TOKENISIERUNG

Von Ludmilla Borisenkov

WAS IST TOKENISIERUNG?

Der Prozess der Segmentierung von Fließtext in Wörter und Sätze

„xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx…“

[XXX, XXXXXXX, X, XXXX, X, XX, XXXX, XXX …]

Was ist Tokenisier-ung?

Ein Token ist

„jedes einzelne Vorkommen eines Wortes (oder einer anderen Einheit) in einem Text “

Bsp.: ‚hallo‘, ‚Los Angeles‘, ‚10.11.2017‘

Definition

‚Token‘

Vereinfachte Sprachverarbeitung

„XXXXX XXXX XXXXXXXX, XX XXXXXXX. XXXXX, XX XXXX XXX, XXXXX XXXX XX.“

(Aufteilung nach Vorkommen von Punkten)

Einteilung in Sätze:

[„XXXXX XXXX XXXXXXXX, XX XXXXXXX.“, „XXXXX, XX XXXX XXX, XXXXX XXXX XX.“]

(Aufteilung nach Leerzeichen)

Einteilung in Tokens:

[„XXXXX“, „XXXX“, „XXXXXXXX“, „,“, „XX“, „XXXXXXX“, „.“, „XXXXX“, „,“, „XX“, „XXXX“, „XXX“, „,“, „XXXXX“, „XXXX“, „XX“, „.“]

AUFKOMMENDE PROBLEME

Keine Leerzeichen

Nicht für jede Sprache geeignet, bsp. Japanisch

富士山（ふじさん、英語：Mount Fuji）は、静岡県（富士宮市、裾野市、富士市、御殿場市、駿東郡小山町）と、山梨県（富士吉田市、南都留郡鳴沢村）に跨る活火山である。標高3776.24 m[4][5]、日本最高峰（剣ヶ峰）[注釈 1]の独立峰で、その優美な風貌は日本国外でも日本の象徴として広く知られている。数多くの芸術作品の題材とされ芸術面で大きな影響を与えただけではなく、気候や地層など地質学的にも大きな影響を与えている。懸垂曲線の山容を有した玄武岩質成層火山で構成され、その山体は駿河湾の海岸まで及ぶ。

„Unsaubere“ Texte

- Rechtschreibfehler
- Unerwartete Zeichen
  - (Chinesisches Schriftzeichen,
  - Mathematische Symbole
  - Smileys …)
- Geteilte Wörter
  - (Sammel- (neue Zeile) band)

Abkürzungen & Kollokationen

- Abkürzungen
  - Beispiele
    - Dr. Who
    - Karl-Wilhelm Str.
  - Keine universell akzeptierte Standards für viele Abkürzungen
    -> Lexikon zum Nachschlagen

- Kollokation
  - Beispiele
    - Los Angeles
    - rock 'n' roll

Numerische oder Spezialbegriffe

- Email-Adressen
- URLs
- Komplexe Aufzählungen von Dingen
- Telefon - Nummern
- Datum
- Zeit
- Maße
- Zitate
- etc.

LÖSUNG

NLTK – Natural Language Toolkit

```
lumia@ubuntu4lumia:~$ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> sentence = 'I am an example sentence which shows you how I am tokenized by n
ltk.'
>>> tokens = nltk.word_tokenize(sentence)
>>> tokens
['I', 'am', 'an', 'example', 'sentence', 'which', 'shows', 'you', 'how', 'I', 'a
m', 'tokenized', 'by', 'nltk', '.']
>>>
```

Entwickler von NLTK

Edward Loper, Ewan Klein, and Steven Bird, Stanford, July 2007

ÜBUNG

NLTK-Tokenizer für Deutsch

„Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen wie “Hauptstr. 3” und mit dem Ausdruck 4.9.2008 (oder auch 4. 9. 2008) einen Datumsausdruck. In den Jahren 1999 und 2000 hat es 1999 Liter geregnet. “

Aufgabe:
Tokenisiert den oberen Beispieltext mit Hilfe von NLTK und regulären Ausdrücken, sodass er wie unten aussieht.

[‚Das‘, ‚hier‘, ‚ist‘, ‚tatsächlich‘, ‚ein‘, ‚Mini-Testtext‘, ‚.‘, ‚Er‘, ‚testet‘, ‚u.a.‘, ‚Abkürzungen‘, ‚wie‘, ‚Hauptstr.‘, ‚3‘, ‚und‘, ‚mit‘, ‚dem‘, ‚Ausdruck‘, ‚4.9.2008‘, ‚(‘, ‚oder‘, ‚auch‘, ‚4.‘, ‚9.‘, ‚2008‘, ‚)‘, ‚einen‘, ‚Datumsausdruck‘, ‚.‘, ‚In‘, ‚den‘, ‚Jahren‘, ‚1999‘, ‚und‘, ‚2000‘, ‚hat‘, ‚es‘, ‚1999‘, ‚Liter‘, ‚geregnet‘, ‚.‘]

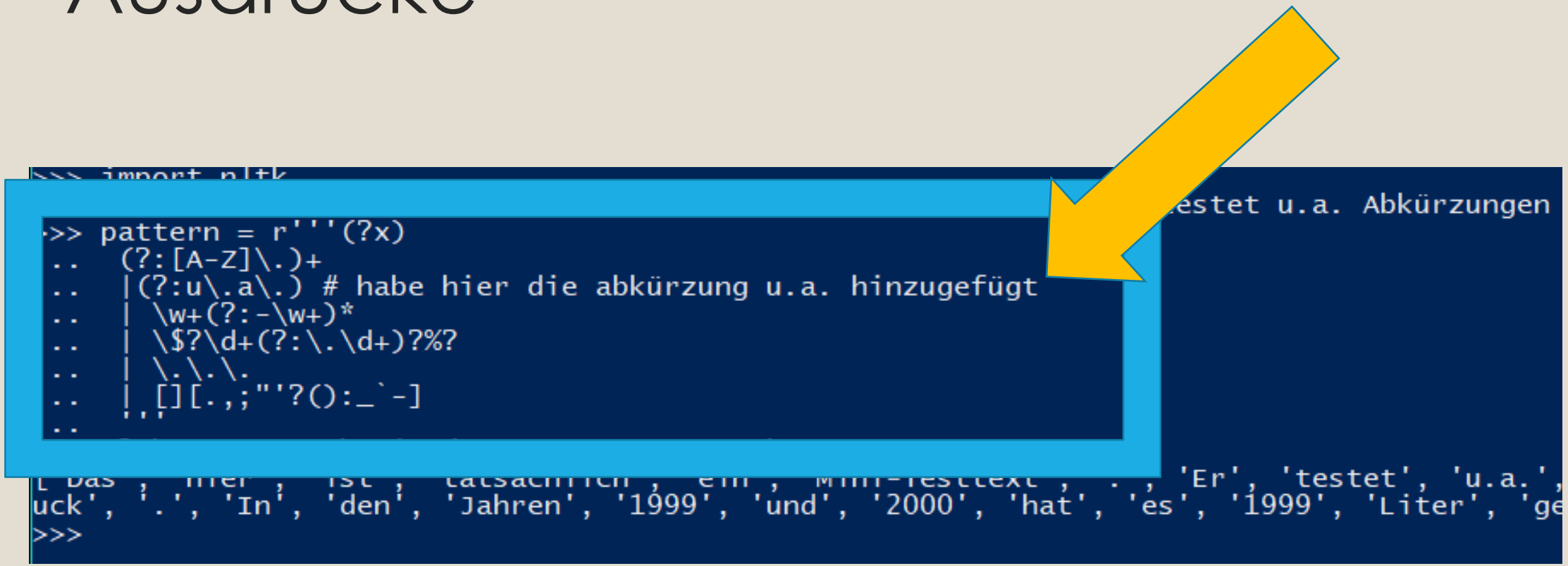
Vorgehensweise

```
>>> import nltk
>>> sentences = """Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen
>>> pattern = r'''(?x)
...   (?:[A-Z]\.)+
...   |(?:u\.a\.) # habe hier die abkürzung u.a. hinzugefügt
...   | \w+(?:-\w+)*
...   | \$?\d+(?:\.\d+)?%?
...   | \.\.\.
...   | [][.,;"'?():_`-]
...   '''
...
>>> nltk.regexp_tokenize(sentences, pattern)
['Das', 'hier', 'ist', 'tatsächlich', 'ein', 'Mini-Testtext', '.', 'Er', 'testet', 'u.a.',
uck', '.', 'In', 'den', 'Jahren', '1999', 'und', '2000', 'hat', 'es', '1999', 'Liter', 'ge
>>>
```

Vorgehensweise – Modul importieren

```
>>> import nltk
>>> sentences = """Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen
>>> pattern = r'''(?x)
...   (?:[A-Z]\.)+
...   |(?:u\.a\.) # habe hier die abkürzung u.a. hinzugefügt
...   | \w+(?:-\w+)*
...   | \$?\d+(?:\.\d+)?%?
...   | \.\.\.
...   | [][.,;"'?():_`-]
...   '''
>>> nltk.regexp_tokenize(sentences, pattern)
['Das', 'hier', 'ist', 'tatsächlich', 'ein', 'Mini-Testtext', '.', 'Er', 'testet', 'u.a.',
uck', '.', 'In', 'den', 'Jahren', '1999', 'und', '2000', 'hat', 'es', '1999', 'Liter', 'ge
>>>
```

Vorgehensweise – Reguläre Ausdrücke



Vorgehensweise - Tokenisierung

```
>>> import nltk
>>> sentences = """Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen
>>> pattern = r'''(?x)
...   (?:[A-Z]\.)+
...   |(?:u\.a\.) # habe hier die abkürzung u.a. hinzugefügt
...   | \w+(?:-\w+)*
...   | \$?\d+(?:\.\d+)?%?
...   | \.\.\.
...   | [][.,;"'?():-_`]
...
>>> nltk.regexp_tokenize(sentences, pattern)
['Das', 'hier', 'ist', 'tatsächlich', 'ein', 'Mini-Testtext', '.', 'Er', 'testet', 'u.a.',
uck', '.', 'In', 'den', 'Jahren', '1999', 'und', '2000', 'hat', 'es', '1999', 'Liter', 'ge
>>>
```

Nicht – Programmierer/ Programmieranfänger

- Veränder den Beispieltext und sucht wo NLTK noch alles Fehler macht
- Gefundene Fehler an die Tafel schreiben

(‚wie es aussehen sollte' -> ‚wie NLTK es tokenisiert')

```
>>> nltk.word_tokenize(sentences)
['Das', 'hier', 'ist', 'tatsächlich', 'ein', 'Mini-Testtext', '.', 'Er', 'testet', 'u.a', '.', '
k', '.', 'In', 'den', 'Jahren', '1999', 'und', '2000', 'hat', 'es', '1999', 'Liter', 'geeignet',
>>>
```

RegEx - CheatSheet

Operator	Behavior
`.`	Wildcard, matches any character
`^abc`	Matches some pattern *abc* at the start of a string
`abc$`	Matches some pattern *abc* at the end of a string
`[abc]`	Matches one of a set of characters
`[A-Z0-9]`	Matches one of a range of characters
`ed|ing|s`	Matches one of the specified strings (disjunction)
`*`	Zero or more of previous item, e.g. `a*`, `[a-z]*` (also known as *Kleene Closure*)
`+`	One or more of previous item, e.g. `a+`, `[a-z]+`
`?`	Zero or one of the previous item (i.e. optional), e.g. `a?`, `[a-z]?`
`{n}`	Exactly *n* repeats where n is a non-negative integer
`{n,}`	At least *n* repeats
`{,n}`	No more than *n* repeats
`{m,n}`	At least *m* and no more than *n* repeats
`a(b|c)+`	Parentheses that indicate the scope of the operators

Abkürzungen im Deutschen

str.	Str.	...	a.	a.D.	a.M.	Abs.	al.	Anm.	apl.
Art.	Az.	Bek.	Bros.	bzw.	ca.	Chr.	Co.	Corp.	d.
d.h.	Dr.	Dr.med.		Dr.phil.	Dr.rer.nat.		e.V.	engl.	etc.
ff.	Fr.	geb.	ggf.	hrsg.	i.A.	Inc.	incl.	inkl.	Kap.
Ltd.	med.	min.	Min.	Mio.	Mr.	Mrd.	nat.	Nr.	phil.
Prof.	Red.	rer.	rer.nat.	s. S.	Sal.	sog.	St.	Std.	Str.
Tel.	u.	u.a.	u.s.f.	u.s.w.	u.v.m.	usw.	v.	v.a.	v.Chr.
vs.	Vol.	z.	z.B.	z.T.	z.Hd.	z.Z.	z.Zt.	Zt.	

ZWISCHEN - BESPRECHUNG

Text in Sätze

„Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen wie “Hauptstr. 3” und mit dem Ausdruck 4.9.2008 (oder auch 4. 9. 2008) einen Datumsausdruck. In den Jahren 1999 und 2000 hat es 1999 Liter geregnet. “

Aufgabe:
Tokenisiert den oberen Beispieltext mit Hilfe von NLTK und regulären Ausdrücken sodass er wie unten aussieht.

[‚Das hier ist tatsächlich ein Mini-Testtext.‘,
‚Er testet u.a. Abkürzungen wie Hauptstr. 3 und mit dem Ausdruck 4.9.2008 oder auch 4. 9. 2008 einen Datumsausdruck.‘,
‚In den Jahren 1999 und 2000 hat es 1999 Liter geregnet.‘]

Text in Sätze

–

Was fällt euch auf?

```
>>> import nltk
>>> sentences = """Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a. Abkürzungen
>>> pattern = r'''(?x)
...   (?:[A-Z]\.)+
...   |(?:u\.a\.) # habe hier die abkürzung u.a. hinzugefügt
...   | \w+(?:-\w+)*
...   | \$?\d+(?:\.\d+)?%?
...   | \.\.\.
...   | [][.,;"'?():_`-]
...   '''
...
>>> nltk.regexp_tokenize(sentences, pattern)
['Das', 'hier', 'ist', 'tatsächlich', 'ein', 'Mini-Testtext', '.', 'Er', 'testet', 'u.a.',
uck', '.', 'In', 'den', 'Jahren', '1999', 'und', '2000', 'hat', 'es', '1999', 'Liter', 'ge
>>>
```

ERWÄHNENSWERTES

Allgemein:

```
$ perl tokenize.perl [OPTIONS] <fileIn.text> <fileOut.tok>
```

Beispiel:

```
Das hier ist tatsächlich ein Mini-Testtext. Er testet u.a.
Abkürzungen wie “Hauptstr. 3” und mit dem Ausdruck 4.9.2008 (oder
auch 4. 9. 2008) einen Datumsausdruck. In den Jahren 1999 und 2000
hat es 1999 Liter geregnet.
```

```
~$ perl tokenize.perl -abbrev abbrev.lex test.txt output.txt
```

```
Das hier ist tatsächlich ein Mini-Testtext .
Er testet u.a. Abkürzungen wie “Hauptstr. 3” und mit dem Ausdruck
4.9.2008 ( oder auch 4. 9. 2008 ) einen Datumsausdruck .
In den Jahren 1999 und 2000 hat es 1999 Liter geregnet .
```

Tokenizer for German

-

Stefanie Dipper

https://www.linguistics.ruhr-uni-bochum.de/~dipper/resources/tokenizer.html

Stanford Tokenizer

Sehr schnelle Verarbeitung von Texten

- hervorragend für große Korpora
- Ca. 1 Million Tokens pro Sekunde
- Für Englisch ausgelegt

https://nlp.stanford.edu/software/tokenizer.shtml

FRAGEN?

Besuchenswerte Seiten

- NLTK Lehrbuch

- Wie installiere ich … auf meinen Windows Computer?
  - … Java
  - … Git
  - … Stanford NLP tools