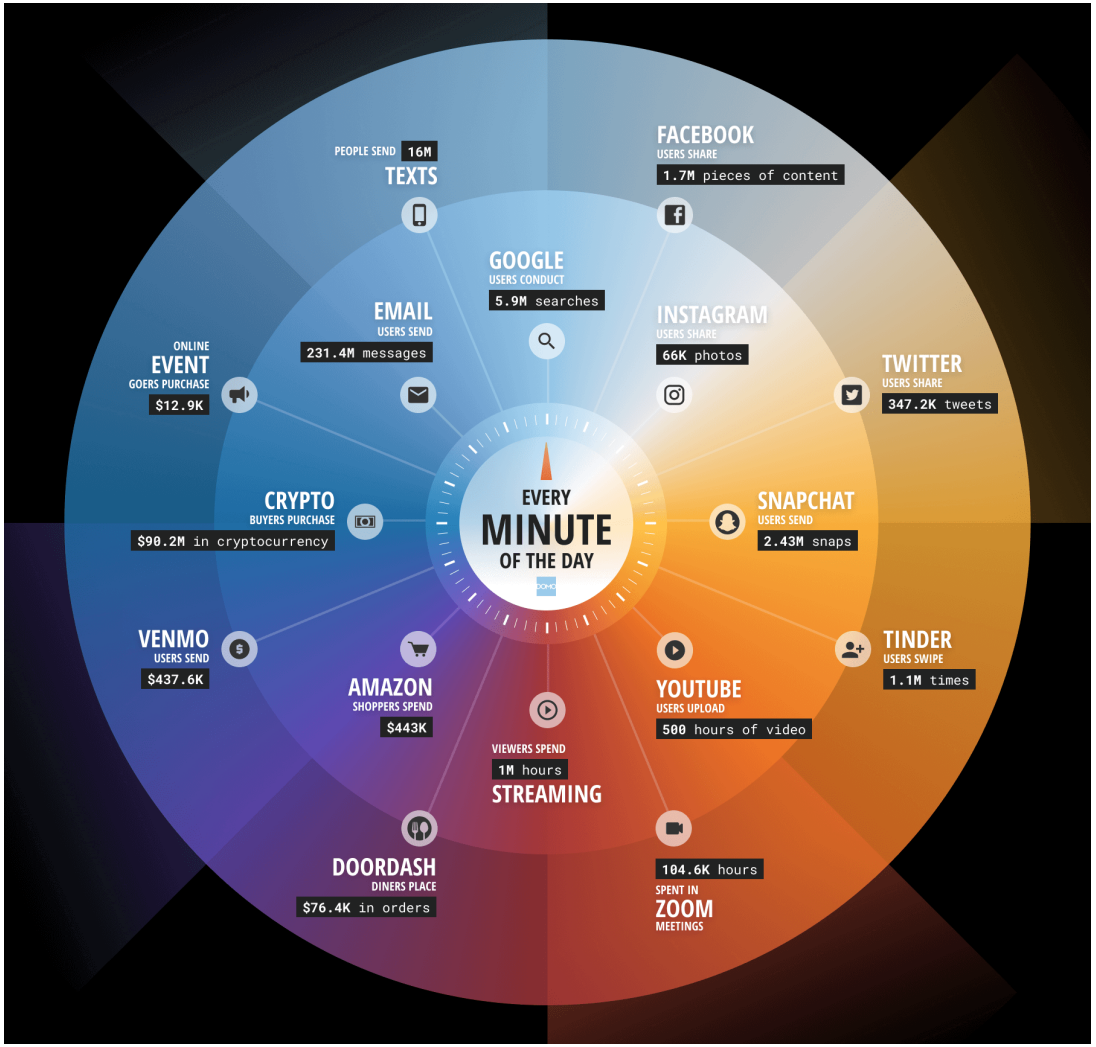
Quellen:

WHY NLP? AND WHAT IS IT ABOUT?

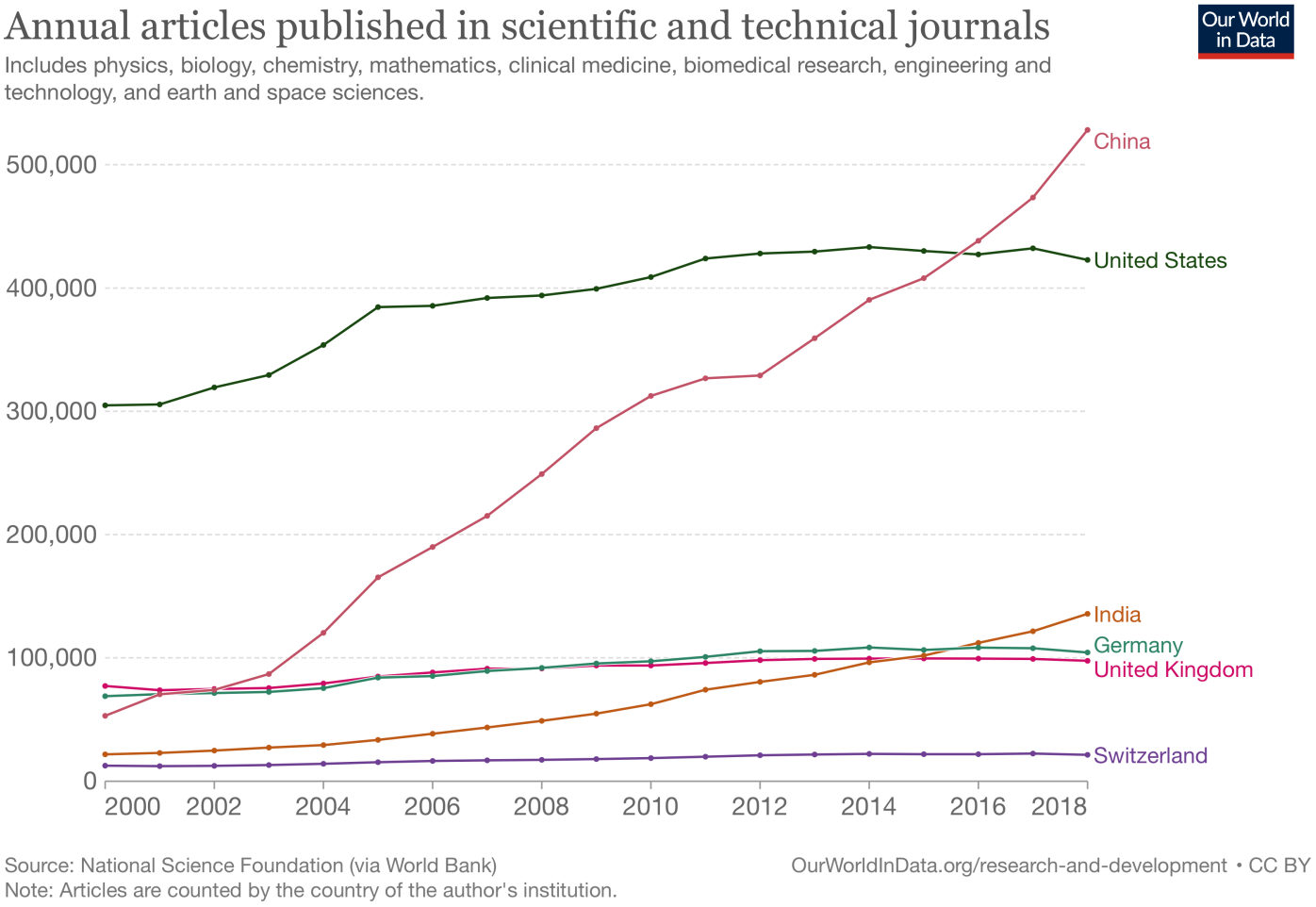
Data Grows and Grows...



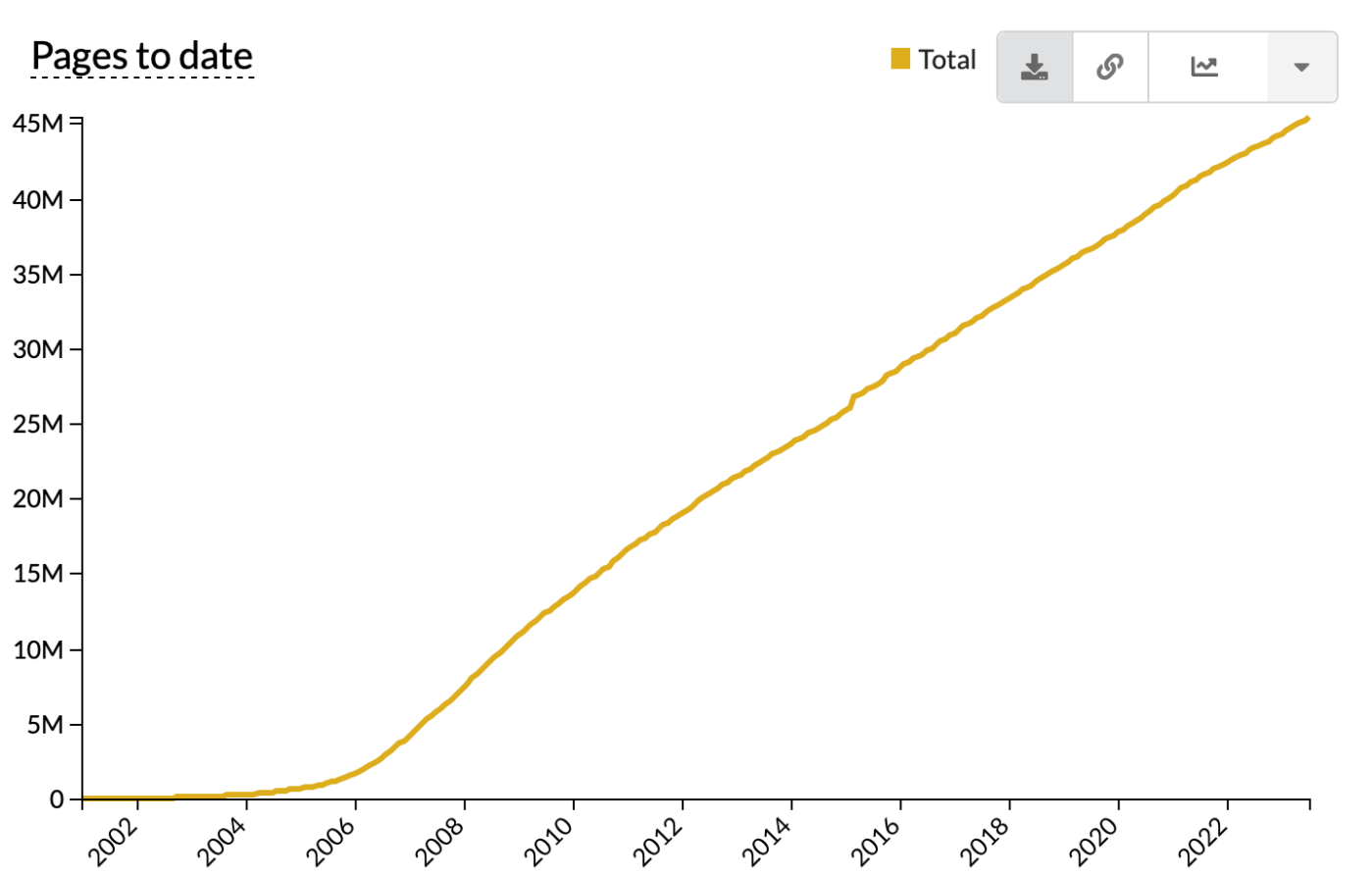
https://www.domo.com/data-never-sleeps



Research: More and more (complex) text...



Number of pages on wikipedia.org



From: stats.wikimedia.org, 04.09.2023

What is NLP?

Please name some NLP tasks…

What is NLP?

Please name some NLP tasks…

- Topic classification
- Sentiment analysis
- Summarization
- Translation
- Question Answering
- Related-content recommendation
- Text completion
- Text generation

Why NLP?

Please think of reasons why we do computer-based NLP

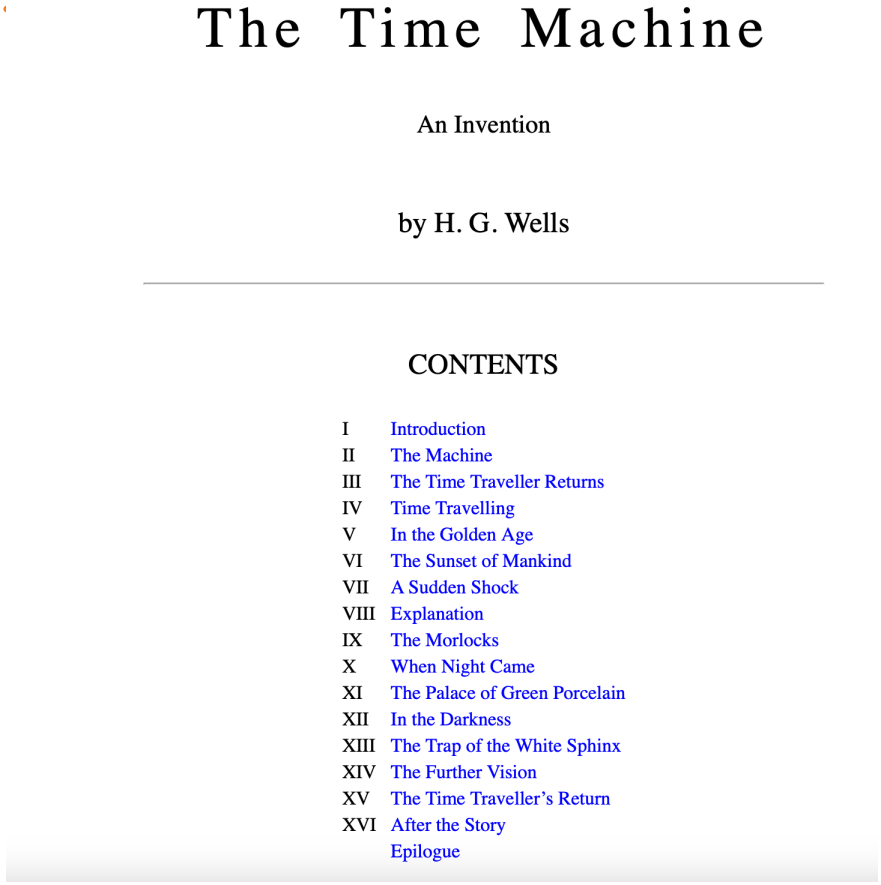
Why NLP?

Please think of reasons why we do computer-based NLP

- There is more text than we can ever read.
- Text-based tasks are boring.
- Text-based task are very time consuming.
- …

An Example Text…

OTH Amberg-Weiden



The Time Machine

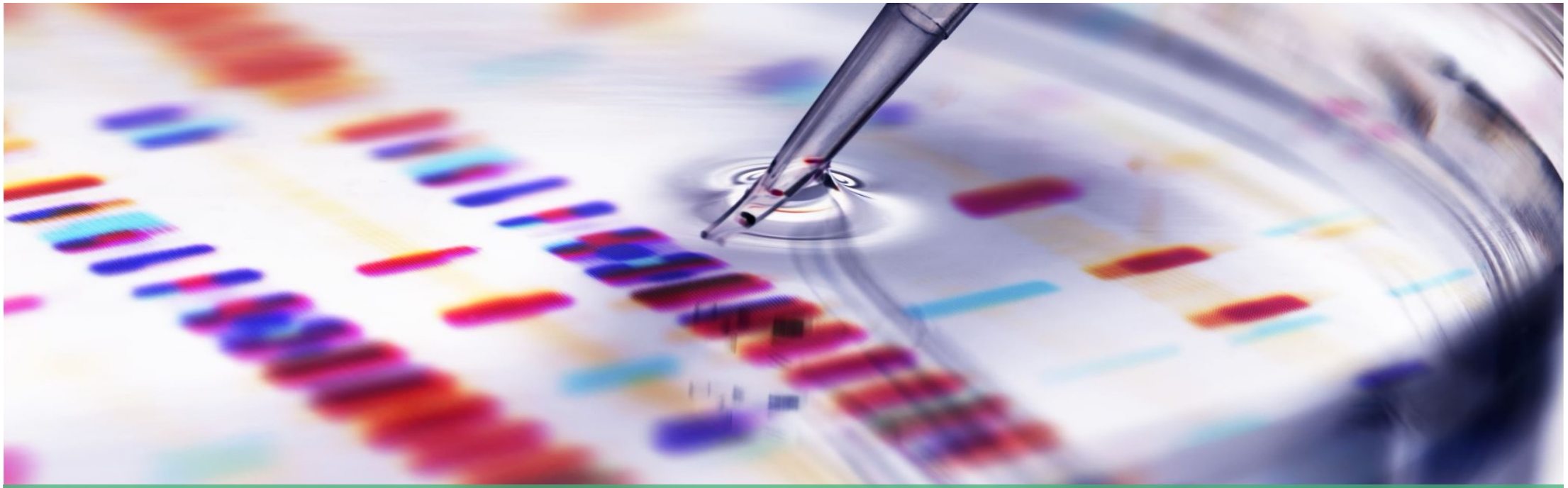
An Invention

by H. G. Wells

CONTENTS

https://www.gutenberg.org/cache/epub/35/pg35-images.html

- We will deal with this one some time…
- We will first see
  - How to retrieve text (from the internet)
  - How to do basic analysis of text, general linguistic features
- We will mostly work with English language
  - Exception: Translation
  - You will be able to generalze to other languages yourself (or you've found an interesting research topic)



INFORMATION RETRIEVAL

Major text sources for NLP applications

OTH Amberg-Weiden

- For some applications internal databases are possible sources, too.
  - Here usually common database technology is applied.
  - Databases for unstructured information.
    - Text is unstructured information, in contrast to structured information like data tables.
  - Usually, access is limited (authentication, authorization concepts in place)
  - Technological aspects are covered in a dedicated lecture about modern database technologies.
- Most text is nowadays published via the internet.
  - We focus on information retrieval from the internet.
- Text processing is the same, independent of the source (internet, database, OCR, …)

Information retrieval from the internet
Warnings

OTH Amberg-Weiden

- Published text is the intelectual property of the creator.
  - He can sell, license or grant his rights to other, e.g. a publisher, his employer, …, or even everybody
- However, any text is protected by intelectual property laws!
- For every source, you need to make sure that you may use the content at all.
  - If you are allowed to use it, you have to check the conditions!

Wikipedia example

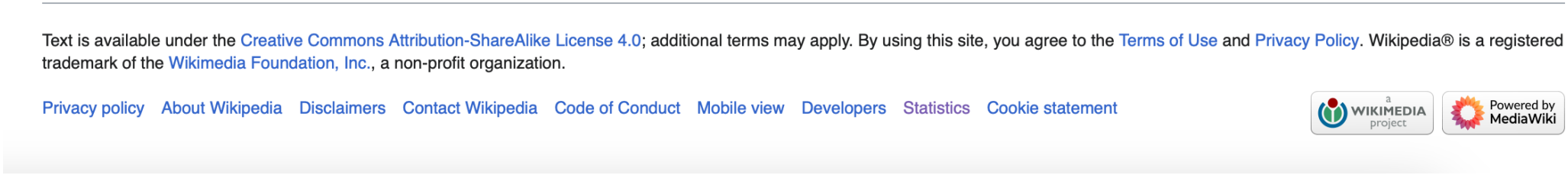
OTH Amberg-Weiden

- Check the wikipedia license.

Wikipedia example

OTH Amberg-Weiden

- https://en.wikipedia.org/wiki/Main_Page



- Creative Commons: https://creativecommons.org/about/cclicenses/
  - **CC BY-SA**: This license allows reusers to distribute, remix, adapt, and build upon the material in any medium or format, so long as attribution is given to the creator. The license allows for commercial use. If you remix, adapt, or build upon the material, you must license the modified material under identical terms.

CC BY-SA includes the following elements:
BY – Credit must be given to the creator
SA – Adaptations must be shared under the same terms

Downloading from the internet

OTH Amberg-Weiden

- We will explore some useful possibilities to get (text) data files from the web.
  - You will download the text for “The Time Machine” by H.G. Wells
  - You will download some Wikipedia stuff
  - We will need both later.
- Task 1:
  - Download the file https://www.gutenberg.org/files/35/35-0.txt using the Linux command wget
  - Store it as time_machine.txt
  - Download this paper: https://arxiv.org/pdf/1706.03762
    - We fully read this later. It is the seminal work all e.g. GPT models build upon! Feel free to also read it.

Downloading from the internet
Robots.txt

OTH Amberg-Weiden

- Before you automatically pull content from the internet, after clarification of the license issues, you need to check the robots.txt file.
- You will find it below the domain (domain/robots.txt), e.g. wikipedia.org/robots.txt
- It specifies what is allowed to be crawled from the website
- Basically it works with 3 keywords: User-agent, Disallow, Allow
- User-agent specifies to whom the following rules apply.
  - Values are either * (everybody) or specific agents (like google)
- Allow specifies which paths of the domain are allowed to be crawled (e.g. / = every, /news = subdirectory /news)
- Disallow consequently specifies which subdirectories are forbidden to be crawled.

Downloading from the internet
Task

- Task 2: Build a small crawler
  - There are tons of examples on the internet, however, try to solve it for yourself
    - Note, for code on the internet the same applies as for any other text, you need a license to use it.
  - Look up the documentation of the mentioned Python packages if you are not familiar with them.
  - Write a Python method crawl(url)
    - It shall send a GET request to url to retrieve the HTML content (use requests package)
    - It shall parse the HTML content and extract the text (use bs4 (beautiful soup) package).
    - Furthermore, it shall extract all links on the page and provide them as a list
    - Return values: text, links
    - -For testing: Crawl (for example) a single Wikipedia article you like.

Practical Advice
Using git

OTH Amberg-Weiden

- If you are not already familiar with it, familiarize with a version control system
- Keep the code you create in these tasks (the internet crawler and, next, the pdf parser)
- You may want to extend it during the course of the lecture
- You need it for furture tasks or maybe the project work.
- You can either use
  - git from OTH git.oth-aw.de
  - Your github
- In case you are not familiar with git: https://git-scm.com/book/en/v2

Portable Document Format
Introduction

OTH Amberg-Weiden

- PDF = Portable Document Formal
- Basically a format to set up a document for sharing and printing across platforms.
  - Standardized in ISO 32000 (first 2008, new version 2020)
  - To get an impression: https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/PDF32000_2008.pdf
- Often we need to deal with information stored in PDF format.
- PDF is difficult to parse. Several open source tools exist for Python:
  - PyPDF4
  - Pdfminer-six
  - MuPDF

check these packages

Parsing PDF
Task

OTH Amberg-Weiden

- For other langugages, own packages exist
  - E.g. PDFBox for Java
- Task: Write a pdf parser to extract the text from a PDF document
  - Ignore the images for now
  - Check the three packages out and select one
  - Check the licenses for each of the mentioned packages.
  - Parse the paper you downloaded before, extract the text and store it in a text file.

What is text?

OTH Amberg-Weiden

I.

Introduction

The Time Traveller (for so it will be convenient to speak of him) was expounding a recondite matter to us. His pale grey eyes shone and twinkled, and his usually pale face was flushed and animated. The fire burnt brightly, and the soft radiance of the incandescent lights in the lilies of silver caught the bubbles that flashed and passed in our glasses. Our chairs, being his patents, embraced and caressed us rather than submitted to be sat upon, and there was that luxurious after-dinner atmosphere, when thought runs gracefully free of the trammels of precision. And he put it to us in this way—marking the points with a lean forefinger—as we sat and lazily admired his earnestness over this new paradox (as we thought it) and his fecundity.

"You must follow me carefully. I shall have to controvert one or two ideas that are almost universally accepted. The geometry, for instance, they taught you at school is founded on a misconception."

"Is not that rather a large thing to expect us to begin upon?" said Filby, an argumentative person with red hair.

"I do not mean to ask you to accept anything without reasonable ground for it. You will soon admit as much as I need from you. You know of course that a mathematical line, a line of thickness *nil*, has no real existence. They taught you that? Neither has a mathematical plane. These things are mere abstractions."

"That is all right," said the Psychologist.

"Nor, having only length, breadth, and thickness, can a cube have a real existence."

"There I object," said Filby. "Of course a solid body may exist. All real things—"

"So most people think. But wait a moment. Can an *instantaneous* cube exist?"

"Don't follow you," said Filby.

"Can a cube that does not last for any time at all, have a real existence?"

Filby became pensive. "Clearly," the Time Traveller proceeded, "any real body must have extension in *four* directions: it must have Length, Breadth, Thickness, and—Duration. But through a natural infirmity of the flesh, which I will explain to you in a moment, we incline to overlook this fact. There are really four dimensions, three which we call the three planes of Space, and a fourth, Time. There is, however, a tendency to draw an unreal distinction between the former three dimensions and the latter, because it happens that our consciousness moves intermittently in one direction along the latter from the beginning to the end of our lives."

"That," said a very young man, making spasmodic efforts to relight his cigar over the lamp; "that . . . very clear indeed."

https://www.gutenberg.org/cache/epub/35/pg35-images.html

What is text?

OTH Amberg-Weiden

I.

Introduction

The Time Traveller (for so it will be convenient to speak of him) was expounding a recondite matter to us. His pale grey eyes shone and twinkled, and his usually pale face was flushed and animated. The fire burnt brightly, and the soft radiance of the incandescent lights in the lilies of silver caught the bubbles that flashed and passed in our glasses. Our chairs, being his patents, embraced and caressed us rather than submitted to be sat upon, and there was that luxurious after-dinner atmosphere, when thought runs gracefully free of the trammels of precision. And he put it to us in this way—marking the points with a lean forefinger—as we sat and lazily admired his earnestness over this new paradox (as we thought it) and his fecundity.

"You must follow me carefully. I shall have to controvert one or two ideas that are almost universally accepted. The geometry, for instance, they taught you at school is founded on a misconception."

"Is not that rather a large thing to expect us to begin upon?" said Filby, an argumentative person with red hair.

"I do not mean to ask you to accept anything without reasonable ground for it. You will soon admit as much as I need from you. You know of course that a mathematical line, a line of thickness *nil*, has no real existence. They taught you that? Neither has a mathematical plane. These things are mere abstractions."

"That is all right," said the Psychologist.

"Nor, having only length, breadth, and thickness, can a cube have a real existence."

"There I object," said Filby. "Of course a solid body may exist. All real things—"

"So most people think. But wait a moment. Can an *instantaneous* cube exist?"

"Don't follow you," said Filby.

"Can a cube that does not last for any time at all, have a real existence?"

Filby became pensive. "Clearly," the Time Traveller proceeded, "any real body must have extension in *four* directions: it must have Length, Breadth, Thickness, and—Duration. But through a natural infirmity of the flesh, which I will explain to you in a moment, we incline to overlook this fact. There are really four dimensions, three which we call the three planes of Space, and a fourth, Time. There is, however, a tendency to draw an unreal distinction between the former three dimensions and the latter, because it happens that our consciousness moves intermittently in one direction along the latter from the beginning to the end of our lives."

"That," said a very young man, making spasmodic efforts to relight his cigar over the lamp; "that . . . very clear indeed."

https://www.gutenberg.org/cache/epub/35/pg35-images.html

- **Sequence**
  - of chars and punctuations
  - of words and punctiations
- **Organized** in words, sentences, paragraphs
- Has certain **patterns**.
- Datatype **string**.

STRINGS

Basic Datatype for Text

Strings
Every Text is a String

OTH Amberg-Weiden

- In Python, like in other programming languages, a special datatype exists for text: string
- A string is defined by single or double quotation marks
  - str = “my text”
  - str = ‘my text’
- Be careful with apostrophes or quotations within the string:
  - str = “She’s had a good time singing ’Dancing Queen’ aloud.”

Strings in Python
Access by Index

OTH Amberg-Weiden

- You can access the chars in the string by an index:
  - str = “name”
  - str[0] = “n”, str[1] = “a”, str[2] = “m”, str[3] = “e”
- This way you can also slice your text
  - str[0:2] = “na”
- However, strings are not lists, you cannot do the following
  - [“hat”, “is”, “your”] + “name”
  - This will lead to a “TypeError: can only concatenate list (not "str") to list”

Important Python String Operations

OTH Amberg-Weiden

Method	Functionality
`s.find(t)`	index of first instance of string `t` inside `s` (`-1` if not found)
`s.rfind(t)`	index of last instance of string `t` inside `s` (`-1` if not found)
`s.index(t)`	like `s.find(t)` except it raises `ValueError` if not found
`s.rindex(t)`	like `s.rfind(t)` except it raises `ValueError` if not found
`s.join(text)`	combine the words of the text into a string using `s` as the glue
`s.split(t)`	split `s` into a list wherever a `t` is found (whitespace by default)
`s.splitlines()`	split `s` into a list of strings, one per line
`s.lower()`	a lowercased version of the string `s`
`s.upper()`	an uppercased version of the string `s`
`s.title()`	a titlecased version of the string `s`
`s.strip()`	a copy of `s` without leading or trailing whitespace
`s.replace(t, u)`	replace instances of `t` with `u` inside `s`

Table from Bird et. al., NLTK Book, Chapter 3

Practical Part

OTH Amberg-Weiden

- See Notebook NLP01
  - Section “String Operations”

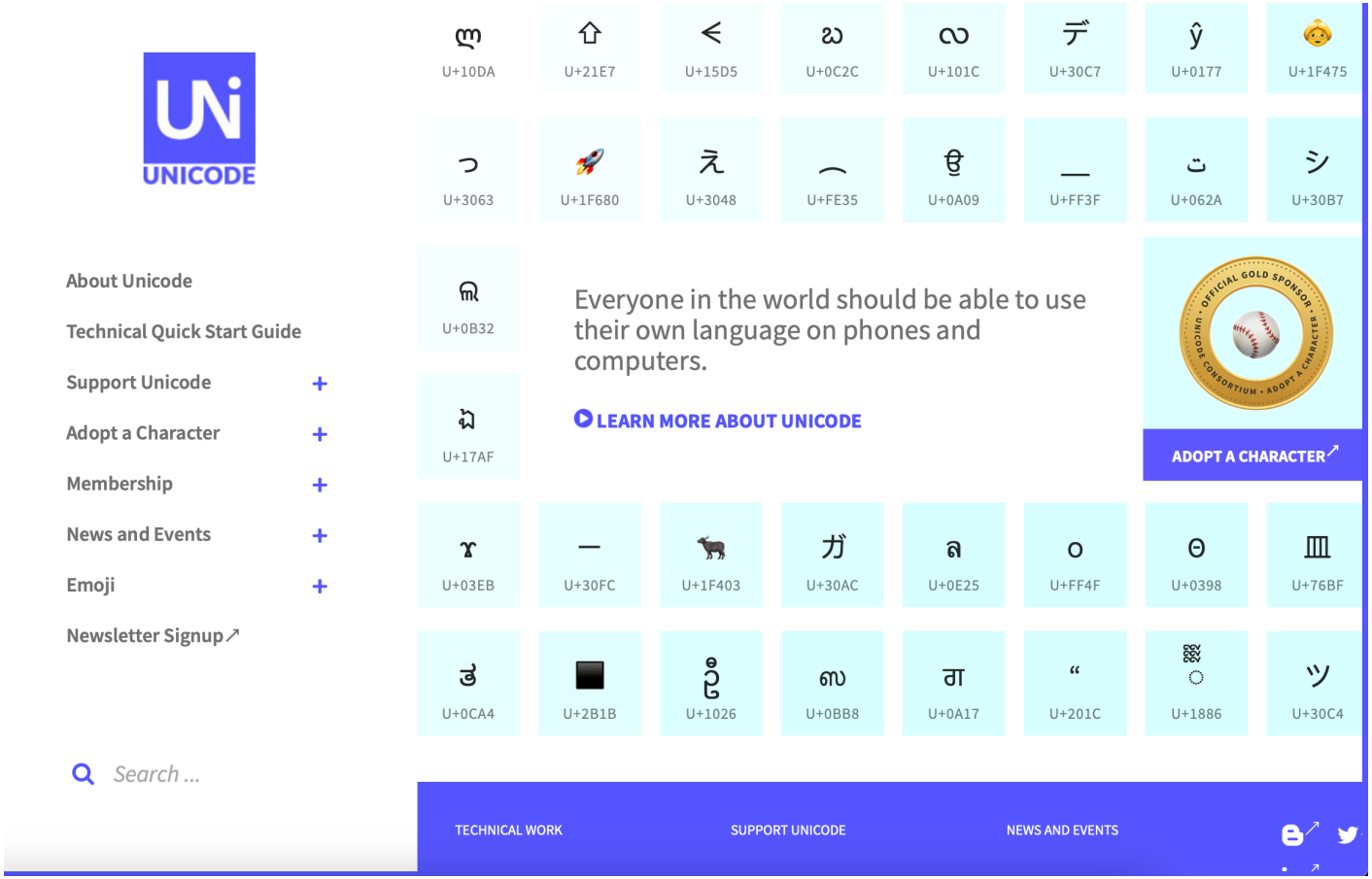
Encodings
Representation of Character Sets

OTH Amberg-Weiden

- In order to work with text on a computer, you have to encode its elements to bytes.
- Text consists of sentences, sentences of words, words consist of characters.
  - Encoding works at the character level
- Each character is encoded usually as a code point, which is some number
- Abstraction from the graphical representation on screen
- Unicode standard
  - www.unicode.com

Unicode examples

- See unicode.com



www.unicode.com

ASCII

OTH Amberg-Weiden

- American Standard Code for Information Interchange
- Early format, therefore limited.
- 128 code points
  - Only Latin letters, numbers, punctuation, some special characters

Unicode in Python

OTH Amberg-Weiden

- Unicode representations: numbers between 0 through 0x10FFFF (hexadecimal)
- Encoding: representation of the number code in bytes
- UTF-8: “Unicode Transformation Format” with 8-bit values
  - Default in Python
  - Number representation < 128: corresponding byte value.
  - Number representation >= 128: sequence of 2, 3, or 4 bytes, each byte between 128 and 255
- Compact representation
- Portable between processor architectures

https://docs.python.org/3/howto/unicode.html

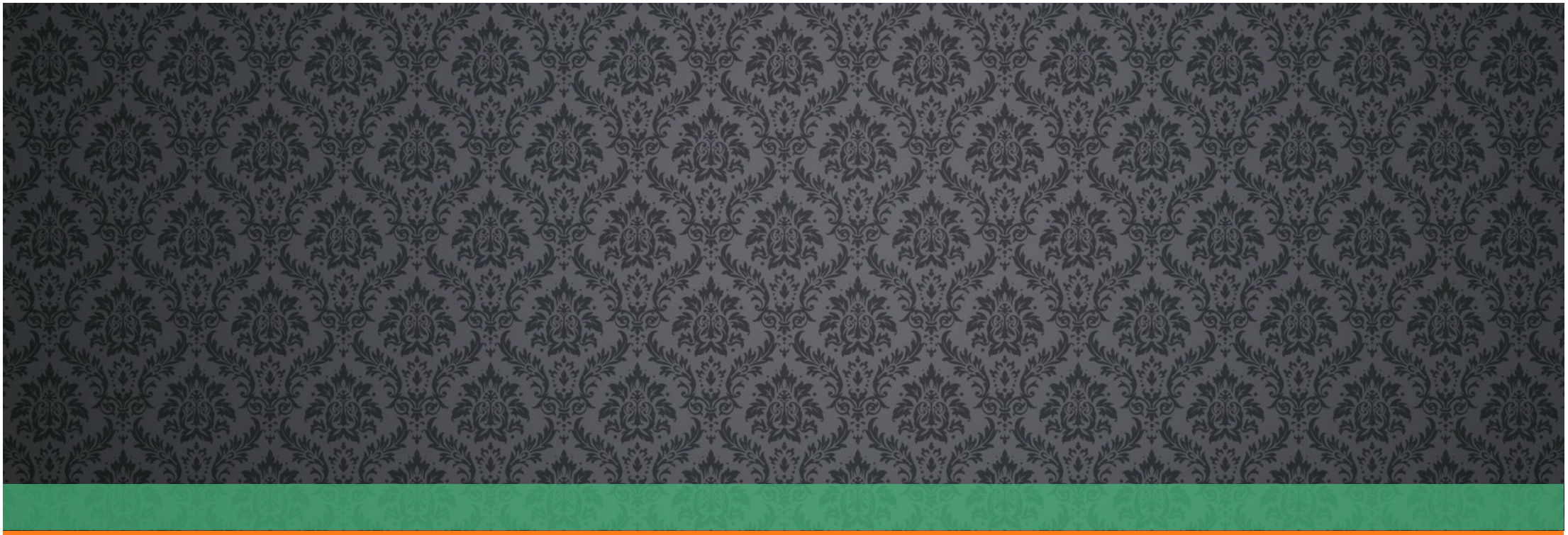
Practical Aspects

OTH Amberg-Weiden

- Often, ambiguous encoding is a source of error in computer programs.
- Ensure a proper UTF-8 representation of your text, e.g. by using a decode method with a “strict” decoding (default).

- See notebook NLP01, section ”Encodings” for some small exercises.

- Further reading (this was only an introduction for awareness)
  - https://docs.python.org/3/howto/unicode.html, and references therein, e.g.
  - https://python-notes.curiousefficiency.org/en/latest/python3/text_file_processing.html



PATTERNS

Regular Expressions

Regular Expressions

OTH Amberg-Weiden

- Reading some random parts of The Time Machine you will realize that time comes written as ”time” or as “Time” (e.g. in “the Time Traveller”).
- How do we search all ocurrences of “time” in the text?
- How do we distinguish it from words like “times”, which containst “time”?
- Can we verify whether and which numbers occur in the text?

Regular Expressions
Syntax

OTH Amberg-Weiden

- Regular expression (short regex) allow you to specify character patterns to be searched for in text (files)
- They work in pure linux but can be applied in Python using e.g. the native re package.

- How do Regex work?
- Special characters (meta characters): . ^ $ * + ? { } [ ] \ | ( )
  - They serve the regex syntax.
  - If you need to search one of these characters in your text you need to escape it (we will see that later).

Regular Expressions
Syntax

OTH Amberg-Weiden

check it out

- [] summarize characters to classes:
  - [xyz] is any of the characters “x”, “y”, or “z”
  - ”-” can be used to formulate “from…to”, so [xyz] is equivalent to [x-z]
  - Popular classes: [a-z]: lower case letters, [A-Z]: upper case letters, [0-9]: single digits
  - Most meta characters are inactive wihtin classes ([$?] finds every “$”, or “?”)
- | is the or operator:
  - the|The: find the word “the” independent of whether it is written with a capital or lower case t.
  - Rather used for synonymes: book|manuscript
  - Character level or expressions rather expressed as classes: a|b|c = [a-c], the|The = [tT]he

Regular Expressions
Syntax

OTH Amberg-Weiden

- ^ serves as not operator *if it is the first symbol.* It negates the whole class within [] :
  - [^4] will match everything except 4
  - [^A-Z] will match anything that is not an uppercase letter
  - [t^] will find a “t” or a caret (“^”)!

Regular Expressions
Syntax

OTH Amberg-Weiden

- * is used to express 0 or more occurrences of the previous character:
  - ab*c matches ac, abc, abbc, abbbc, abbbbc, …
- + is used to express 1 or more occurrences of the previous character:
  - ab+c matches abc, abbc, abbbc, … it does not match ac
- ? is used to express 0 or 1 occurrences of the previous character:
  - ab?c matches ac and abc, it does not match abbc, abbbc, …
- . replaces any character
  - a.c matches ac, aac, abc, acc, adc, a3c, …

Regular Expressions
Syntax

OTH Amberg-Weiden

- The characters ^, $ are used as anchors:
- ^ outside a class marks the beginning of a string:
  - ^[a-z] matches “alphabet” (starts with a lower case letter)
  - ^[A-Z] does not match “alphabet” (does not start with an uppercase letter)
- $ marks the end of a string:
  - y$: String ends with a y: “theory”, “melody”, “entropy”, …
  - E.g. [a-z]$: matches everything that ends with a lower case letter.

Regular Expressions
Syntax

OTH Amberg-Weiden

- How do I find a string that ends with a fullstop (“.”)
  - Is the pattern .$ going to work?
- To use a special character as search pattern we have to escape it using “\”:
  - “.” = \.
  - “^” = \^
  - “\” = “\\”

Regular Expressions
Syntax

OTH Amberg-Weiden

- The following abbreviations exist
  - \d: decimal digit, equivalent to [0-9]
  - \D: non-digit character, equivalent to [^0-9]
  - \s: whitespace, equivalent to [\t\n\r\f\v]
  - \S: non-whitespace, equivalent to [^ \t\n\r\f\v]
  - \w: alphanumeric character, equivalent to [a-zA-Z0-9_]
  - \W: non-alphanumeric character, equivalent to [^a-zA-Z0-9_]

Tasks
Notebook

- (1) See notebook NLP01
  - Section “Regular Expressions”

Tasks
Extend Your Crawler

OTH Amberg-Weiden

- (2) Extend your Python method crawl(url)
  - You have realized that some links lead to other wiki pages, while others are just meta information.
  - A good link pattern to follow for Wikipedia is any link that starts with “/wiki”. However, this pattern is only valid for Wikipedia.
  - Now, the crawler shall additionally retrieve the text of the first n_links that follow a certain pattern.
  - The pattern is provided as argument to keep the crawler generally useful.
  - Do not follow links on the linked page, otherwise your crawler might pull lots of pages.
  - New signature: crawl_with_links(url, link_pattern, n_links=20)
  - Return the main page text, plus a list of texts from the links

Tasks
Extend Your Crawler

OTH Amberg-Weiden

- Hints for the crawler
  - Check how the URL looks
  - Check how your links are provided in the page
  - Print out every link you follow

Literature
Zweite Headline

OTH Amberg-Weiden

- Daniel Jurafsky, James H. Martin , Speech and Language Processing, Copyright © 2023. All rights reserved. Draft of January 7, 2023.
  Online: https://web.stanford.edu/~jurafsky/slp3/ed3book_jan72023.pdf
- Steven Bird, Ewan Klein, and Edward Loper, Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit, O-Reilly, 3rd edition (2019), Online Version https://www.nltk.org/book/
- https://docs.python.org/3/howto/regex.html#regex-howto

Quellen: