Patent Analysis Experiment

Mengke Hu, David Cinciruk

ASPITRG Group, ECE Department
Drexel University

mengke.hu@gmail.com,davidcinciruk@gmail.com

August 6, 2014

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

What are Patents?

- A patent is a set of exclusive rights granted by a sovereign state to an inventor or assignee for a limited period of time in exchange for detailed public disclosure of an invention
- Need to satisfy a novelty requirement and a non-obviousness requirement
- Three Types of Patents
  - Utility
  - Design
  - Plant

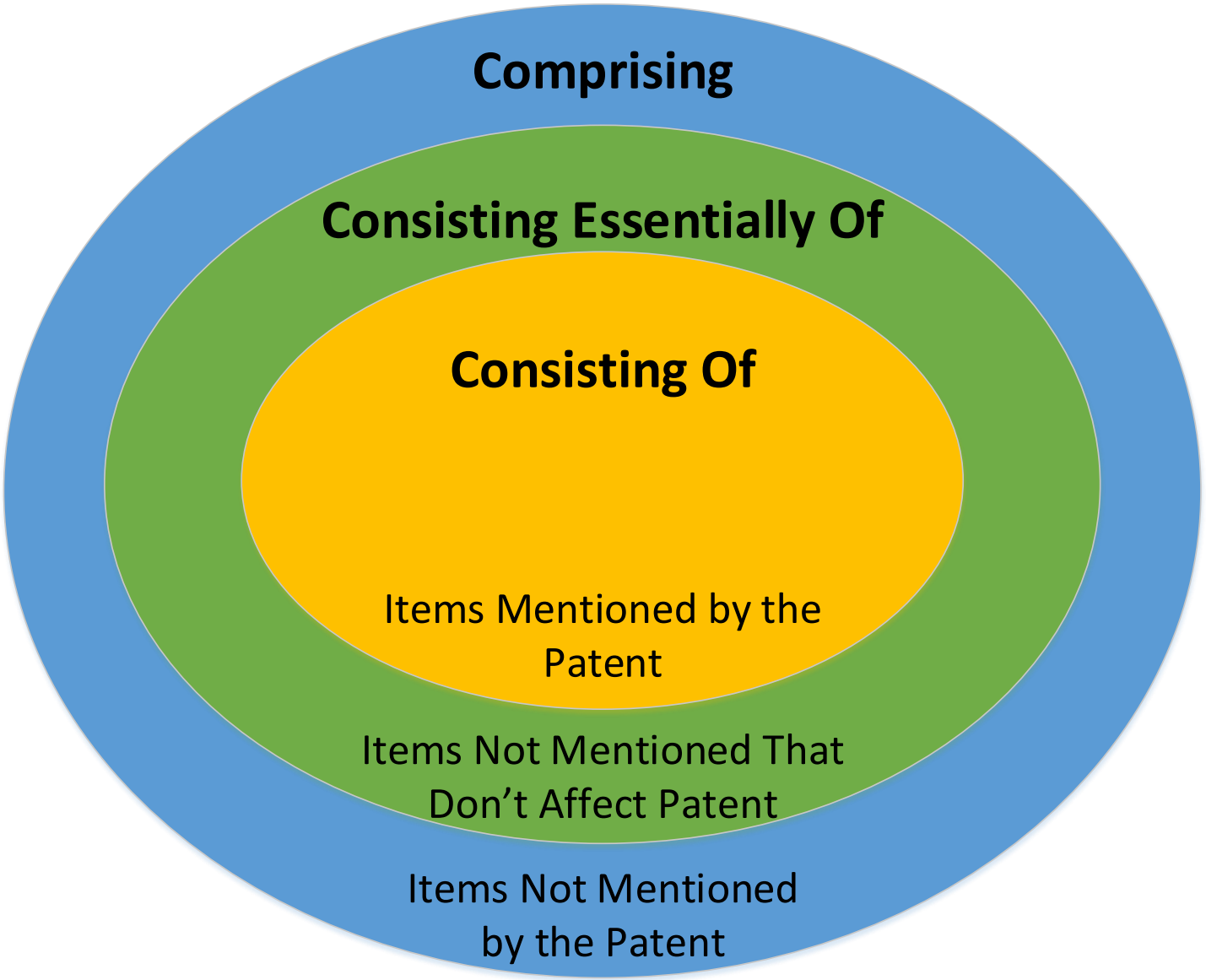
Utility Patents

- The "typical patent"
- Contains both legal and technical descriptions of the invention
  - **Claims (Legal Description)** - description of the invention in the broadest possible way to avoid infringing on currently developed inventions
  - **Specifications (Technical Description)** - description of the invention the way the inventor sees it, may include some more details about how it will be constructed and what it will be used for as well as sketches of the invention. Descriptions in this section help to define terms in claims

Structures of Patent Claims

- Patent claims are written with very specific grammars and a limited vocabulary
- Indendent claims typically contain the invention in question as the subject, a verb (usually either comprising or consisting of), and many object phrases describing the items that make up the invention
- Dependent claims reference a term from a specific patent claim (referencing that claim) and narrow down the term to an iteration that the inventor hoped to produce
- The items that make up the invention are referred to with unique terms and adjectives describing them. In dependent claims, these items are referred to with the exact terms and adjectives used to describe them as before
- Patent claim structure is consistent between different patents
- Systems could be set up to exploit this single unified structure

Structures of Patent Claims - Comprising/Consisting of



Claims of Patent #8,669,676

1. A system, comprising: a source resonator coupled to an energy source generating a field having magnetic material; and a second resonator located a variable distance from the source resonator having magnetic material and not connected by any wire or shared magnetic material to the source resonator, wherein the source resonator and the second resonator are coupled to provide near-field wireless energy transfer among the source resonator and the second resonator; wherein the field of at least one of the source resonator and the second resonator is shaped using magnetic materials to increase the coupling factor among the resonators.
2. The system of claim 1, wherein the distance between the resonators is greater than 5 cm.
3. The system of claim 1, wherein the distance between the resonators is greater than 10 cm.

US Patent Classification System (USPC)

- One of the many different classification systems used by the USPTO
- Each patent is listed under one (or several) of many different categories
- Each category is also broken down into subcategories
- Represented by numbers for both class and subclass
- Example: 800/29
  - Class 800: Multicelcullar living organisms and unmodified parts thereof and related processes
  - Subclass 29: Higher plant, seedling, plant seed, or plant part (i.e., angiosperms or gymnosperms):

Our Experiments

- Patent Classification using SVMs - is it part of one class or another?
- Work comes with comparing the most basic features (based off just word counting) and advanced structures that take the sentence structure into account
- Work for the latter hasn't been completed yet, but features have been generated and will be talked about

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Information Retrieval and Information Science

1. Information Retrieval (IR) deals with the representation, storage, organization of and access to information items. (Baeza-Yates and Ribeiro-Neto 2004)
2. Information Science is about "gathering organizing, storing, retrieving and dissemination of information" (Bates 1999)

Traditional IR Experiment

- Document Collection
- A set of queries
- Relevance judgement
- evaluation metrics (precision and recall)

Basic Assumption of IR

1. Collection: Fixed set of documents
2. Goal: Retrieve documents with information that is **relevant** to user's information need and helps the user complete a task

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Document Preprocessing Techniques

Tokenization

- Input: "Friends, Romans and Countrymen"
- Output(Tokens):
  - Friends
  - Romans
  - Countrymen
- Term Normalization: Deleting periods: U.S.A$->$ USA; deleting hyphens: anti-bio $->$ antibio
- Stemming: Variation of the words: automates, automatic, automation -¿ automat
- Deliminator: . : * etc.
- Stop Words: a, the, and, to, be
- Sometimes stop words cannot be removed: song name "Let it be"

n-gram tokenization

$n = 2$ gram example

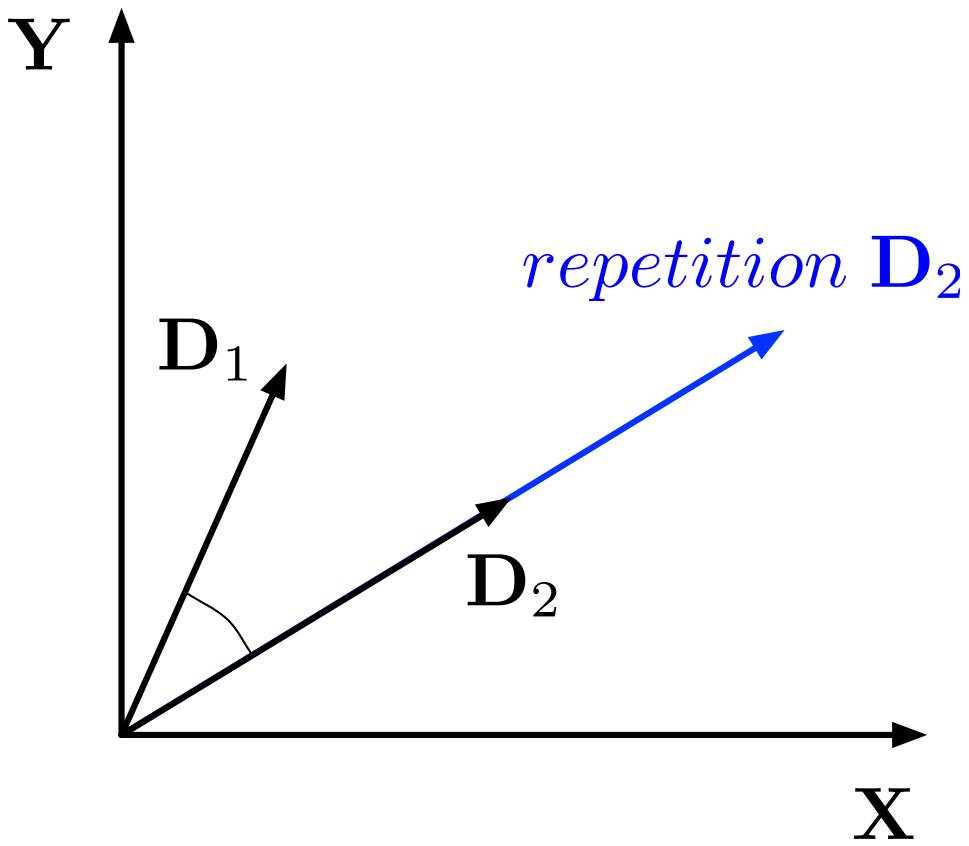
- Input: "Friends, Romans and Countrymen"
- Tokens:
  - friend roman
  - roman and
  - and countrymen

Compare with bag of words, n-gram can capture more semantic meaning from documents.

Document Representation in IR

Vector Space Model

Document is represented by the angel of a vector (amplitude is useless)



Vector Space Model

TF-IDF weighting

- Term Frequency (frequency of term i in document j):

$$tf_{i,j} = \begin{cases} 1 + \log f_{i,j} & \text{if } f_{i,j} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

- Inverse Document Frequency of term i

$$IDF_i = \log \frac{N}{n_i}$$

- tf-idf weighting

$$w_{i,j} = \begin{cases} (1 + \log f_{i,j}) \times \log \frac{N}{n_i} = tf_{i,j} \times IDF_i & \text{if } f_{i,j} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Vector Space Model

TF-IDF weighting Properties

- $TF_i = 1 + \log \sum_{j=1}^{N} f_{i,j}$ decreases with respect to the term ranking
- $IDF_i = \log \left(\frac{N}{n_i} \right)$ increases with respect to the term ranking
- Common terms such as stop words and rare terms are not of great value for weight ranking

Document Representation in Experiment

- Tokenization: Obtain the basis for the vector
- Calculate the $TF - IDF$ weighting for each tokens
- Form the vector representation for each document

Outline

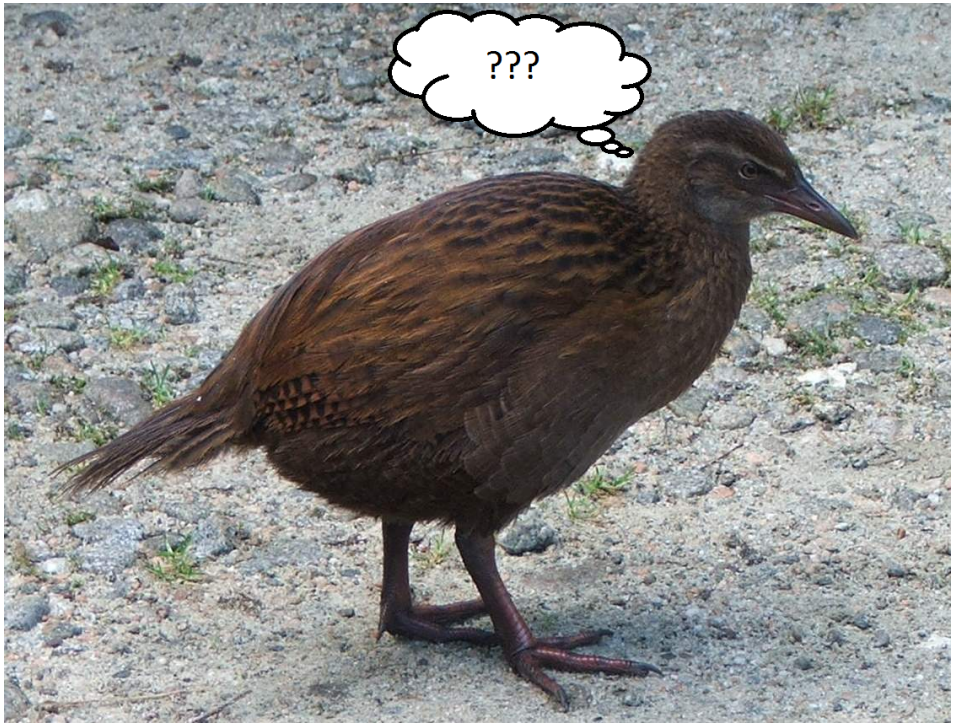
1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

What is Weka?

- Software package created by the Machine Learning Group at the University of Waikato
- Collection of Machine Learning Algorithms for data mining tasks
- Available both as a GUI executable as well as functions in Java



This is also a Weka but not the kind we're talking about

Our Usage of Weka

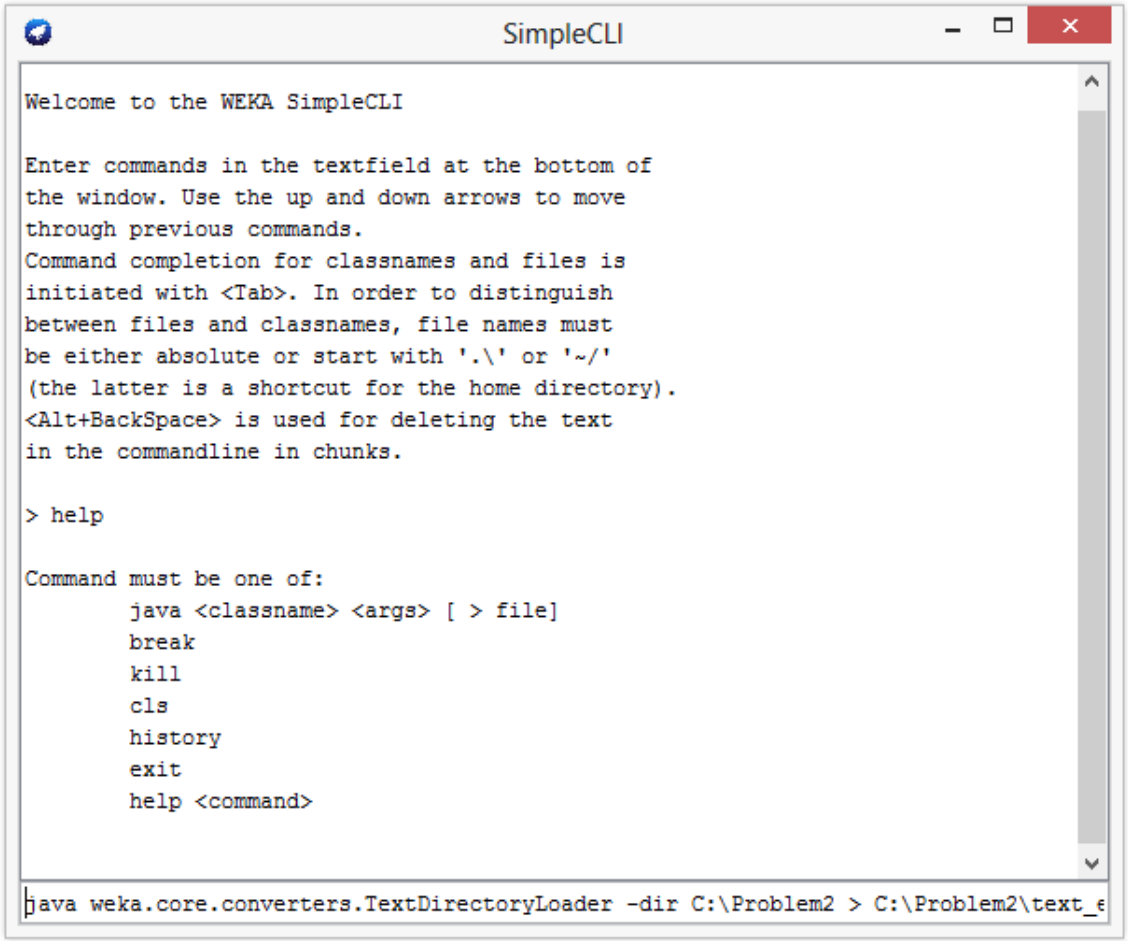
- Can extract bag of words and n-gram vector space models of a large collection of documents automatically
- Can then run SVMs or other classifiers on the data (for some SVM algorithms even Multiclass is allowed)

Java Functions

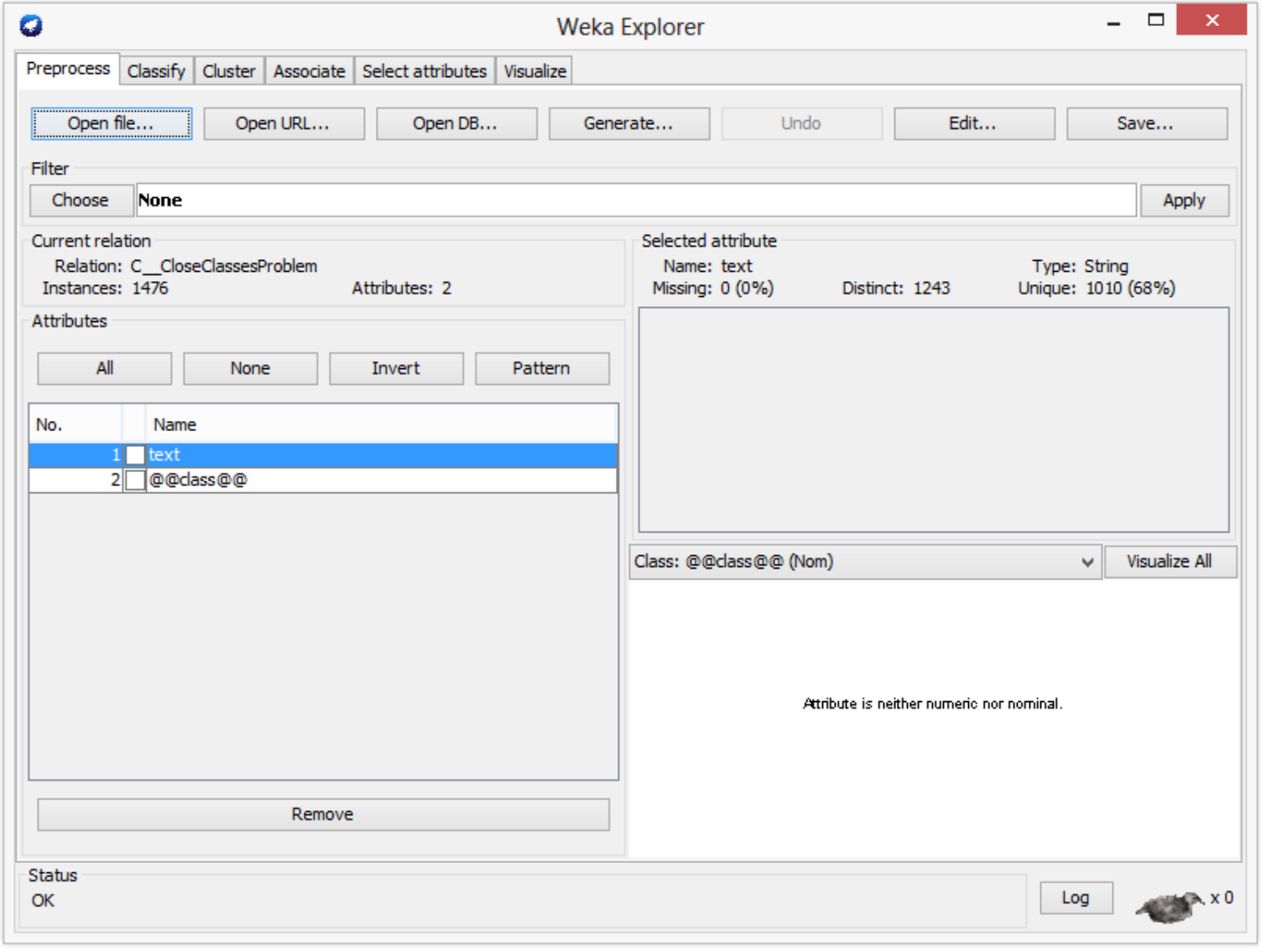
```
StringToWordVector filter = new StringToWordVector();

//*the tokenizer algorithm to use
WordTokenizer wt = new WordTokenizer();
wt.setDelimiters("(.,:;'\"?!@#$%    ^&*{} |[]\\<>/`~1234567890-=_+)\n");
filter.setTokenizer(wt);
//*Sets whether output instances contain 0 or 1 indicating word presence, c
filter.setOutputWordCounts(true);
//*Sets whether if the word frequencies should be transformed into log(1+f
filter.setTFTransform(true);
//*Sets whether if the word frequencies in a document should be transformed
//*where fij is the frequency of word i in document(instance) j.
filter.setIDFTransform(true);
//*Sets whether if the tokens are to be downcased or not.
filter.setLowerCaseTokens(true);
//*the stemming algorithm to use, null means no stemming at all (i.e., the
LovinsStemmer stem = new LovinsStemmer();
filter.setStemmer(stem);
//*Sets whether if the words that are on a stoplist are to be ignored (The
filter.setUseStoplist(true);
```

GUI Version - Home Screen

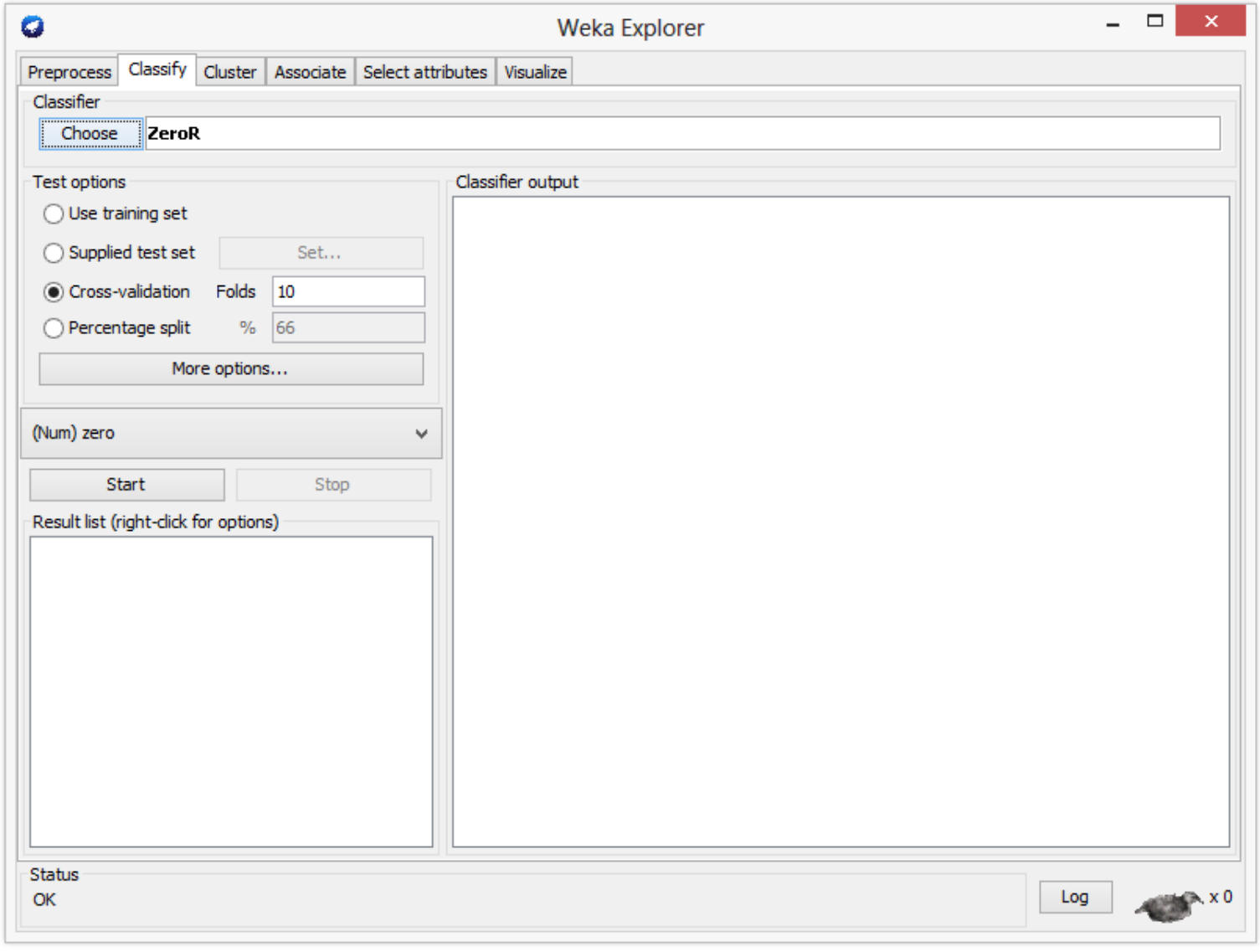
GUI Version - Command Line Interface



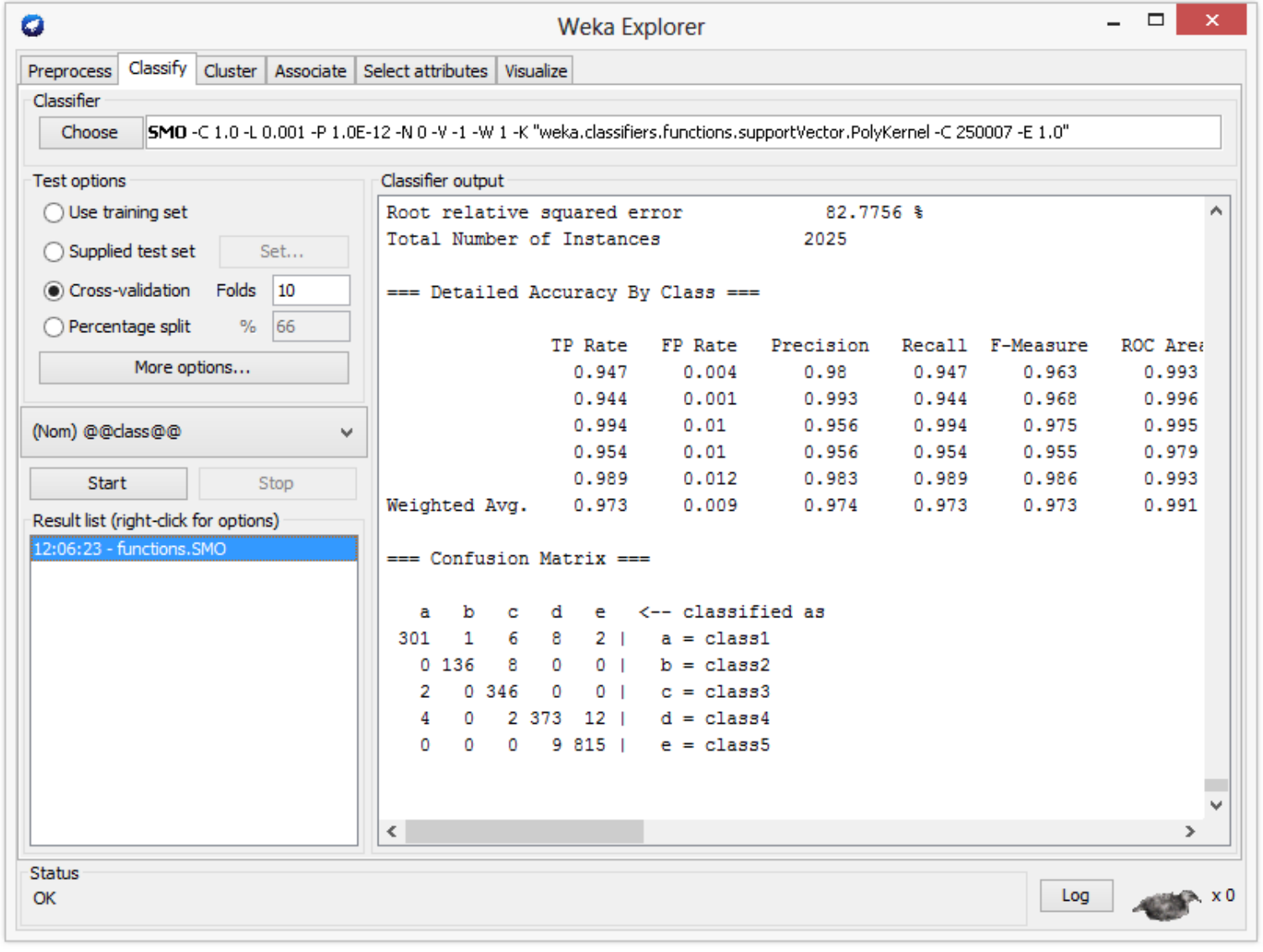
GUI Version - Explorer



GUI Version - Classifier



GUI Version - Running Experiments



Our Experiments

- Two main experiments in classification:
  - Classifying patents into one of two classes with many overlapping patents (and those overlapping patents removed)
  - Classifying patents into one of five distinct categories
- Experiments run using Weka for SVM training with cross-validation for SVM testing using the Pegasos Algorithm
- Features include:
  - Bag of Words
  - n-grams (min monograms, max bigrams)
  - n-grams (min monograms, max trigrams)
  - Trigrams exclusively

The Categories

Close Classes Problem

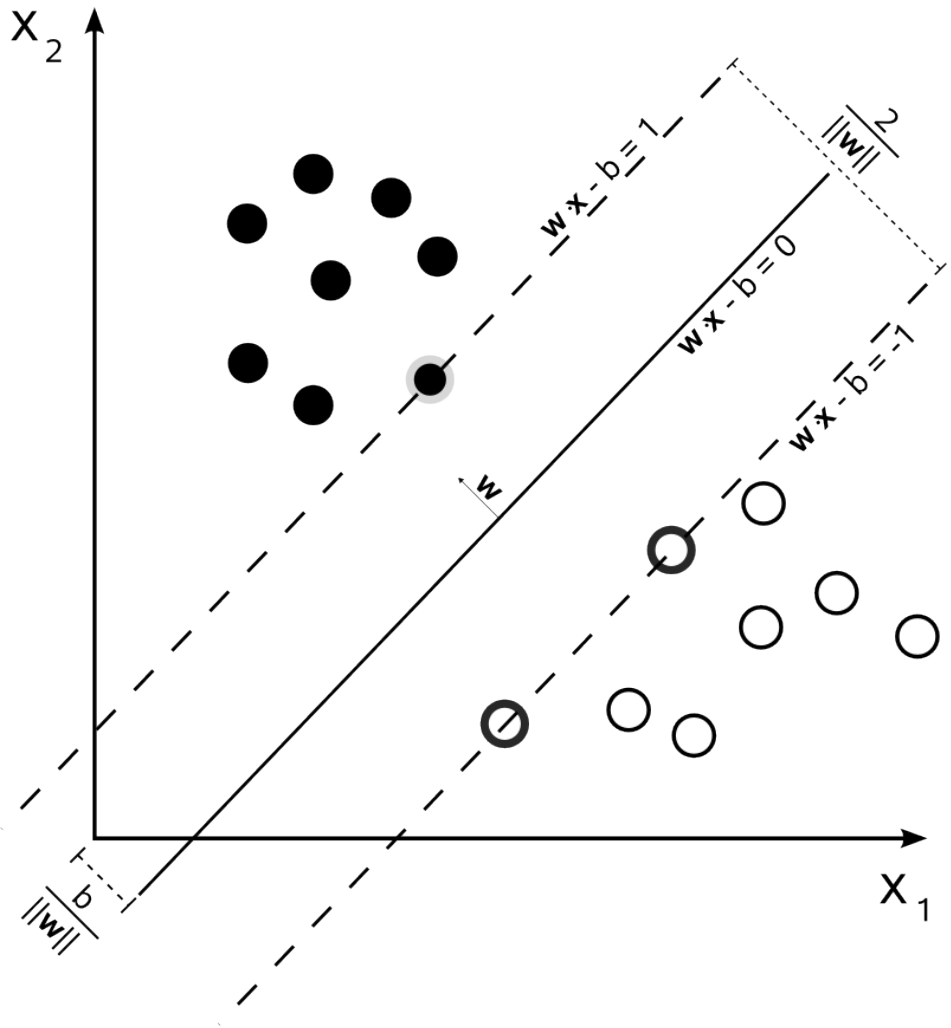
- Both are Class 714 - Error Detection/correction and fault detection/recovery
  - Subclass 776 - For packet or frame multiplexed data
  - Subclass 784 - Reed-Solomon Codes

Five Class Problems

- 99/468 - Food and Beverages: Apparatus / Automatic control or time means
- 229/110 - Envelopes, wrappers, and paperboard boxes / Hexagonal
- 296/29 - Land vehicles: bodies and tops / Corners and joints
- 463/39 - Amusement devices: games / Wireless signal
- 714/776 - Error detection, correction and fault detection, recovery / For packet or frame multiplexed data

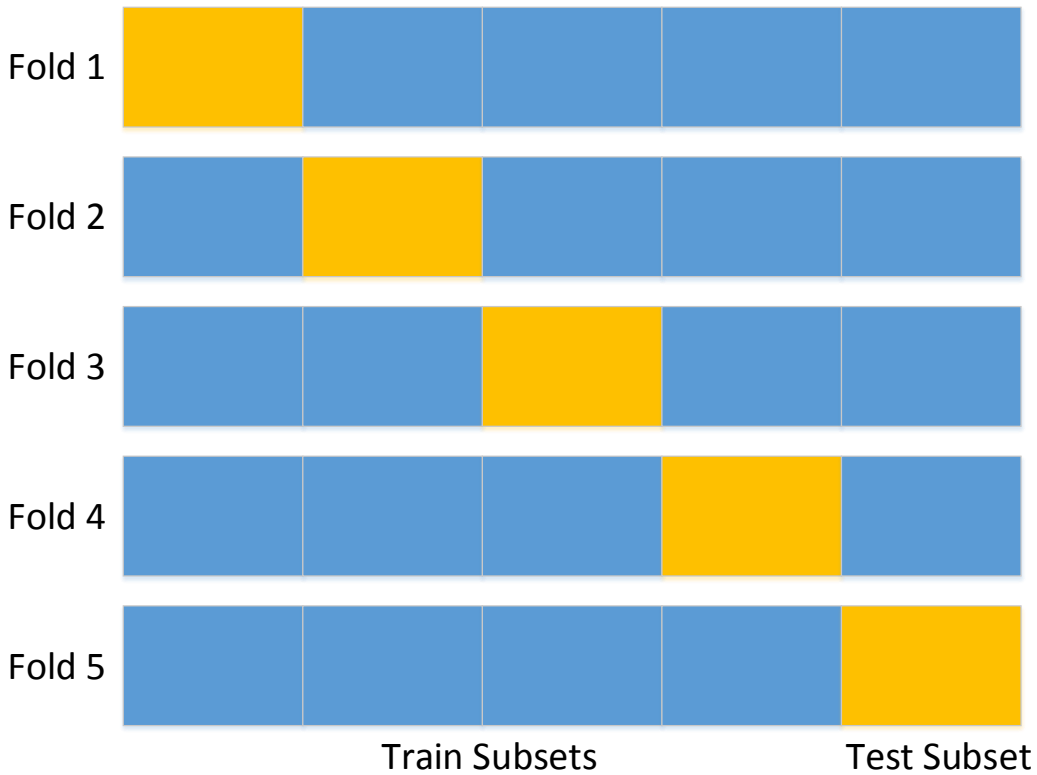
SVM Review

- SVMs attempt to find a hyperplane that maximally separates the data between two classes
- Can separate non-linearly separable data using the kernel trick to project data to higher dimensions
- Forms a convex optimization problem with the data
- For an n-class problem, generally trains n SVMs comparing each class against the rest and chooses the best



Cross Validation

- The data is partitioned into a training and testing subset
- Multiple iterations are performed and results are averaged
- k-fold Cross Validation: original sample is partitioned into k subsamples. One of them is used for testing and the rest are used for training. This is repeated for all k subsamples



SVM Results for Different Experiments

Experiment	Bag of Words	Bigrams (1-2 words)
Close Classes Problems	90.176%	90.447%
Five Classes Problems	97.531%	97.284%

Experiment	Trigrams (1-3 words)	Trigrams (only 3 words)
Close Classes Problems	90.447%	82.385%
Five Classes Problems	97.333%	88.840%

Table : Results for Experiments with Our Patent Databases

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Context Free Grammar

In Computational Theory

- CFG is used for generating strings
- Formal Definition:
  CFG is a $4-$ tuple $G = (T, N, S, R)$, where
  - T is a finite set of terminals
  - N is a finite set of variables (non-terminal)
  - R is a set of substitution rules, where each rule consists of a variable on the left side and a string of variables and terminals at the right side of the arrow. e.g. $A \to 0A1$
  - S is the start variable

Context Free Grammar

Terminology

- A derives B, denoted by $A \to B$, if
  - $A = B$, or
  - $\exists A_1, \cdots, A_k$ with $k \geq 0$ s.t. $A \to A_1 \to \cdots \to A_k \to B$
- For a grammar $G = (T, N, S, R)$, language $L(G) = \{w \in T | S \to w\}$
- Theory: Any regular language can be described by a CFG

Context Free Grammar

In Natural Language Processing:Constituency

Group of words may behave as a single unit or phrase. sentences have parts, some of which appear to have subparts. Theses groupings of words that go together we will call constituents.
Examples: Noun Phrase

- December twenty-sixth
- they
- the reason he is running for president

Constituent Phrases

- the man from Philadelphia Noun Phrase (NP) because the head man is a noun
- extremely clever Adjective Phrase (AP) because the head clever is an adjective
- down the river Prepositional Phrase (PP) because the head down is a preposition
- chase the rabbit Verb Phrase(VP) because the head chase is a verb

Ambiguity in Constituents

1. I hit the man with a cleaver
   - I hit [the man with a cleaver].
   - I hit [the man] with a cleaver
2. You could not go to her party
   - You [could not] go to her party.
   - You could [not go] to her party

Context-free grammar in NLP

CFG is the most common way of modeling constituency

- grammar $G =< T, N, S, R >$ generates a language L
- T is set of terminals (lexicon, words from given sentence)
- N is set of non-terminals (Phrase tagging: eg. NP, VP)
- S is start symbol (one of the nonterminals)
- R is rules/productions of the form $\mathbf{X} \rightarrow \gamma$, where $\mathbf{X}$ is a nonterminal and γ is a sequence of terminals and nonterminals

Context-free grammar example in NLP

$$G =< T, N, S, R >$$

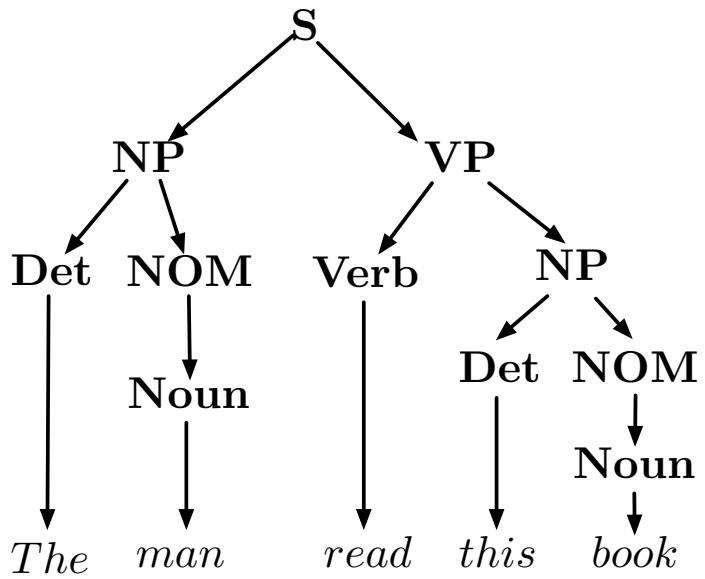
$$T = \{\text{that, this, a, the, man, book, flight, meal, include, read, does}\}$$

$$N = \{S, NP, NOM, VP, Det, Noun, Verb, Aux\}$$

$$\begin{aligned} R = \{ & S \rightarrow NPVP \quad S \rightarrow AuxNPVP \quad S \rightarrow VP \\ & NP \rightarrow Det\ NOM \\ & NOM \rightarrow Noun \quad NOM \rightarrow Noun\ NOM \\ & VP \rightarrow Verb \quad VP \rightarrow VerbNP \\ & Det \rightarrow that|this|a|the \\ & Noun \rightarrow book|flight|meal|man \\ & Verb \rightarrow book|include|read \\ & Aux \rightarrow does\} \end{aligned}$$

Parsing Tree Example

$S \rightarrow NPVP \rightarrow DetNOMVP \rightarrow TheNOMVP$
$\rightarrow TheNounVP \rightarrow The\ man\ VP \rightarrow The\ man\ Verb\ NP$
$\rightarrow The\ man\ read\ NP \rightarrow The\ man\ read\ Det\ NOM \rightarrow$
$The\ man\ read\ this\ NOM \rightarrow The\ man\ read\ this\ Noun \rightarrow$
$The\ man\ read\ this\ book$



Recursion Problem

CFGs capture the recursion problem:

- $PP \rightarrow PrepNP$
- $NP \rightarrow NounPP$
- [S The mailman ate his [NP lunch [PP with his friend [PP from the cleaning staff [PP of the building [PP at the intersection [PP on the north end [PP of town]]]]]]]
- The recursion will stop when the parsing process reaches the bracket.

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Penn Treebank Project

- The project annotates naturally-occurring text for linguistic structure
- Project forms parse tree structures for natural text
- Corpra available include The Wall Street Journal, The Brown Corpus, Switchboard, and ATIS

Example of a Penn Treebank sentence

This sentence is an example for the Penn Treebank Format

```
(ROOT
  (S
    (NP (DT This) (NN sentence))
    (VP (VBZ is)
      (NP
        (NP (DT an) (NN example))
        (PP (IN for)
          (NP (DT the) (NNP Penn) (NNP Treebank) (NNP Format)))))))
```

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

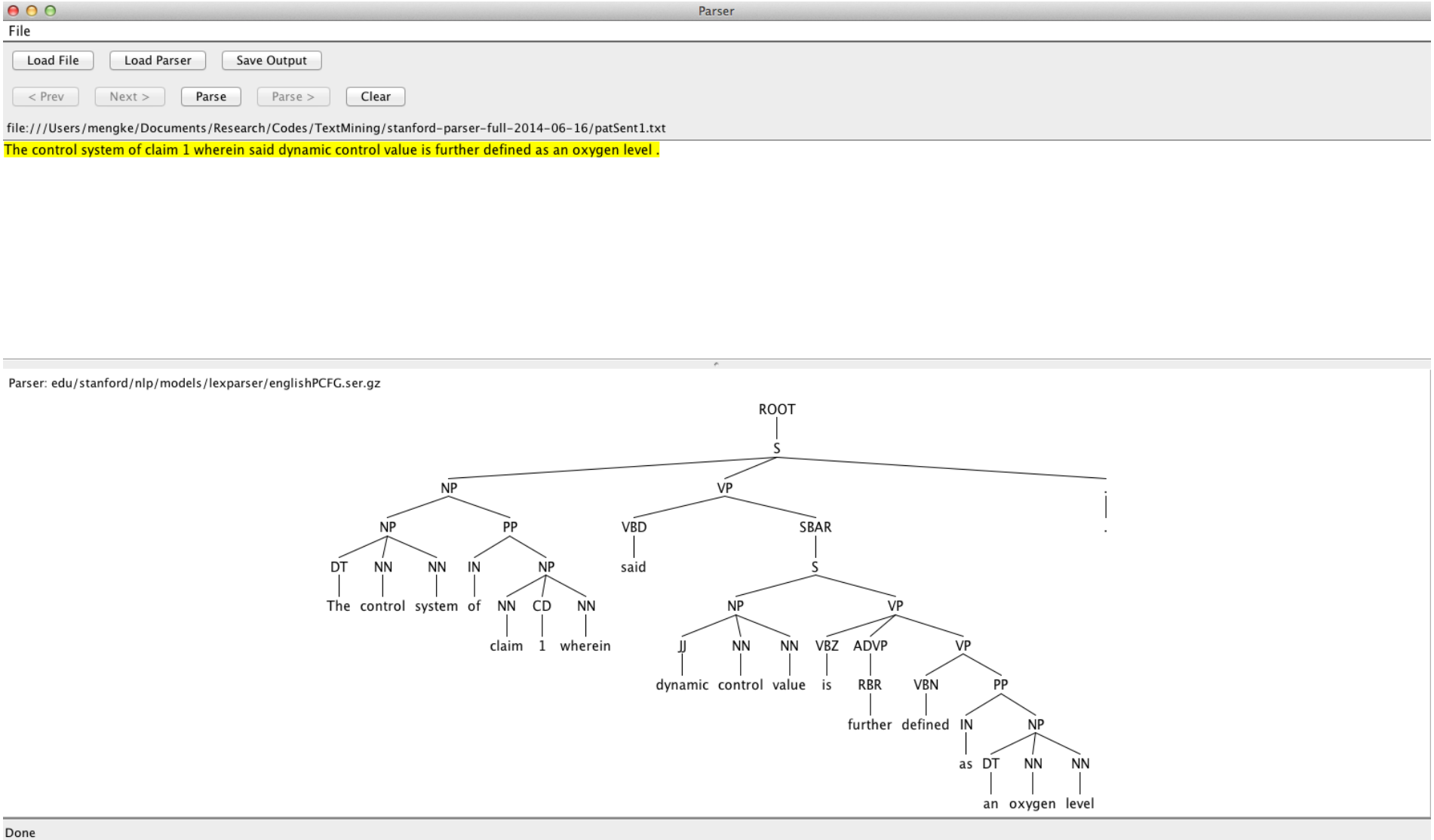
Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Stanford NLP Software

- Stanford NLP Group created a package of software that performs many different NLP tasks such as tolkenization, part of speech tagging, named entry recognition, creating parse trees, and classifying
- Set of functions that can be called via java commands in the command line as well as a GUI version that can perform the same tasks

Parse Tree of Patent



Parse Tree of Patent in Penn Treebank format

(ROOT (S (NP (NP (DT The) (NN control) (NN system)) (PP (IN of) (NP (NN claim) (CD 1) (NN wherein)))) (VP (VBD said) (SBAR (S (NP (JJ dynamic) (NN control) (NN value)) (VP (VBZ is) (ADVP (RBR further)) (VP (VBN defined) (PP (IN as) (NP (DT an) (NN oxygen) (NN level)))))))) (. .)))

Outline

1. The Patent System
2. Introduction to Information Retrieval Techniques
   - Introduction
   - Text Mining Techniques
3. Introduction to Weka
4. Experiments in Classification
5. Natural Language Processing
   - Introduction
6. Penn Treebank Format
   - Deriving a Parse Tree Structure for a US Patent
7. Conclusion

Conclusion

- Tolkenization in terms of bag of words methods and n-grams methods allows us to develop characterizations of text documents
- Vector Space models can take those tolkenized documents and develop numerical vectors off the words inside a document
- Using parse trees, one can create context free grammar that represents the structure of a sentence

Future Work

- Attempting to make feature vectors from the parse tree
- Developing new features that could more accurately relate the elements inside a patent claim than standard models for speech and basic IR models