Text-Denoising mit Go(lang)

Bedcon 2015 – Dennis Kluge – Charité Berlin

DATAFLEX

VISION THEMEN KONTAKT

FORSCHUNG DURCH VERNETZUNG

Das Forschungprojekt DataFlex hat zum Ziel, anonyme medizinische Daten miteinander zu vernetzen, auszuwerten und adäquat zu vermitteln. Dank **innovativer technischer Verfahren** ist der Datenschutz beim Zugriff auf die enorme Datenmenge garantiert.

Kontakt aufnehmen

dataflex-science.de

THEMEN

Validierung eines **neuartigen** Datenmanagements

Innerhalb des Forschungsprojekts OpEN.SC der Forschungsgruppe Digitale Pathologie wurde in der Charité-Universitätsmedizin Berlin ein neuartiges Datenmanagement entwickelt. Es ermöglicht, ein Netzwerk von Daten aufzubauen ohne bestimmte Profilinformationen oder personenbezogene Daten zu speichern. Das

NLP

Natural Language Processing

Die **@bedcon** ist die großartigste **#Konferenz** des Jahres. 😍

http://bedcon.org

Die bedcon ist die großartigste Konferenz des Jahres

die bedcon ist die großartigste konferenz des jahres

[“di”, “ie”, “e_”, “_b”, “be”, “ed”…]


1
2
3
4
5

GO(LANG)

- 2009 - erschienen … 2012 - Version 1.0
- kompiliert, stark typisiert, imperativ, strukturiert
- optimiert für Nebenläufigkeit
- Garbage Collection
- C angelehnte Syntax

STRINGS

- Go source code is always UTF-8.
- A string holds arbitrary bytes.
- A string literal, absent byte-level escapes, always holds valid UTF-8 sequences.
- Those sequences represent Unicode code points, called runes.
- No guarantee is made in Go that characters in strings are normalized.
- https://blog.golang.org/strings

```
1  var awesomeTweet string = "Die @bedcon ist die großartigste
2                             #Konferenz des Jahres. 😍
3  http://bedcon.org"
```

```
1  var awesomeTweet string = "Die @bedcon ist die großartigste
2                             #Konferenz des Jahres. 😍
3                            http://bedcon.org"
```

Boolean, Numeric,
String, Array, Slice,
Map, Interface, Map, Channel

```
1   awesomeTweet := "Die @bedcon ist die großartigste
2                   #Konferenz des Jahres. 😍
3                   http://bedcon.org"
```

```
1  awesomeTweet := "Die @bedcon ist die großartigste
2                  #Konferenz des Jahres. 😍
3                  http://bedcon.org"
```

Type Inference

```
type tweet struct {
  text string
  metadata map[string]string
}

tweet := &tweet{text: "foobar",
                metadata: map[string]string{"location": "berlin"}}
```

```
1  lowerCasedTweet := strings.ToLower(awesomeTweet)
```

```
1  lowerCasedTweet := strings.ToLower(awesomeTweet)
```

Package

```
1   package strings
2
3   func ToLower(text string) {...}
```

```
1   package strings
2
3   func ToLower(text string) {...}
```

Großbuchstabe deklariert public

```
1   fmt.Printf("Lower Case: %s \n", lowerCasedTweet)
```

```
1   fmt.Printf("Lower Case: %s \n", lowerCasedTweet)
```

C-Style

```
1  tweetWithoutHashtags := strings.Replace(lowerCasedTweet, "#", "", -1)
```

```
1  urlRegexp := regexp.MustCompile(
2    `((([A-Za-z]{3,9}:(?:\/\/)?)(?:[-;:&=\+\$,\w]+@)?
3    [A-Za-z0-9.-]+|(?:www.|[-;:&=\+\$,\w]+@)
4    [A-Za-z0-9.-]+)((?:\/[\+~%\/.\w-_]*)
5    ?\??(?:[-\+=&;%@.\w_]*)#?(?:[\w]*))?)`)
6  withoutURL := urlRegexp.ReplaceAllString(withoutMention, "")
```

MATCHING URLS

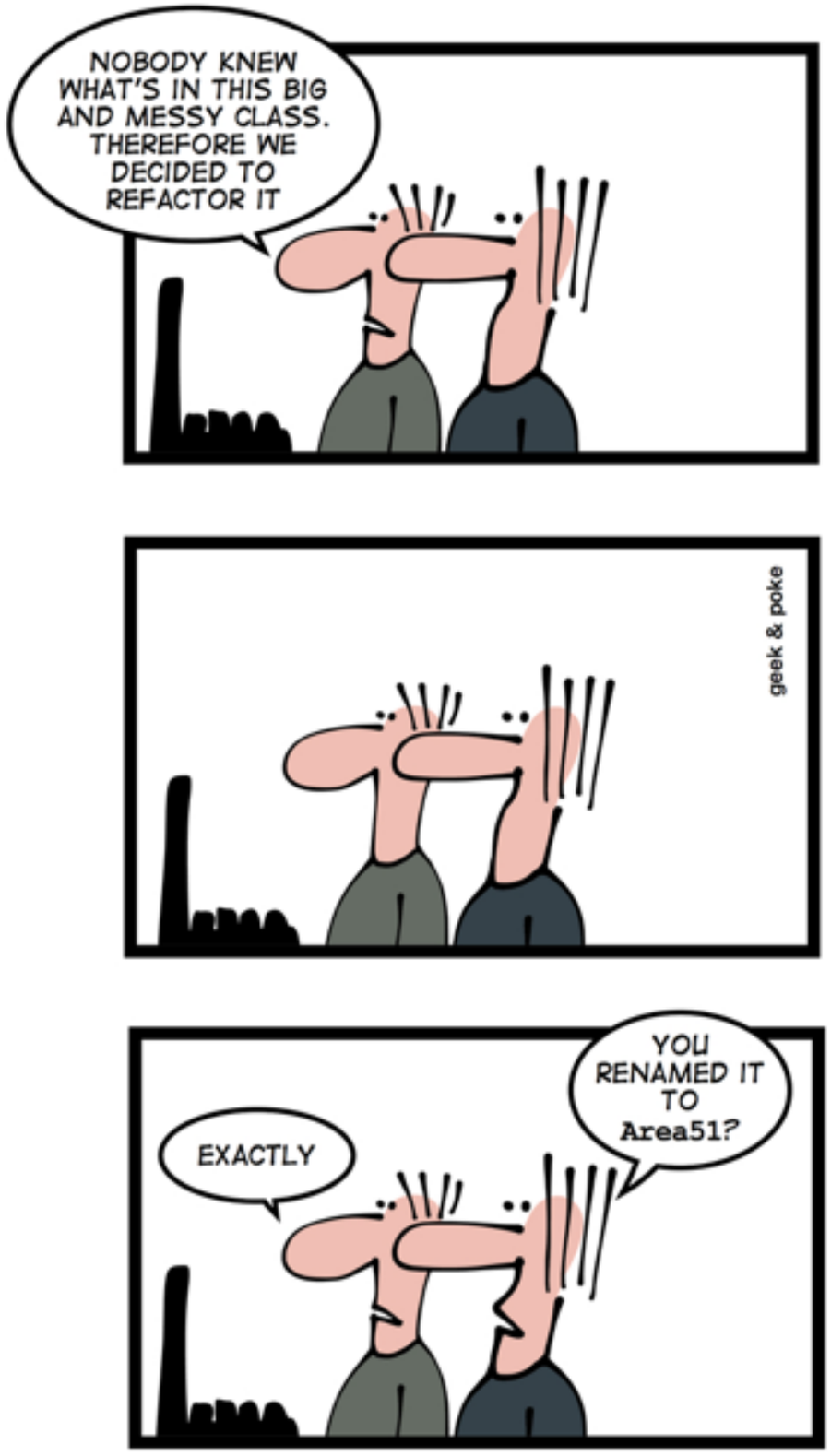
- mathiasbynens.be/demo/url-regex
- stackoverflow.com/questions/161738/what-is-the-best-regular-expression-to-check-if-a-string-is-a-valid-url

```go
func deleteEmoticons(text string) string {
  normalizedString := bytes.NewBufferString("")
  for _, runeValue := range text {
    if runeValue >= rune(0x1F600) && runeValue <= rune(0x1F64F) {
      normalizedString.WriteString("")
    } else {
      normalizedString.WriteRune(runeValue)
    }
  }
  return normalizedString.String()
}
```

```go
func extractNGrams(text string, steps int) []string {
  var ngrams []string
  for i := 0; i < (len(text) - steps + 1); i++ {
    ngrams = append(ngrams, text[i:i+steps])
  }
  return ngrams
}

```

```go
package main

import (
  "bytes"
  "fmt"
  "regexp"
  "strings"
)

func deleteEmoticons(text string) string {
  normalizedString := bytes.NewBufferString("")
  for _, runeValue := range text {
    if runeValue >= rune(0x1F600) && runeValue <= rune(0x1F64F) {
      normalizedString.WriteString("")
    } else {
      normalizedString.WriteRune(runeValue)
    }
  }
  return normalizedString.String()
}

func extractNGrams(text string, steps int) []string {
  var ngrams []string
  for i := 0; i < (len(text) - steps + 1); i++ {
    ngrams = append(ngrams, text[i:i+steps])
  }
  return ngrams
}

func main() {
  awesomeTweet := "Die @bedcon ist die großartigste #Konferenz des Jahres. 😍 http://bedcon.org"
  lowerCasedTweet := strings.ToLower(awesomeTweet)
  fmt.Printf("Lower Case: %v \n", lowerCasedTweet)

  withoutHashtag := strings.Replace(lowerCasedTweet, "#", "", -1)
  fmt.Printf("Without Hashtag: %v \n", withoutHashtag)

  withoutMention := strings.Replace(withoutHashtag, "@", "", -1)
  fmt.Printf("Without Mention: %v \n", withoutMention)

  urlRegexp := regexp.MustCompile(`(((
    [A-Za-z]{3,9}:(?:\/\/)?)
    (?:[-;:&=\+\$,\w]+@)?
    [A-Za-z0-9.-]+|(?:www.|[-;:&=\+\$,\w]+@)
    [A-Za-z0-9.-]+)((?:\/[\+~%\/.\w-_]*)?
    \??(?:[-\+=&;%@.\w_]*)#?(?:[\w]*))?)`)
  withoutURL := urlRegexp.ReplaceAllString(withoutMention, "")
  fmt.Printf("Without URL: %v \n", withoutURL)

  withoutEmoticons := deleteEmoticons(withoutURL)
  fmt.Printf("Without Emoticons: %v \n", withoutEmoticons)

  nGrams := extractNGrams(withoutEmoticons, 2)
  fmt.Printf("NGrams: %v \n", nGrams)
}
```

REFACTORING IS KEY



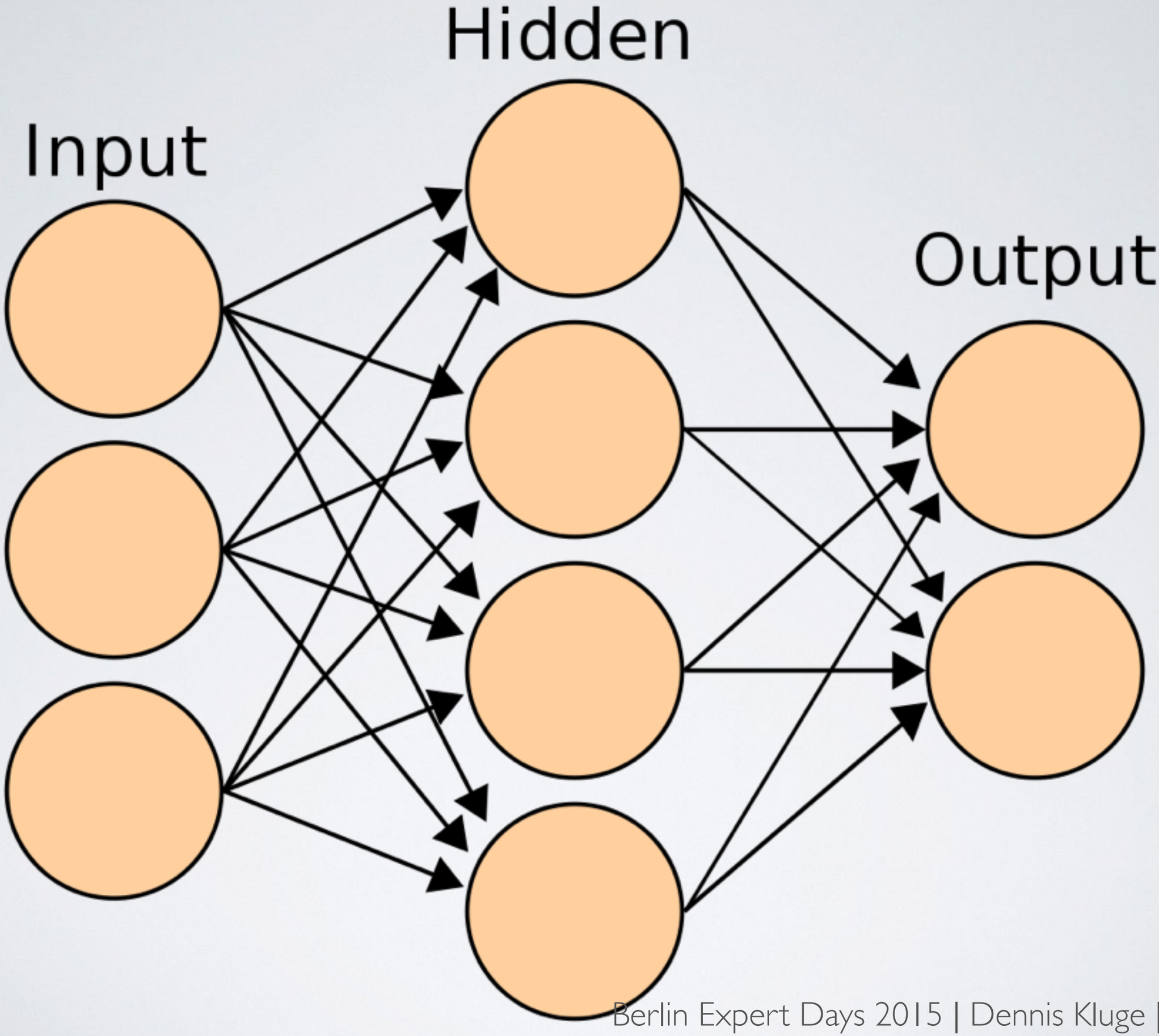
```
chain := NewChainNormalizer(
NewLowerCaseNormalizer(),
NewUnicodeRangeNormalizerFromChart(normalizers.EmoticonChart, ""),
NewUrlReplacementNormalizer(""),
NewRangeTableNormalizer(unicode.Latin, " "),
NewStringReplacementNormalizer([]string{"?", ".", ","}, ""),
NewWhitespaceNormalizer())
```

github.com/horstmumpitz/bedcon2015

BACK TO THE FUTURE™

BAG OF BIGRAMS

Bigram	Tweet 1	Tweet 2	Tweet 3
di	1	2	0
ie	0	5	4
e_	3	0	9
…			


Hidden
Input
Output

–dennis.kluge@charite.de
– @HorstMumpitz