Vybrané aktuální projekty Centra ZPJ

Jan Rygl, Zuzana Nevěřilová

E-mail: xrygl@fi.muni.cz,xpopelk@fi.muni.cz
http://nlp.fi.muni.cz/poc_lingv/

Obsah:

Stylometry

Stylometry is the application of the study of linguistic style.

Study of linguistic style:

- Find out text features.
- Define author's writeprint.

Applications:

- Define the author (person, nationality, age group, . . .).
- Filter out text features not usuable by selected application.

Examples of application:

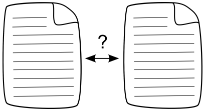
- **Authorship recognition**
  - Legal documents (verify the author of last will)
  - False reviews (cluster accounts by real authors)
  - Public security (find authors of anonymous illegal documents and threats)
  - School essays authorship verification (co-authorship)
  - Supportive authentication, biometrics (e-learning)
- **Age** detection (pedophile recognition on children web sites).
- author **mother language** prediction (public security).
- **Mental disease** symptons detection (health prevention)
- **HR** applications (find out personal traits from text)
- **Automatic translation** recognition.

Stylometry analysis techniques

1. ideological and thematic analysis
   historical documents, literature
2. documentary and factual evidence
   inquisition in the Middle Ages, libraries
3. language and stylistic analysis –
   1. manual (legal, public security and literary applications)
   2. semi-automatic (same as above)
   3. automatic (false reviews and generally all online stylometry applications)

Stylometry Verification

Definition



- decide if two documents were written by the same author category (1v1)
- decide if a document was written by the signed author category (1vN)

Examples

- The Shakespeare authorship question
- The verification of wills

Authorship Verification

The Shakespeare authorship question

Mendenhall, T. C. 1887.
The Characteristic Curves of Composition. Science Vol 9: 237–49.

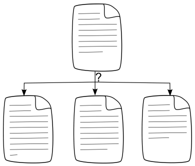
- The first algorithmic analysis
- Calculating and comparing histograms of word lengths

Oxford, Bacon
Derby, Marlowe

http://en.wikipedia.org/wiki/File:ShakespeareCandidates1.jpg

Stylometry Attribution

Definition



- find out an author category of a document
- candidate authors' categories can be known (e.g. age groups, healthy/unhealthy person)
- problems solving unknown candidate authors's categories are hard (e.g. online authorship, all clustering tasks)

Examples

- Anonymous e-mails

Authorship Attribution

Judiciary

- The police falsify testimonies
  Morton, A. Q. Word Detective Proves the Bard wasn't Bacon. Observer, 1976.
- Evidence in courts of law in Britain, U.S., Australia
- Expert analysis of courtroom discourse, e.g. testing "patterns of deceit" hypotheses

NLP Centre stylometry research

Authorship Recognition Tool

- Ministry of the Interior of CR within the project `VF20102014003`
- Best security research award by Minister of the Interior

Small projects (bachelor and diploma theses, papers)

- detection of automatic translation, gender detection, . . .

TextMiner

- multilingual stylometry tool + many other features not related to stylometry
- authorship, mother language, age, gender, social group detection

Obsah

Computional stylometry

Updated definition

techniques that allow us to find out information about the authors of texts on the basis of an automatic linguistic analysis

Stylometry process steps

1. **data acquisition** – obtain and preprocess data
2. **feature extraction methods** – get features from texts
3. **machine learning** – train and tune classifiers
4. **interpretation of results** – make machine learning reasoning readable by human

Data acquisition – collecting

Free data

- For big languages only
- Enron e-mail corpus
- Blog corpus (*Koppel, M, Effects of Age and Gender on Blogging)*

Manually annotated corpora

1. ÚČNK school essays
2. FI MUNI error corpus

Web crawling

Data acquisition – preprocessing

Tokenization, morphology annotation and desambiguation

- morphological analysis

```
je      byt     k5eAaImIp3nS
spor    spor    k1gInSc1
mezi    mezi    k7c7
Severem sever   k1gInSc7
a       a       k8xC
Jihem   jih     k1gInSc7
<g/>
.       .       kIx.
</s>
<s desamb="1">
Jde     jit     k5eAaImIp3nS
```

Selection of feature extraction methods

Categories

- Morphological
- Syntactic
- Vocabulary
- Other

Analyse problem and select only suitable features. Combine with automatic feature selection techniques (entropy).

Tuning of feature extraction methods

Tuning process

Divide data into three independet sets:

- Tuning set (generate stopwords, part-of-speech n-grams, . . .)
- Training set (train a classifier)
- Test set (evaluate a classifier)

Features examples

Word length statistics

- Count and normalize frequencies of selected word lengths (eg. 1–15 characters)
- Modification: word-length frequencies are influenced by adjacent frequencies in histogram, e.g.: `1: 30%, 2: 70%, 3: 0%` is more similar to `1: 70%, 2: 30%, 3: 0%` than `1: 0%, 2: 60%, 3: 40%`

Sentence length statistics

- Count and normalize frequencies of
  - word per sentence length
  - character per sentence length

Features examples

Stopwords

- Count normalized frequency for each word from stopword list
- Stopword $\sim$ general word, semantic meaning is not important, e.g. prepositions, conjunctions, . . .
- *stopwords* **ten, by, člověk, že** *are the most frequent in selected five texts of Karel Čapek*

Wordclass (bigrams) statistics

- Count and normalize frequencies of wordclasses (wordclass bigrams)
- *verb is followed by noun with the same frequency in selected five texts of Karel Čapek*

Features examples

Morphological tags statistics

- Count and normalize frequencies of selected morphological tags
- *the most consistent frequency has the genus for family and archaic freq in selected five texts of Karel Čapek*

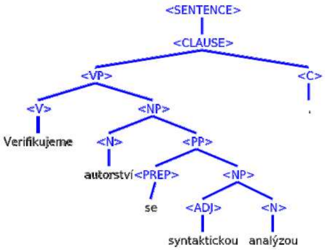
Word repetition

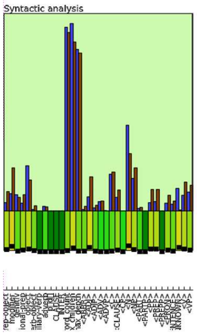
- Analyse which words or wordclasses are frequently repeated through the sentence
- *nouns, verbs and pronous are the most repetetive in selected five texts of Karel Čapek*

Features examples

Syntactic Analysis

- Extract features using SET (Syntactic Engineering Tool)





- *syntactic trees have similar depth in selected five texts of Karel Čapek*

Features examples

Other stylometric features

- typography (number of dots, spaces, emoticons, . . .)
- errors
- vocabulary richness

Features examples

Implementation

```
features = (u'kA', u'kY', u'kI', u'k?', u'k0',
    u'k1', u'k2', u'k3', u'k4', u'k5', u'k6',
    u'k7', u'k8', u'k9')

def document_to_features(self, document):
    """Transform document to tuple of float features.
    @return: tuple of n float feature values, n=|get_features|"""
    """"
    features = np.zeros(self.features_count)
    sentences = self.get_structure(document, mode=u'tag')
    for sentence in sentences:
        for tag in sentence:
            if tag and tag[0] == u'k':
                key = self.tag_to_index.get(tag[:2])
                if key: features[key] += 1.
    total = np.sum(features)
    if total > 0: return features / total
    else: return features
```

Machine learning

Tools

- use **frameworks** over your own implementation (ML is HW consuming and needs to be optimal)
- programming language doesn't matter, but high-level languages can be better (**readability** is important and performance is not affected – ML frameworks use usually C libraries)
- for Python, good choice is **Scikit-learn** (http://scikit-learn.org)

Machine learning tuning

- try different machine learning techniques (Support Vector Machines, Random Forests, Neural Networks)
- use grid search/random search/other heuristic searches to find optimal parameters (use **cross-validation** on train data)
- but **start with the fast and easy to configure** ones (Naive Bayes, Decision Trees)
- **feature selection** (more is not better)
- make experiments **replicable** (use random seed), repeat experiments with different seed to check their performance
- always implement a **baseline** algorithm (random answer, constant answer)

Machine learning tricks

Replace feature values by ranking of feature values

Book:

long coherent text

Blog:

medium-length text

E-mail:

short noisy text

- Different “document conditions” are considered
- Attribution: replace similarity by ranking of the author against other authors
- Verification: select random similar documents from corpus and replace similarity by ranking of the document against these selected documents

Interpretation of results

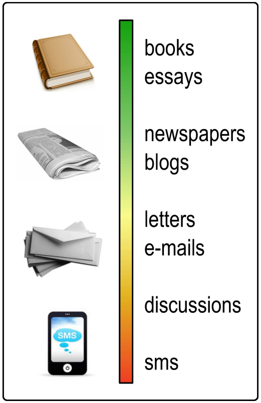
Machine learning readable

Explanation of ML reasoning can be important. We can

1. not to interpret data at all (we can't enforce any consequences)
2. use one classifier per feature category and use feature categories results as a partially human readable solution
3. use ML techniques which can be interpreted:
    - Linear classifiers
      each feature f has weight $w(f)$ and document value $val(f)$,
      $\sum_{f \in F} w(f) * val(f) \geq threshold$
    - Extensions of black box classifiers, for random forests
      https://github.com/janrygl/treeinterpreter
4. use another statistical module not connected to ML at all

Performance (Czech texts)

Balanced accuracy: Current (CS) → Desired (EN)



Verification:

- books, essays: 95 % → 99 %
- blogs, articles: 70 % → 90 %

Attribution (depends on the number of candidates, comparison on blogs):

- up to 4 candidates: 80 % → 95 %
- up to 100 candidates: 40 % → 60 %

Clustering:

- the evaluation metric depends on the scenario (50–60 %)

I want to try it myself

How to start

- Select a problem
- Collect data (gender detection data are easy to find – crawler dating service)
- Preprocess texts (remove HTML, tokenize)
- Write a few feature extraction methods
- Use a ML framework to classify data

I want to try it really quick

Quick start

Style & Identity Recognizer

`https://github.com/janrygl/sir`.

- In development, but functional.
- Contains data from dating services.
- Contains feature extractors.
- Uses free RFTagger for morphology tagging.

Development at FI

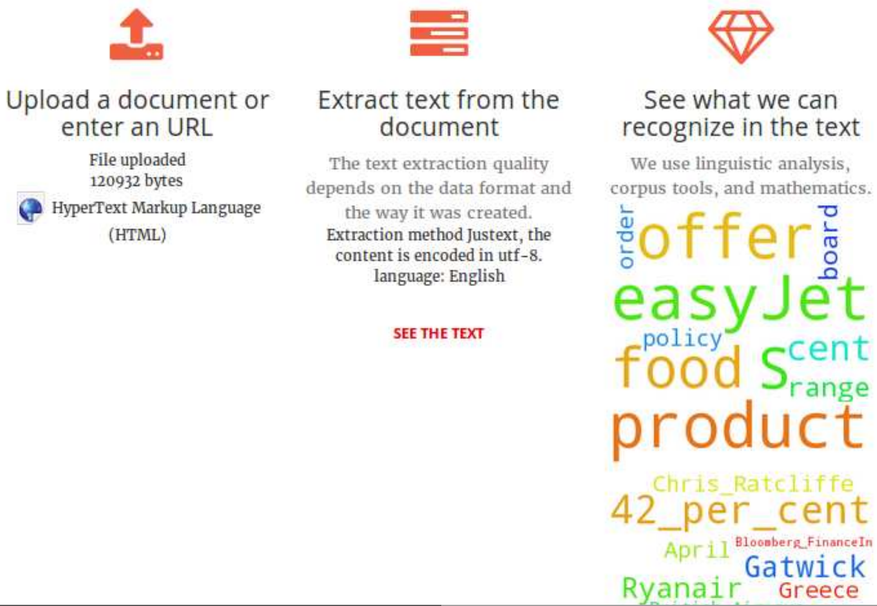
TextMiner

- more languages,
- more feature extractors,
- more machine learning experiments,
- better visualization,
- and much more

Thank you for your attention

Savage Chickens
by Doug Savage
...AND SO, MY STUDY PROVES ONCE AND FOR ALL THAT WILLIAM SHAKESPEARE WROTE SHAKESPEARE'S PLAYS
GIVE ME SOME MORE FUNDING AND I'LL PROVE THAT I WROTE MY STUDY
©2011 BY DOUG SAVAGE
www.savagechickens.com

TextPerience: Experience the Text



Upload a document or enter an URL

File uploaded
120932 bytes

HyperText Markup Language (HTML)

Extract text from the document

The text extraction quality depends on the data format and the way it was created.
Extraction method Justext, the content is encoded in utf-8.
language: English

SEE THE TEXT

See what we can recognize in the text

We use linguistic analysis, corpus tools, and mathematics.

TextPerience: Experience the Text

- a demo application
- complete processing pipeline from data to knowledge:
  1. convert an uploaded file to text
  2. detect language
  3. apply keyword search
  4. recognize named entities
  5. perform Wikipedia search of the keywords
  6. evaluate the results
- so far, keywords in 12 languages (cs, sk, en, de, fr, it, es, vi, pl, hu, da, ru)
- so far, named entities in 5 languages (cs, en, es, de, ru)

Outline

TextPerience I: Extract Texts from Files

After a successful MIME type guessing . . .

- use Apache Tika for Office-like texts and PDFs
- use JusText for extracting text from web pages, Pomikálek (2011)
- take plain texts as they are

many other file types:

- tables (Excel, TSV, CSV)
- presentations
- images (OCR)
- source codes (plain text with a particular extension)
- videos (subtitles)
- . . .

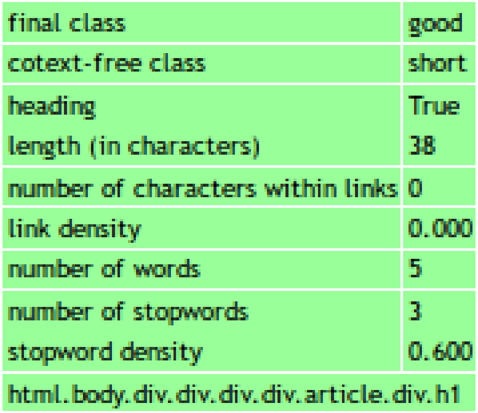
TextPerience I: Extract Texts from Files

"repair" extracted texts:

- distorted diacritics in PDFs
- headers, footers, page numbers
- hyphenation

TextPerience I: Extract Texts from Files

selecting the right content from web pages:
JusText parameters vs. customization



final class	good
cotext-free class	short
heading length (in characters)	True 38
number of characters within links	0
link density	0.000
number of words	5
number of stopwords stopword density	3 0.600
html.body.div.div.div.div.article.div.h1	

TextPerience II: Detect Language(s)

the `langid` module, Lui and Baldwin (2012)

- trained for $\approx$ 90 languages, excl. cestina
- Naive-Bayes classifier on letter n-grams
- good for longer texts
- noise reduction needed (HTML tags and entities, URLs, non-words etc.)
- how about multi-lingual documents?

TextPerience III: Keyword Search

“language independent” way: TF-IDF, Wikipedia (2016)

- tokenize the text
- detect POS
- assume keywords to be only nouns
- lemmatize nouns
- compute TF-IDF on lemmata
- what about keyphrases?

TextPerience III: Keyword Search

Tagging and Lemmatization: a very simplistic approach

- find a word form in corpus (limit to 10 occurrences)
- save the most frequent lemma and tag
- works for all POS-annotated corpora with lemmatization
- fast and “independent” but not very precise

TextPerience III: Keyword Search

TF-IDF on noun lemmata

- corpora of different sizes for different languages
- precomputed d_fs in our corpora
- what about words not present in our corpora?

TextPerience IV: Named Entity Recognition

Stanford NER with different models, Finkel et al. (2005)

- ready to use models for es, en, de
- training data for cs and ru (see IA161) → new models
- very different model parameters for e.g. English and German
- apply models for a language on close languages (e.g. es model on fr)
- for training models, we need NE annotated corpus

TextPerience V: Wikipedia Search

use of API MediaWiki (2015)

- for each keyword/NE try to get the most relevant Wikipedia article
- no keyword scoring employed
- no graph-based scoring employed
- no generalization methods
- different sizes of Wikipedias in different languages

TextPerience VI: Evaluation

users can click on the evaluation icons ✓ ✗

- 355 files
- 1451 evaluations: 1045 entities/keywords correct, 406 incorrect
- no learning from evaluations

Conclusion and Further Work

1. convert an uploaded file to text

   repair texts from PDFs

2. detect language

   detect multi-lingual documents, noise reduction

3. apply keyword search

   better ways to fast lemmatization
   keyword search in multi-lingual documents

   keyphrase search (what is a phrase?)

4. recognize named entities

   evaluation of existing models

   evaluation of using existing models on close languages

5. perform Wikipedia search of the keywords

   keyword weighting
   graph algorithms for disambiguation

   generalization methods (Wikipedia categories)

6. evaluate the results

   learning methods

Try It

Try TextPerience
`http://nlp.fi.muni.cz/projects/textperience`

References I

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lui, M. and Baldwin, T. (2012). Langid.py: An off-the-shelf language identification tool. In Proceedings of the ACL 2012 System Demonstrations, ACL '12, pages 25–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

MediaWiki (2015). Api:search — mediawiki, the free wiki engine. [Online; accessed 16-May-2016].

Pomikálek, J. (2011). justext.

Wikipedia (2016). Tf-idf — wikipedia, the free encyclopedia. [Online; accessed 16-May-2016].