Organizational

- Acquisition of credit points (3 CP)
  - Pass the final written exam (or oral exam)
  - probable date: 17th of July (last lecture)
- Dates
  - Lecture: every Tuesday 15:15 to 16:45, A-1.2
  - Two exercises instead of lectures
  - (already known) exceptions
    - lecture on 1st of May will be dropped due to public holiday
Organizational

- **Exercises**
  - Emission of exercise sheets one week in advance of the exercise date
  - Discussion of your solutions on the exercise date

- **Homework**
  - Slides, Exercises
  - Books

Outline the semester

1. Motivation and Basic Definitions for IF&IR
2. (Computer) Linguistics
3. IF/IR Models and their Classification
4. Practical Usage and Quantitative Evaluation
Motivation and Basic Definitions

Current Situation

- Around 1980 computers get small enough to be used on the desk
  - increase of office applications
  - increase of natural language documents processed and stored by computers
- Since 1990: World Wide Web
  - world-wide distributed documents can be accessed easily
  - publication of documents is easy
=> Huge number of documents are accessible
Infoglut / Information Overload

Situation
– For problem solving and decision making information is essential
– Computers and networks make a huge amount of information available

Problem
Human capacity for information processing is limited

Result
not all information can be processed, relevant information may founder in the flood of information

Solution attempt

• Computer help on filtering and retrieval of relevant information
• Challenges
  – Which information is relevant?
  – Information is written/coded in natural language
  – Natural languages are ambiguous (in comparison to formal languages)
  – Natural languages have a high expressiveness
  – Many different natural languages exist
Information Filtering from the user’s perspective

Profile: “IF&IR”
Information Filtering System

document stream e.g. daily news
relevant docs
not relevant docs

docs about IF&IR are important

© Dominik Kuropka 2007
Findings: IF vs. IR
from the user’s point of view

- Information Filtering (IF) is…
  the selection of documents
  from a dynamic stream of documents
  using some kind of static profile.
  IMPORTANT: good and “future-compatible” profiles!

- Information Retrieval (IR) is…
  the selection of documents
  from a static set of documents
  using an ad-hoc query.
  IMPORTANT: fast retrieval of documents!

- Conclusion
  IF and IR are different tasks!
  IF and IR should be optimized differently.
Information Filtering from the system’s point of view

- Author creates and publishes a document.
- The document is represented.
- A document representation model is created.
- The model is used to match found documents.
- The user has a long-dated information demand.
- The demand is expressed by a profile.
- The profile is used to read and evaluate found documents.
- If necessary, the model is modified.

Information Retrieval from the system’s point of view

- Author creates and publishes a document.
- The document is represented.
- A document representation model is created.
- The model is used to match found documents.
- The user has a concrete task.
- The task is expressed by an information demand.
- The demand is matched with found documents.
- The user reads and evaluates the documents.
- If necessary, the model is modified.
Findings: IF vs. IR
from the system's point of view

- IF and IR are structurally similar even though they are different tasks.

Document Representation Model

- Computer need formal, computable models

- Document Representation Models define...
  - ...how documents are represented
  - ...how user profiles / queries have to be specified
  - ...the similarity between
    - two documents or
    - a document and a query

- Document Representation Models are in the focus of this lecture!
Simple example for a Document Representation Model

- Documents are represented as a set of words
  - “Hello World” => \(d_1 = \{\text{Hello, World}\}\)
  - “My house is my castle” => \(d_2 = \{\text{My, house, is, castle}\}\)
  - “The world is my house” => \(d_3 = \{\text{The, world, is, my, house}\}\)

- a simple similarity definition
  \[\text{sim}(d_x, d_y) = |d_x \cap d_y|\]

  => \(\text{sim}(d_1, d_2) = 0\),
  => \(\text{sim}(d_1, d_3) = 0\) (in case World ≠ world) otherwise 1
  => \(\text{sim}(d_2, d_3) = 2\) (in case My ≠ my) otherwise 3

(User) Interaction Model

- Defines how user interaction is handled, e.g.
  - No interaction beyond query/profile deployment
  - Query Expansion
  - User Relevance Feedback
  - Automatic Local/Global Analysis
  - Adaptive approaches for profile optimization
  - etc.

- Options of the user interaction model depend on the chosen document representation model
Why is Linguistics important?

• IF&IR Systems are designed to process natural language documents.
• Natural languages are
  – complex (grammar/syntax, semantic, pragmatics)
  – expressive (nearly everything can be expressed)
  – ambiguous (context sensitiveness)
• Keeping linguistics in mind helps to circumvent pitfalls on the way to IF&IR.
Phonology

• Phonology describes how...
  – sounds are related to a language, and their influence on
    • Morphology
    • Syntax
    • Semantics
  – sounds are created (=> biological aspect)
    • E.g: Which sounds are possible at all?
  – sounds have to be interpreted (voice recognition)
• Assumption: all processed documents are available in written form
  => Phonology has inferior impact on this lecture

Morphology

• Different forms of words are existing
  – Engl. dog: dog, dogs
  – Germ. dog: Hund, Hunde, Hunden, Hundes
• Words may be a start point for new words
  – Engl. sub + marine => submarine
  – Engl. master + mind => mastermind
  – Engl. drink => drinkable
  – Germ. Haus + Hund => Haushund
  – Germ. Hund => Hündchen
Words and Word Forms / Flexion

• A word is a unit of spoken or written language
  – In most languages a word is marked in text by spaces
  – A word may have several different (grammatical) word forms, they can depend on case, tempus, number, etc.

• Samples
  – Engl. word “house”, word form: “houses”
  – Engl. word “go”, word forms: “goes”, “went”
  – Germ. word “gehen”, word forms: “geht”, “ging”
• The creation of different (grammatical) word forms is also known as Flexion

Composition

• Creation of a new word (compounds) from other words
• English
  – doghouse
  – mastermind
  – submarine
• German
  – Hausmeister
  – Wolkenkratzer
  – Donauschifffahrtsgesellschaft
Derivation

- Modification of an existing word by the use of some linguistic rules
- English
  - Adjective-to-noun: slow => slowness
  - Noun-to-verb: glory => glorify
  - Verb-to-adjective: drink => drinkable
- German
  - Minimization: Haus => Häuschen
  - Minimization: Zwerg => Zwerglein
  - Adjective-to-noun: langsam => Langsamkeit

Overview on Morphology

Words
- dog
- doghouse
- house
- govern
- governance

Morphology
- Flexion
- Composition
- Flexion
- Flexion
- Derivation
- Flexion

Word Forms
- doghouse
doghouses
house
houses
govern
governed
governance
Stemming / Normalization

- Process to gain
  - **strong stemming**: stem of a word
  - **weak stemming**: basic form (e.g. nominative singular) of a word

- Stemming is often used in context of IF&IR
  - Reduces variability of words
  - Has often positive influence on IF&IR quality

- Samples

Simple example for influence of stemming on IF&IR

- Docs are repres. as a set of words (**without** stemming)
  - “my dog is nice” => \( d_1 = \{ \text{my, dog, is, nice} \} \)
  - “dogs are nice” => \( d_2 = \{ \text{dogs, are, nice} \} \)
  - “cats are lazy” => \( d_3 = \{ \text{cats, are, lazy} \} \)

- Docs are repres. as a set of words (**with** stemming)
  - “my dog is nice” => \( d_4 = \{ \text{my, dog, is, nice} \} \)
  - “dogs are nice” => \( d_5 = \{ \text{dog, is, nice} \} \)
  - “cats are lazy” => \( d_6 = \{ \text{cat, is, lazy} \} \)

- Similarities
  \[
  \begin{align*}
  \text{sim}(d_1, d_2) &= |d_1 \cap d_2| \\
  \text{sim}(d_1, d_2) &= 1 \\
  \text{sim}(d_1, d_3) &= 0 \\
  \text{sim}(d_2, d_3) &= 1 \\
  \text{sim}(d_4, d_5) &= 3 \\
  \text{sim}(d_4, d_6) &= 1 \\
  \text{sim}(d_5, d_6) &= 1
  \end{align*}
  \]
Lexicon based Stemming

<table>
<thead>
<tr>
<th>Word Form</th>
<th>Stem</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>dog</td>
</tr>
<tr>
<td>dogs</td>
<td>dog</td>
</tr>
<tr>
<td>mouse</td>
<td>mouse</td>
</tr>
<tr>
<td>mice</td>
<td>mouse</td>
</tr>
<tr>
<td>run</td>
<td>run</td>
</tr>
<tr>
<td>runs</td>
<td>run</td>
</tr>
<tr>
<td>ran</td>
<td>run</td>
</tr>
<tr>
<td>go</td>
<td>go</td>
</tr>
<tr>
<td>goes</td>
<td>go</td>
</tr>
<tr>
<td>went</td>
<td>go</td>
</tr>
<tr>
<td>gone</td>
<td>go</td>
</tr>
</tbody>
</table>

• Use a lexicon database for stemming
• Pros
  – Algorithm is easy to implement
  – Can handle (in theory) all possible Flexion mechanisms (including irregular ones)
  – Algorithm is language independent
• Cons
  – Database has to be created manually, and is language depended
  – Database tends to be very large (especially for languages like German)
  – No stemming for unknown words
Algorithm based Stemming

String stemming(String word) {
    if(word.endsWith("s"))
        word.removeLastChar();
    if(word.endsWith("e"))
        word.removeLastChar();
    if(word.endsWith("n"))
        word.removeLastChar();
    return word;
}

Algorithm based Stemming

String stemming(String word) {
    if(word.endsWith("s"))
        word.removeLastChar();
    if(word.endsWith("e"))
        word.removeLastChar();
    if(word.endsWith("n"))
        word.removeLastChar();
    return word;
}
Algorithm based Stemming

String stemming(String word) {
    if (word.endsWith("s"))
        word.removeLastChar();
    if (word.endsWith("e"))
        word.removeLastChar();
    if (word.endsWith("n"))
        word.removeLastChar();
    return word;
}
Algorithm based Stemming

String stemming(String word) {
    if(word.endsWith("s"))
        word:lastChar();
    if(word.endsWith("e"))
        word.removeLastChar();
    if(word.endsWith("n"))
        word.removeLastChar();
    return word;
}

© Dominik Kuropka 2007 1-35

Algorithm based Stemming

String stemming(String word) {
    if(word.endsWith("s"))
        word.removeLastChar();
    if(word.endsWith("e"))
        word.removeLastChar();
    if(word.endsWith("n"))
        word.removeLastChar();
    return word;
}

© Dominik Kuropka 2007 1-36
Algorithm based Stemming

String stemming(String word) {
    if (word.endsWith("s"))
        word.removeLastChar();
    if (word.endsWith("e"))
        word.removeLastChar();
    if (word.endsWith("n"))
        word.removeLastChar();
    return word;
}

Algorithm based Stemming

• Use an algorithm for stemming
• Pros
  – Usually low memory consumption and fast
  – Stemming of unknown/unforeseen words is possible
    (Composition and Derivation)
• Cons
  – Design and implementation of algorithm can be tricky
  – Problems of Over- and Under-Stemming may occur
    especially for irregular Flexion mechanisms
  – Algorithm is language dependent
Combined Stemming Approaches

```java
String stemming(String word) {
    if(lexicon.contains(word))
        word = lexicon.getStem(word);
    else if(word.endsWith("s"))
        word.removeLastChar();
    return word;
}
```

<table>
<thead>
<tr>
<th>Word Form</th>
<th>Stem</th>
</tr>
</thead>
<tbody>
<tr>
<td>mouse</td>
<td>mouse</td>
</tr>
<tr>
<td>mice</td>
<td>mouse</td>
</tr>
<tr>
<td>ran</td>
<td>run</td>
</tr>
<tr>
<td>go</td>
<td>go</td>
</tr>
<tr>
<td>goes</td>
<td>go</td>
</tr>
<tr>
<td>went</td>
<td>go</td>
</tr>
<tr>
<td>gone</td>
<td>go</td>
</tr>
</tbody>
</table>

Pros
- Usually lower memory consumption and faster than pure lexicon based approaches
- Probability of Over- and Under-Stemming lower than for pure algorithm based approaches
- Design and implementation of algorithm usually easier than for pure algorithm based approaches
- Stemming of unknown/unforeseen words is possible (Composition and Derivation)

Cons
- Algorithm and database are language dependent
- Database has to be created manually (but is smaller than for pure lexicon based approaches)
Common Stemming Approaches

• English
  – Algorithm based
    • Successor Variety Stemmer
    • n-Gram Stemmer
    • Affix Removal Stemmer (especially PORTER)

• German
  – Lexicons
    • CISLEX
    • Wortschatz-Lexicon
  – Combined
    • Morphy

Next Lecture

• Syntax
  – Overview on grammars and syntax analysis

• Semantics
  – Especially lexical semantics

• Pragmatics
  – Short overview

• Ontologies
  – Definition, classification and potential for IF&IR