Tag Recommendations

in an Enterprise 2.0 Tool

Stefan Deser
Betreuer: Alexander Steinhoff

28. März 2011

Inhalt

Motivation und Ziele der Arbeit

Motivation: Folksonomies als *die* Form der Wissensrepräsentation des Webs 2.0 weisen Nachteile auf:

- Tagging ist kognitiv anspruchsvoll
- Oft (zu) geringe Indexierungstiefe
- Inkonsistenzen im Vokabular

Abhilfe durch *Tag Recommender Systems* (*TRS*)

Ziele der Arbeit:

- Verschiedene Ansätze für TRS untersuchen
- Integration eines TRS in Tricia

Tag Recommender Systems

Drei Ansätze.

Tag Recommender Systems

Drei Ansätze:

1. **„Wörter“**: Tags als natürlichsprachliche Wörter und maschinenlesbare Wissensrepräsentationen
   - Einsatz von Thesauri, Wörterbüchern wie z.B. *WordNet*
2. **„Inhalt“**: Inhaltsbasierte Tag-Empfehlungen
   - Extraktion relevanter Wörter aus dem Text
   - Techniken des Machine Learning, wie z.B.: Bayes-Klassifikator oder Neuronale Netze.
3. **Struktur**: Strukturbasierte Generierung
   - *FolkRank* Algorithmus in Anlehnung an *PageRank*
   - Co-Occurrence-basierte Ansätze

1. Wörter und Wissensrepräsentationen

Tags als Wörter und Tags aus Wörterbüchern.

Wörter und Wissensrepräsentationen

Idee: Tags als natürlichsprachliche Wörter, Information aus maschinenlesbaren Wissensrepräsentationen extrahieren

Beispiel: *airliner* in WordNet 3.0

- Hypernyme:
  - *airplane*, *aeroplane*, *plane*
- Hyponyme:
  - *airbus*, *narrowbody aircraft*, *narrow-body aircraft*, *narrow-body*, *widebody aircraft*, *wide-body aircraft*, *wide-body*, *twin-aisle airplane*
- Meronyme:
  - *galley*, *plane seat*
- Assoziationen:
  - *biplane*, *bomber*, *delta wing*, *fighter*, *fighter aircraft*, *attack aircraft*, *jet*, *jet plane*, *jet-propelled plane*, *monoplane*, *multiengine airplane*, *multiengine plane*, *propeller plane*, ...

Bewertung Ansatz „Wörter“

Vorteil:

- Kontextuelle Einordnung möglich

Probleme:

- Keine Lösung des Cold Start-Problems
- Schreibweise der Tags problematisch
- Sprachabhängigkeit
- Kein Standardformat für Wissensrepräsentationen
- Geringe Abdeckung in der Praxis: 39% (163 von 418) durch WordNet und OpenThesaurus

2. Inhaltsbasierte Tag-Generierung

Tags aus dem Text.

Idee und Ablauf

Idee: Extrahiere die „relevanten“ Wörter des Textes.

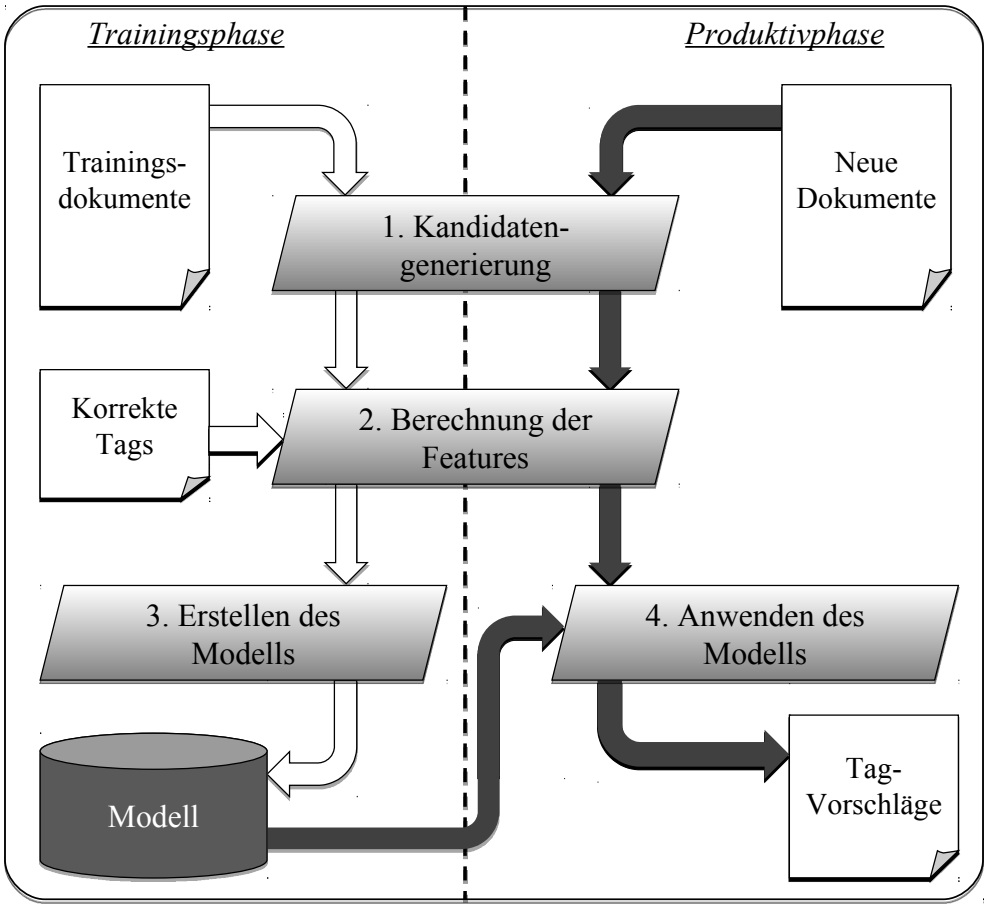

Abb.: Arbeitsablauf nach Medelyan (2009), S. 106

2.1 Kandidatengenerierung

Problem: Welche Wörter kommen als relevante Wörter in Frage?

Beispiel:

> *Die Technische Universität München (kurz TUM oder TU München) ist die einzige Technische Universität in Bayern.*

Zwei Arten der Kandidatengenerierung:

1. n-Gramm-Extraktion
2. PoS-Tagging

n-Gramm-Extraktion

Ein *n-Gramm* ist ein „Wort“ w der Länge n über dem „Alphabet“ Σ: $w = (w_1, \ldots, w_n) \in \Sigma^n$.

- $n = 1$, **Monogramme**:
  - Die
  - Technische
  - Universität
  - ...
- $n = 2$, **Bigramme**:
  - Die Technische
  - Technische Universität
  - Universität München
  - München (kurz
  - ...

Entfernung von Satz- und Sonderzeichen, Benutzung von *Stoppwortlisten*

Wortartbestimmung

Motivation: ca. 90% aller Tags sind Nomen, filtere Nomen durch PoS-Kürzel (part-of-speech: Wortarten)

Wort	**PoS-Kürzel**	**Bedeutung**
Die	ART	Artikel
Technische	ADJA	Adjektiv
Universität	NN	Nomen
München	NN	
...	...	
ist	VAFIN	finites Verb, aux
die	ART	
einzige	ADJA	
Technische	ADJA	
Universität	NN	
in	APPR	Präposition
Bayern.	NE	Eigenname

2.2 Berechnung der Features

Jeder Kandidat wird als **Feature-Vektor** repräsentiert.
Dieser dient als Eingabe für Klassifizierer („Tag“, „Kein Tag“).

Klassifizierer fungiert als „Black Box“
(konkrete Implementierung austauschbar)

Dafür sind verschiedene Features möglich, wie etwa:

- TF (Term Frequency)
- Auftretensposition
  - First Occurrence
  - Last Occurrence
  - Occurrence Spread
- ...

2.3/2.4 Erstellen und Anwenden des Modells

Erstellen eines **Klassifikationsmodells** anhand eines **Trainigsdatensatzes**.

Eingabe: Feature-Vektor **und** Klasse („Tag“, „Kein Tag“)

Anschließend wird das Modell **gespeichert**,
d.h. Lernen nur einmal erforderlich.

Verschiedene Klassifizierer möglich (WEKA):

- Künstliche Neuronale Netze
- Naïver Bayes
- Decision Tree Learner

Beispiel für Klassifikation

Die Technische Universität München (kurz TUM oder TU München) ist die einzige Technische Universität in Bayern.

Kandidat „Universität“ (k_{uni}) weist folgende Feature-Werte auf:

- Term Frequency: $\frac{2}{16} = 0,125$
- First Occurrence: $\frac{3}{16} = 0,1875$

Klassifizierer entscheidet über Klassenzugehörigkeit von k_{uni} anhand des gelernten Modells:

$$P(k_{uni} \in \text{„Tag“}) = 0,89$$

Bewertung Ansatz „Inhalt“

Vorteile:

- Lösung des Cold Start-Problems
- Generierung neuer Tags

Probleme:

- Keine kontextuelle Einordnung
  Ausschließlich Tags aus dem Text
- Kein Rückgriff auf vorhandene Tags (Konsistenz!)
- Kandidatengenerierung und Feature-Auswahl
- Training für Modelle erforderlich
  - Auswahl der Trainingsdaten
  - Heterogenität der Ressourcentypen

Strukturbasierte Ansätze

Tags aus der Folksonomy.

„Struktur“: FolkRank-Algorithmus

Grundidee: Modifizierter PageRank-Algorithmus für Nutzung in Folksonomies.

Dieser Ansatz wurde nicht weiter verfolgt, da er eine Broad Folksonomy benötigt, Tricia diese jedoch nicht unterstützt.

„Struktur“: Co-Occurrence-basierte Ansätze

Zwei verschiedene Möglichkeiten für Kandidatengenerierung:

- Verwendung bereits annotierter Tags
- Ähnliche Dokumente

Grundsätzliches Vorgehen:

- Sammle alle gemeinsam auftretenden Tags
- Wähle die „relevanten“ aus
  Berücksichtigung der Power Law-Verteilung, z.B.:

$$M := \{r \mid r \text{ ist annotiert mit } k\}$$

$$descriptive(k) := \frac{p_d}{p_d + abs(p_d - log(|M|))}$$

Verteilung der Tags in untersuchter Folksonomy

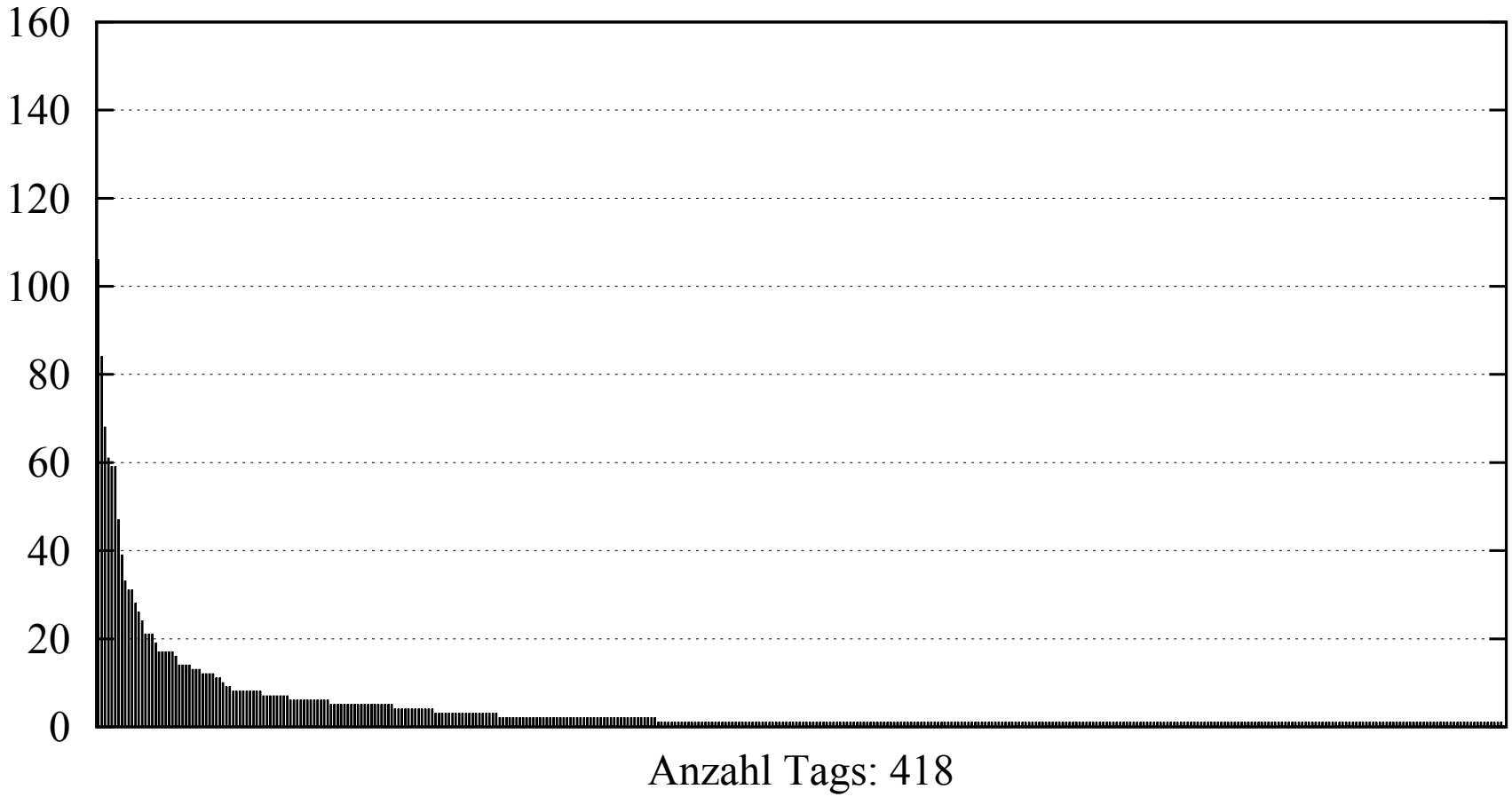

Abb.: Die Verteilung der Tags folgt, wie erwartet, dem Power Law.

Bewertung

Vorteile:

- Wahrung der Konsistenz
- Einordnung in den Kontext der Folksonomy

Probleme:

- Keine Vorschläge neuer (besserer?) Tags
- Matthäus-Effekt

Kombination der Ansätze

Kein Ansatz für sich ist ausreichend:

- Abdeckung der Tags in Wörterbüchern äußerst problematisch
- Cold Start-Problem kann nicht strukturbasiert gelöst werden
- Inhaltsbasierte Ansätze: kein Zugriff auf vorhandene Tags

Implementierung in Tricia:

- Inhaltsbasierte Ansätze
  - n-Gramm-Extraktion
  - PoS-Tagging
- Co-Occurrence Ansatz

TRS in der Praxis

Vorführung des implementierten TRS.

Zusammenfassung

1. Tags als natürlichsprachliche Wörter – NLP und maschinenlesbare Wissensrepräsentationen
   - Erfolgversprechend in der Theorie
   - Enttäuschend in der Praxis
2. Inhaltsbasierte Tag-Empfehlungen
   - Lösung des Cold Start-Problems
   - Vorschläge neuer Tags
3. Strukturbasierte Generierung
   - Je mehr Daten, umso besser
   - Wahrung der Konsistenz
   - Gefahr des Matthäus-Effekts

- Kombination der Ansätze erforderlich!
- Erfolgreiche Implementierung in Tricia

Vielen Dank für die Aufmerksamkeit!