Similarity Tester - SIM

(Dick Grune)

Dozent: Raphael Wimmer

Referenten: Wolfgang Groß, Oliver Spies

Lehrstuhl für Medieninformatik

Universität Regensburg

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester - Hintergrund

- Entwickelt von Dick Grune an der Universität Amsterdam
- Hauptaufgabe: Überprüfung von Programmcode auf mögliche Plagiate im universitären Umfeld
    - Wird an der Universität Amsterdam eingesetzt, um die aktuellen Arbeiten der Studenten mit den Arbeiten etlicher vergangener Jahre zu vergleichen
    - Liefert laut Entwickler effizient sehr gute Ergebnisse

VU VRIJE UNIVERSITEIT AMSTERDAM

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 1 – Einlesen der Textdateien

```
sources = list()


if(len(sys.argv) > 1):
    for x in range(1,len(sys.argv)):
        source = open(sys.argv[x], "r")
        clean = str(source.read())
        clean = clean.replace("\n"," ")
        sources.append(clean)
```

```
C:\sim>python sim_v3.1.py wikiinformatikalt.txt wikiinformatikneu.txt
```

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 2 – Aufsplitten der Texte in Sätze

```
numberTexts = len(sources) # Anzahl der Texte

content = [] # Liste mit Listen aller Saetze a

for i in range(numberTexts):
    contentTemp = []
    contentTempStep = sources[i].split(".") #
    for k in range(len(contentTempStep)):
        contentTemp.append(contentTempStep[k])
    content.append(contentTemp)
```

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 3 – Aufsplitten der Sätze in einzelne Tokens

```
words = [] # Liste mit allen Woertern aller Saet

for i in range(numberTexts):
    numberSentences = len(content[i])
    wordsSentence = []
    for j in range(numberSentences):
        wordsTemp = []
        wordsTempStep = content[i][j].split(" ")
        for k in range(len(wordsTempStep)):
            wordsTemp.append(wordsTempStep[k])
        wordsSentence.append(wordsTemp)
    words.append(wordsSentence)
wordsOr = words #Saves the original Text
```

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 4 – Entfernen der Satzzeichen

```
for i in range(numberTexts):
    numberSentences = len(words[i])
    for j in range(numberSentences):
        numberWords = len(words[i][j])
        for k in range(numberWords):
            words[i][j][k] = words[i][j][k].replace(",","")
            words[i][j][k] = words[i][j][k].replace("\"","")
            words[i][j][k] = words[i][j][k].replace("(","")
            words[i][j][k] = words[i][j][k].replace(")","")
            # beliebig erweiterbar
```

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 5 – Stoppwörter entfernen

```
stopWords = []
fobj = open("stopwords.txt", "r")
for line in fobj:
    stopWords.append(line.replace("\r\n",""))
fobj.close()



numberStopWords = len(stopWords)
for i in range(len(words)):
    for j in range(len(words[i])):
        for k in range(len(words[i][j])):
            for l in range(numberStopWords):
                if words[i][j][k] == stopWords[l]:
                    words[i][j][k] = "" # Loeschen
```

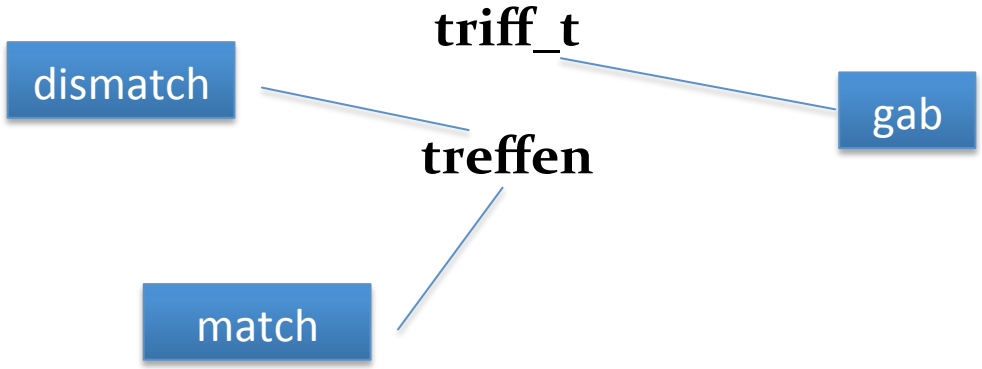
Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Sim Tester – Der Algorithmus

Schritt 6 – Score

2 Strings miteinander vergleichen liefert einen Numerischen wert



Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Der Algorithmus

```
def getStringAlignmentMatrix(S,T):
    #S and T are the strings to compare
    #String alignment score is im
    g = -2      #g := gab score
    n = len(S)
    k = len(T)
    D = [[0]*(n+1) for i in range(k+1)]
    #fill the gap scores
    for i in range(0,n):
        D[0][i+1] = g*(i+1)
    for i in range(0,k):
        D[i+1][0] = g*(i+1)

    #fil the table
        for i in range(1,k+1):
            for j in range(1,n+1):
                match = D[i-1][j-1] + sim_score(S[j-1],T[i-1])
                gaps = D[i][j-1] + g
                gapt = D[i-1][j] + g
                D[i][j] = max(match,gaps,gapt)
    return D
```

		t	r	i	f	f	t
	0	-2	-4	-6	-8	-10	-12
t	-2	0	0	0	0	0	0
r	-4	0	0	0	0	0	0
e	-6	0	0	0	0	0	0
f	-8	0	0	0	0	0	0
f	-10	0	0	0	0	0	0
e	-12	0	0	0	0	0	0
n	-14	0	0	0	0	0	0

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

```
def getStringAlignmentMatrix(S,T):
    #S and T are the strings to compare
    #String alignment score is im
    g = -2      #g := gab score
    n = len(S)
    k = len(T)
    D = [[0]*(n+1) for i in range(k+1)]
    #fill the gap scores
    for i in range(0,n):
        D[0][i+1] = g*(i+1)
    for i in range(0,k):
        D[i+1][0] = g*(i+1)

    #fil the table
        for i in range(1,k+1):
            for j in range(1,n+1):
                match = D[i-1][j-1] + sim_score(S[j-1],T[i-1])
                gaps = D[i][j-1] + g
                gapt = D[i-1][j] + g
                D[i][j] = max(match,gaps,gapt)
    return D
```

		t	r	i	f	f	t
	0	-2	-4	-6	-8	-10	-12
t	-2	1	0	0	0	0	0
r	-4	0	0	0	0	0	0
e	-6	0	0	0	0	0	0
f	-8	0	0	0	0	0	0
f	-10	0	0	0	0	0	0
e	-12	0	0	0	0	0	0
n	-14	0	0	0	0	0	0

match = 1

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

```
def getStringAlignmentMatrix(S,T):
    #S and T are the strings to compare
    #String alignment score is im
    g = -2       #g := gab score
    n = len(S)
    k = len(T)
    D = [[0]*(n+1) for i in range(k+1)]
    #fill the gap scores
    for i in range(0,n):
        D[0][i+1] = g*(i+1)
    for i in range(0,k):
        D[i+1][0] = g*(i+1)

    #fil the table
        for i in range(1,k+1):
            for j in range(1,n+1):
                match = D[i-1][j-1] + sim_score(S[j-1],T[i-1])
                gaps = D[i][j-1] + g
                gapt = D[i-1][j] + g
                D[i][j] = max(match,gaps,gapt)
    return D
```

		t	r	i	f	f	t
	0	-2	-4	-6	-8	-10	-12
t	-2	1	0	0	0	0	0
r	-4	0	0	0	0	0	0
e	-6	0	0	0	0	0	0
f	-8	0	0	0	0	0	0
f	-10	0	0	0	0	0	0
e	-12	0	0	0	0	0	0
n	-14	0	0	0	0	0	0

gaps = -4
gapt = -4

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

		t	r	i	f	f	t
	0	-2	-4	-6	-8	-10	-12
t	-2	1	-1	0	0	0	0
r	-4	0	0	0	0	0	0
e	-6	0	0	0	0	0	0
f	-8	0	0	0	0	0	0
f	-10	0	0	0	0	0	0
e	-12	0	0	0	0	0	0
n	-14	0	0	0	0	0	0

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

		t	r	i	f	f	t
	0	-2	-4	-6	-8	-10	-12
t	-2	1	-1	-3	-5	-7	-9
r	-4	-1	2	0	-2	-4	-6
e	-6	-3	0	1	-1	-3	-5
f	-8	-5	-2	-1	2	0	-2
f	-10	-7	-4	-3	0	3	1
e	-12	-9	-6	-5	-2	1	2
n	-14	-11	-8	-7	-4	-1	0

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Next Steps

```
def getScore(S,T):
    D = getStringAlignmentMatrix(S,T)
    if D[len(T)][len(S)]>0:
        return D[len(T)][len(S)]
    else:
        return 0
```

Score ist immer ≥ null

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

```
def getScoreSentence(S,T):
    #S and T are lists of tokens
    sum = 0;
    for i in range(0,len(S)):
        for j in range(0,len(T)):
            sum += getScore(S[i],T[j])
    return sum
```

Score von jedem Wort in jedem anderen Wort summieren.

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

```
def getProc(S,T):
    if float(getScoreSentence(S,S))>0.0 or float(getScoreSentence(T,T))>0.0:
        a = (2*float(getScoreSentence(S,T)))/(float(getScoreSentence(S,S))+float(getScoreSentence(T,T)))
    else:
        a = 0.0
    return a
```

Score normalisieren

neuer Wert ist zwischen 0 und 1

$$\frac{2 * scoreSentence(S,T)}{scoreSentence(S,S) + scoreSentence(T,T)}$$

Universität Regensburg

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

```
def simSent(words):
    result = list()
    for i in range(1,len(words)):
        #all senteces in original
        simSent = 0
        for j in range(0,len(words[0])):
            #all sentences in other files
            for n in range(0,len(words[i])):
                temp = getProc(words[0][j],words[i][n])
                if temp > 0.5:
                    print "\n\nin Text:"+sys.argv[i+1]+"\n"
                    simSent += 1
        result.append(simSent)
    return result
```

Schwellenwert

Jeden Satz aus Dokument (Original) mit

jedem Satz aus

allen andern Quellen vergleichen

Wolfgang Groß, Oliver Spies
Lehrstuhl für Medieninformatik
FAKULTÄT FÜR SPRACH-, LITERATUR- UND KULTURWISSENSCHAFTEN

Quellen

http://dickgrune.com/Programs/similarity_tester/ [10.05.2012]
http://www.biorecipes.com/DynProgBasic/code.html [23.05.2012]