Seminararbeit

Erkennen von Emotionen in Tweets

Adrian Schmid - schmiad1@students.zhaw.ch

Zürich, 16.06.2014

Inhaltsverzeichnis

1 Einleitung

1.1 Thema, Motivation und Ziel der Arbeit

Als Basis für viele soziale Netzwerkdatenanalysen ist es wichtig erkennen zu können welche Gefühle mit bestimmten Textfragmenten verknüpft werden. Es gibt verschiedene Ansätze und Algorithmen um dies zu erreichen. Im Rahmen dieser Seminararbeit möchte ich mich mit verschiedenen Algorithmen und Ansätzen zum Erkennen einer negativen oder positiven Grundhaltung beziehungsweise Grundstimmung in kurzen Textfragmenten festzustellen auseinandersetzen.

Ziel der Arbeit ist es zuerst verschiedene Verfahren zu untersuchen und kennenzulernen um danach entscheiden zu können welche Verfahren im Rahmen dieser Arbeit implementiert werden könnten. Schlussendlich sollen die umsetzbaren Verfahren implementiert werden.

Die Implementation soll dabei folgende Funktionalität aufweisen:

1. In einem ersten Schritt wird ein Thema erfasst. Ein Thema beinhaltet einen „Titel“, „Stichwörter“ und einem Zeitrahmen. Nach diesen Attributen sollten Tweets auf Twitter gefunden werden können.
2. Das Programm soll mit diesen Daten relevante Tweets herauslesen.
3. Nun sollen Parallel verschieden Algorithmen angestossen werden welche entscheiden ob die definierten Tweets eher positiv oder eher negativ sind.
4. Dem Benutzer wird ausgegeben welche Algorithmen wieviele Tweets aus der zuvor zusammengestellten Menge „positiv“ oder „negativ“ bewerten.

1.2 Planung

Informationen beschaffen & Einarbeitung in Twitter API und Algorithmen	13.03.2014 – 20.03.2014
Erstellen eines Konzepts, Festlegen welche Algorithmen implementiert werden sollen bzw können.	20.03.2014 – 02.04.2014
Milestone 1: Konzept erstellt	**02.04.2014**
Analyse und Design der Software	03.04.2014 – 16.04.2014
Implementation	17.04.2014 – 07.05.2014
Testing / Dokumentation	08.05.2014 – 14.05.2014
Milestone 2: Implementation abgeschlossen	**14.05.2014**
Vergleich der Ergebnisse	15.05.2014 – 21.05.2014
Dokumentation	22.05.2014 – 28.05.2014
Endkorrektur / Abschluss	29.05.2014 – 16.06.2014
Milestone 3: Arbeit fertig	**16.06.2014**
Abgabe der Arbeit	16.06.2014
Präsentation	19.06.2014

Tabelle 1: Projektplanung

2 Grundlagen

2.1 Algorithmen zur Erkennung der Gefühlslage in Texten

Es wird grundsätzlich zwischen zwei Ansätzen zur Analyse der Gefühlslage in Texten unterschieden. Der erste Ansatz verwendet machine-learning Technologien der zweite basiert auf der lexikalischen Analyse der Texte. Für den ersten Ansatz wird ein Korpus mit gelabelten Trainingsdaten benötigt. [12] Der Hauptvorteil von machine-learning basierten Methoden ist, dass sie sich an neue Gegebeneheiten anpassen können. Der grosse Nachteil ist, dass keine gelabelten Trainingsdaten für neue Themen existieren und dass das erstellen eines neuen Datenkorpus sehr aufwändig und teuer ist. Die lexikalischen Methoden haben eine vordefinierte Liste von Wörtern welche mit bestimmten Gefühlen verknüpft sind. Um gute Resultate zu erhalten, müssen diese Wortlisten der entsprechenden Domäne angepasst werden. So werden zum Beispiel im Web-Slang andere Wörter mit anderen Gefühlen verknüpft als in einem formalen Text. [10]

Im Rahmen dieser Arbeit werden sowohl machine-learning basierte als auch lexikalische Algorithmen zur Erkennung der Gefühlslage in Texten untersucht und - wenn in Python umsetzbar - auf Tweets zu verschiedenen Themen angewandt.

2.1.1 Emoticons

Der einfachste Ansatz zum erkennen der Grundhaltung des Authors gegenüber eines Themas ist die Untersuchung der Emoticons die es beinhaltet. Emoticons sind meist Kombinationen von ASCII Zeichen die einen Gesichtsausdruck darstellen. Dieser kann Gefühle unter anderem gefühle wie glücklich oder traurig representieren. Für diesen Algorithmus wurde eine Menge von gängigen Emoticons aus dem Web [1][7][3] zusammengesucht und manuell in «positiv», «neutral», und «negativ»klassifiziert. Mithilfe solcher Listen kann die Anzahl Emoticons der entsprechenden Kategorie in einem Text zählen und gegeneinander Abwägen. [10] Die Umsetzung eines solchen Algorithmus in Python stellt keine Probleme dar.

2.1.2 LIWC - Linguistic Inquiry and Word Count

LIWC ist ein Textanalysetool welche emotionale, kognitive und strukturelle Komponenten von Texten mithilfe eines Wörterbuchs klassifiziert. Die LIWC Software wird kommerziell vertrieben und kann - bei Bedarf - um eigene Wörterbücher ergänzt werden. LIWC liest Text ein und generiert daraus ein Tab-Delimited File welches zum Beispiel in einem Python Programm oder mithilfe von SPSS weiter verarbeitet werden könnte.

Es existieren Versionen für Windows und Macintosh Computer. [10][4] Die fehlende Verfügbarkeit für Linux, die komplizierte Anbindung an ein Python Programm und ökonomische Gründe haben dazu geführt, dass im Rahmen dieser Arbeit keine LIWC-Python Anbindung implementiert wurde.

2.1.3 SentiStrength

SentiStrength ist ein machine-learning basiertes Textanalyse Tool welches spezialisiert ist auf die Analyse von Texten aus dem Social-Web. Die Freie Version ist auf der Basis des .Net Technologiestacks implementiert und nur für Windows verfügbar. Eine kommerzielle Version des Tools wurde in Java geschrieben. Die kommerzielle Java Version ist für Forschungszwecke frei verfügbar und kann per Email angefordert werden. Diese Version liesse sich wie im Listing 1 aufgezeigt an Python Programme anbinden. [5][10]

```
#Alec Larsen - University of the Witwatersrand, South Africa, 2012 import shlex, subprocess

def RateSentiment(sentiString):
    #open a subprocess using shlex to get the command line string into the correct args list format
    p = subprocess.Popen(shlex.split("java -jar SentiStrength.jar stdin sentidata C:/SentStrength_Data/"),stdin=subprocess.PIPE,stdout=subprocess.PIPE,stderr=subprocess.PIPE)
    #communicate via stdin the string to be rated. Note that all spaces are replaced with +
    stdout_text, stderr_text = p.communicate(sentiString.replace(" ","+"))
    #remove the tab spacing between the positive and negative ratings. e.g. 1-5 -> 1-5
    stdout_text = stdout_text.rstrip().replace("\t","")
    return stdout_text
```

Listing 1: Python Anbindung an SentiStrength (JAVA)

Der entsprechende Email-Kontakt hat nicht innert nützlicher Frist geantwortet, weshalb dieser Algorithmus aus dieser Arbeit ausgeklammert werden musste.

2.1.4 SentiWordNet

Das SentiWordNet ist eine offen verfügbare lexikalische Ressource (Wörterbuch) mithilfe welchem ein Text auf folgendes untersucht werden kann:

- Subjektivität-Objektivität: Besteht ein gegebener Text primär aus Fakten, oder haften ihm Emotionen an?

- Positiv-Negativ: Sind die Emotionen die im Text ausgedrückt werden primär positiv oder primär negativ?
- Stärke der Emotion: Wie stark ist die positive oder negative Emotion in einem Text?

Das Wörterbuch wird als Tabulator-getrenntes Textfile unter `http://sentiwordnet.isti.cnr.it/download.php` zu Verfügung gestellt. Das Wörterbuch enthält folgende Spalten:

- die Wortart ('a' = Adjektiv, 'n' = Nomen, ...)
- eine ID (z.B. 00004980)
- einen Wert der die Stärke der positiven Emotionen anzeigt (Zwischen 0 und 1, folgend auch *posScore* genannt)
- einen Wert der die Stärke der negativen Emotionen anzeigt (Zwischen 0 und 1, folgend auch *negScore* genannt)
- das Wort an sich
- eine Liste von Synonymen
- eine Beschreibung des Wortes

Den Wert für die Objektivität (*objScore*) lässt sich wie folgt berechnen:

$$objScore = 1 - (negScore + posScore) \tag{1}$$

Zum besseren Verständnis sind im Listing 2 drei Zeilen aus dem aktuellen SentiWordNet Wörterbuch abgebildet.

```
a 00004980 0 0 unabridged#1 (used of texts) not shortened; "an
    unabridged novel"
a 00005107 0.5 0 uncut#7 full-length#2 complete; "the full-length play"
a 00007813 0 0.5 nonabsorptive#1 nonabsorbent#1 not capable of absorbing
    or soaking up (liquids)
```

Listing 2: SentiWordNet Zeile)

Die Gute Dokumentation und die freie Verfügbarkeit aller Ressourcen lassen eine Python Implementation des SentiWordNet Wörterbuches zu.

2.1.5 SenticNet

SenticNet ist eine weitere lexikalische Resource welche im Opinion Mining verwendet werden kann. Das SenticNet kann unter `http://sentic.net/downloads/` heruntergeladen werden. Das Wörterbuch ist in diesem Falle als XML abgespeichert. Ein Wort-Eintrag enthält neben dem Wort selbst vor allem vier Werte für pleasentness, attention, sensitivity und aptitude. Aus diesen Werten lässt sich mit folgender Formel die allgemeine Polarität (positiv oder negativ) berechnen:[8]

$$p = \sum_{i=1}^{N} \frac{Plsnt(c_i) + |Attnt(c_i)| - |Snst(c_i)| + Aptit(c_i)}{3N} \tag{2}$$

In Version 2 des SenticNet wird dieser Wert aus praktischen Gründen jeweils pro Wort direkt im Wörterbuch mitgeliefert. Um die Polarität eines ganzen Textes zu berechnen wird jedoch weiterhin die oben genannte Formel empfohlen. [8] Zum einfacheren Verständnis befindet sich unter Listing 3 Beispielhaft ein Eintrag eines Wortes.

```
<rdf:Description rdf:about="http://sentic.net/api/en/concept/worship">
  <rdf:type rdf:resource="http://sentic.net/api/concept"/>
  <text xmlns="http://sentic.net/api/">worship</text>
  <semantics xmlns="http://sentic.net/api/" rdf:resource="http://sentic.
    net/api/en/concept/hope"/>
  <semantics xmlns="http://sentic.net/api/" rdf:resource="http://sentic.
    net/api/en/concept/religious_purpose"/>
  <semantics xmlns="http://sentic.net/api/" rdf:resource="http://sentic.
    net/api/en/concept/trust"/>
  <semantics xmlns="http://sentic.net/api/" rdf:resource="http://sentic.
    net/api/en/concept/devotion"/>
  <semantics xmlns="http://sentic.net/api/" rdf:resource="http://sentic.
    net/api/en/concept/religious"/>
  <pleasantness xmlns="http://sentic.net/api/" rdf:datatype="http://www.w3
    .org/2001/XMLSchema#float">+0.265</pleasantness>
  <attention xmlns="http://sentic.net/api/" rdf:datatype="http://www.w3.
    org/2001/XMLSchema#float">+0.601</attention>
  <sensitivity xmlns="http://sentic.net/api/" rdf:datatype="http://www.w3.
    org/2001/XMLSchema#float">-0.207</sensitivity>
  <aptitude xmlns="http://sentic.net/api/" rdf:datatype="http://www.w3.org
    /2001/XMLSchema#float">+0.373</aptitude>
  <polarity xmlns="http://sentic.net/api/" rdf:datatype="http://www.w3.org
    /2001/XMLSchema#float">+0.344</polarity>
</rdf:Description>
```

Listing 3: SenticNet Wort

Alle Grundlagen und das Wörterbuch des SenticNet sind frei zugänglich, weshalb sich auch hier eine entsprechende Python Implementation anbot.

2.1.6 SASA - SailAil Sentiment Analyzer

Der SailAil Sentiment Analyzer ist eine machine-learning basierte Library die direkt für Python verfügbar ist. In dieser Arbeit wurde die Library in der Version 0.1.3 verwendet (`https://pypi.python.org/pypi/sasa/0.1.3`)

2.1.7 Klassifizierung mit NLTK und naivem Bayes

Bei der Recherche zu dieser Arbeit bin ich über einen Artikel von Jacob Perkins [13] gestolpert. Er beschreibt wie man mithilfe des Natural Language Toolkits (NLTK) [9] einen einfachen Sentiment Analyzer implementieren kann. Er trainiert einen `nltk.classify.NaiveBayesClassifier` mithilfe des NLTK eigenen Film-Review Korpus (`nltk.corpus.movie_reviews`).

2.2 Twitter Search API

Das Twitter Search API[2] ist ein REST API welches es erlaubt Tweets zu bestimmten Themen zu suchen. Christian Koepp hat eine Python Anbindung an dieses API implementiert[11]. Mithilfe dieser Anbindung lässt sich in wenigen Zeilen (Siehe Listing 4) eine Twitter Anbindung realisieren.

```
from TwitterSearch import *
try:
    tso = TwitterSearchOrder()
    tso.setKeywords(['Worldcup', 'Brazil'])
    tso.setLanguage('en')
    tso.setIncludeEntities(False)

    ts = TwitterSearch(
      consumer_key = 'aaabbb',
      consumer_secret = 'cccddd',
      access_token = '111222',
      access_token_secret = '333444'
    )

    for tweet in ts.searchTweetsIterable(tso):
        print( '@%s tweeted: %s' % ( tweet['user']['screen_name'], tweet['
    text'] ) )
except TwitterSearchException as e:
    print(e)
```

Listing 4: TwitterSearch Python-Twitter Anbindung

Was das Twitter Search API nicht bietet, ist die Möglichkeit nach Tweets in bestimmten Zeiträumen zu suchen. Man sucht jeweils im aktuellen SearchIndex welcher sowohl aus populären als auch sehr aktuellen Tweets besteht. Für die in der Aufgabenstellung beschriebene Zeitraum-Auswahl-Funktion könnten Daten von einem kommerziellen Anbieter bezogen werden oder man könnte sich selber ein Twitter Archiv mithilfe der Streaming API [6] anlegen. Im Rahmen dieser Arbeit wurde darauf verzichtet. Das heisst, in der im Rahmen dieser Arbeit erstellten Applikation kann «nur»nach aktuellen oder populären Tweets mithilfe von Stichwörtern gesucht werden.

3 Umsetzung

3.1 Android App

3.2 Distanzberechnung

3.3 Akkumessung

3.4 GPS XML

4 Analyse

4.1 Genauigkeit der Daten

4.2 Akkuverbrauch

4.3 Verwendbarkeit der Daten

5 Fazit

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Literatur

[1] Emoticons - show your friends how you really feel. `https://messenger.yahoo.com/features/emoticons`.

[2] Get search/tweets. `https://dev.twitter.com/docs/api/1.1/get/search/tweets`.

[3] List of text emoticons – the ultimate resource. `http://cool-smileys.com/text-emoticons`.

[4] The liwc2007 application. `http://www.liwc.net/howliwcworks.php`.

[5] Sentistrength - java version. `http://sentistrength.wlv.ac.uk/#Java`.

[6] The streaming apis. `https://dev.twitter.com/docs/api/streaming`.

[7] What is the full list of emoticons? `https://support.skype.com/en/faq/FA12330/what-is-the-full-list-of-emoticons`.

[8] Erik Cambria, Catherine Havasi, and Amir Hussain. Senticnet 2: A semantic and affective resource for opinion mining and sentiment analysis. 2012.

[9] Dan Garrette, Peter Ljunglöf, Joel Nothman, Mikhail Korobov, Morten Minde Neergaard, and Steven Bird. Natural language toolkit. `http://www.nltk.org/`.

[10] Pollyanna Goncalves, Fabricio Benevenuto, Matheus Araujo, and Meeyoung Cha. Comparing and combining sentiment analysis methods. *Proceedings of the first ACM conference on Online social networks.*

[11] Christian Koepp. Twittersearch. `https://github.com/ckoepp/TwitterSearch`.

[12] B Pang, L Lee, and S Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. *ACL Conference on Empirical Methods in Natural Language Processing.*

[13] Jacob Perkins. Text classification for sentiment analysis - naive bayes classifier. `http://streamhacker.com/2010/05/10/text-classification-sentiment-analysis-naive-bayes-classifier/`.