|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

COURSE OUTCOMES

CO-1	Analyze the natural language text at word level and syntactic structures.
CO-2	Discuss the importance of natural language.
CO-3	Identify the concepts ofText mining.
CO-4	Apply information retrieval techniques.

Department Vision

We envision our department as a catalyst for developing educated, engaged and employable individuals whose collective energy will be the driving force for prosperity and the quality of life in our diverse world.

Department Mission

Our mission is to provide quality technical education in the field of information technology and to strive for excellence in the education by developing and sharpening the intellectual and human potential for good industry and community.

Program Outcomes (POs)

1. **Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem Analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/Development of Solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct Investigations of Complex Problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

Conti...

5. **Modern Tool Usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The Engineer and Society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and Sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and Team Work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project Management and Finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long Learning**: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Program Educational Objectives (PEO)

- **PEO1:** possess expertise in problem solving, design and analysis, technical skills for a fruitful career accomplishing professional and social ethics with exposure to modern designing tools and technologies in Information Science and Engineering.
- **PEO2:** excel in communication, teamwork and multiple domains related to engineering issues accomplishing social responsibilities and management skills.
- **PEO3:** outclass in competitive environment through certification courses, gaining leadership qualities and progressive research to become successful entrepreneurs.

Program Specific Outcome (PSO)

- **PSO1:** Apply the Knowledge of Information Science to develop software solutions in current research trends and technology.
- **PSO2:** Create Social awareness & environmental wisdom along with ethical responsibility to lead a successful career and sustain passion using optimal resources to become an Entrepreneur.

SYLLABUS

MODULE-1

- **Overview and language modeling:** Overview: Origins and challenges of NLP-Language and Grammar-Processing Indian Languages- NLP Applications-Information Retrieval. Language Modeling: Various Grammar- based Language Models-Statistical Language Model.

MODULE-2

- **Word level and syntactic analysis:** Word Level Analysis: Regular Expressions-Finite-State Automata-Morphological Parsing-Spelling Error Detection and correction-Words and Word classes-Part-of Speech Tagging. Syntactic Analysis: Context-free Grammar-Constituency- Parsing-Probabilistic Parsing.

MODULE-3

- **Extracting Relations from Text: From Word Sequences to Dependency Paths:**
- Introduction, Subsequence Kernels for Relation Extraction, A Dependency-Path Kernel for Relation Extraction and Experimental Evaluation.
- **Mining Diagnostic Text Reports by Learning to Annotate Knowledge Roles:** Introduction, Domain Knowledge and Knowledge Roles, Frame Semantics and Semantic Role Labeling, Learning to Annotate Cases with Knowledge Roles and Evaluations.
- **A Case Study in Natural Language Based Web Search:** InFact System Overview, The GlobalSecurity.org Experience.

MODULE-4

- **Evaluating Self-Explanations in iSTART: Word Matching, Latent Semantic Analysis, and Topic Models:** Introduction, iSTART: Feedback Systems, iSTART: Evaluation of Feedback Systems,
- **Textual Signatures: Identifying Text-Types Using Latent Semantic Analysis to Measure the Cohesion of Text Structures:** Introduction, Cohesion, Coh-Metrix, Approaches to Analyzing Texts, Latent Semantic Analysis, Predictions, Results of Experiments.
- **Automatic Document Separation: A Combination of Probabilistic Classification and Finite-State Sequence Modeling:** Introduction, Related Work, Data Preparation, Document Separation as a Sequence Mapping Problem, Results.
- **Evolving Explanatory Novel Patterns for Semantically-Based Text Mining:** Related Work, A Semantically Guided Model for Effective Text Mining.

MODULE-5

- **INFORMATION RETRIEVAL AND LEXICAL RESOURCES:** Information Retrieval: Design features of Information Retrieval Systems-Classical, Non classical, Alternative Models of Information Retrieval – valuation Lexical Resources: World Net-Frame Net- Stemmers-POS Tagger-Research Corpora.

BOOKS

Sl. No.	Name of Book	Author Name	Publication
1	Natural Language Processing and Information Retrieval	Tanveer Siddiqui, U.S. Tiwary,	**Oxford University Press, 2008**
2	Natural Language Processing and Text Mining	Anne Kao and Stephen R. Poteet	**Springer-Verlag London Limited 2007**
3	Speech and Language Processing: Anintroduction to Natural Language Processing, Computational Linguistics and SpeechRecognition	Daniel Jurafsky and James H Martin	**2nd Edition, Prentice Hall, 2008**
4	Natural Language Understanding	James Allen	**2nd edition, Benjamin/Cummingspublishing company, 1995**
5	**Gerald J. Kowalski and Mark.T. Maybury**	**Information Storage and Retrieval systems**	**Kluwer academic Publishers, 2000.**

INTRODUCTION

OVERVIEW

- This gives an idea of natural language processing and information retrieval.
- Various levels of analysis involved in NLP along with the knowledge used by these levels of analysis are discussed.
- Some of the difficulties in analyzing text and specific factors that make automatic processing of languages difficult are also touched upon.


The 21st Century
Percentage
Unstructured
Structured
BIG DATA

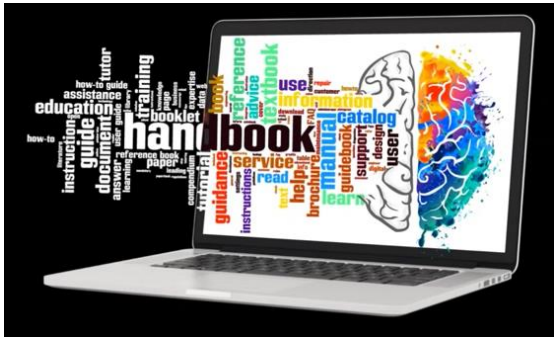
WHAT IS NATURAL LANGUAGE PROCESSING

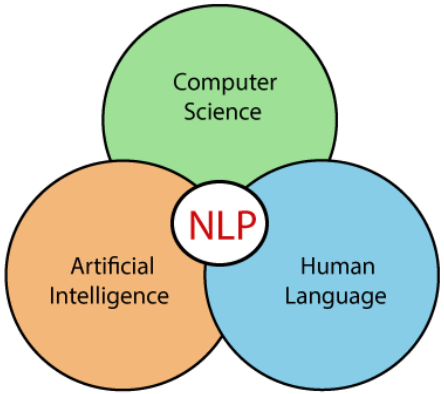
- Language is the primary means of communication used by humans. It is the tool used to express the greater part of ideas and emotions.
- ***Natural Language Processing (NLP) is concerned with the development of computational models of aspects of human language processing.***
- Two main reasons for development:

1. To develop automated tools for language processing.
2. To gain a better understanding of human communication.

TEXT MINING AND NLP

Text Mining/Text Analytics is the process of deriving meaningful Information from natural language text.





NLP: Natural Language Processing is a part of computer science and Artificial intelligence which deals with human languages.

Two Major Approaches to NLP

- **Rationalist Approach:** Assumes the existence of some language faculty in human brain.

Supporters argue that it is not possible for children to learn a complex thing like natural language from limited sensory inputs.

- **Empiricist Approach:** They do not believe in existence of a language faculty. They believe in existence of principles like pattern recognition, generalization and association.

History of NLP

First Phase (Machine Translation Phase) - Late 1940s to late 1960s

- The research on NLP started in early 1950s after Booth & Richens' investigation and Weaver's memorandum on machine translation in 1949.
- 1954 was the year when a limited experiment on automatic translation from Russian to English demonstrated in the Georgetown-IBM experiment.
- In the same year, the publication of the journal MT (Machine Translation) started.
- The first international conference on Machine Translation (MT) was held in 1952 and second was held in 1956.
- In 1961, the work presented in Teddington International Conference on Machine Translation of Languages and Applied Language analysis was the high point of this phase.

HISTORY of NLP

Second Phase (AI Influenced Phase) – Late 1960s to late 1970s

- In early 1961, the work began on the problems of addressing and constructing data or knowledge base. This work was influenced by AI.
- In the same year, a BASEBALL question-answering system was also developed. The input to this system was restricted and the language processing involved was a simple one.
- A much advanced system was described in Minsky (1968). This system, when compared to the BASEBALL question-answering system, was recognized and provided for the need of inference on the knowledge base in interpreting and responding to language input.

HISTORY of NLP

Third Phase (Grammatico-logical Phase) – Late 1970s to late 1980s

- The grammatico-logical approach, towards the end of decade, helped us with powerful general-purpose sentence processors like SRI's Core Language Engine and Discourse Representation Theory, which offered a means of tackling more extended discourse.
- In this phase we got some practical resources & tools like parsers, e.g. Alvey Natural Language Tools along with more operational and commercial systems, e.g. for database query.
- The work on lexicon in 1980s also pointed in the direction of grammatico-logical approach.

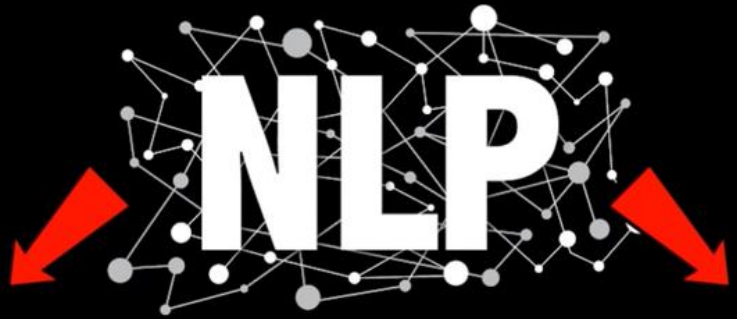
HISTORY of NLP

Fourth Phase (Lexical & Corpus Phase) – The 1990s

- The phase had a lexicalized approach to grammar that appeared in late 1980s and became an increasing influence.
- There was a revolution in natural language processing in this decade with the introduction of machine learning algorithms for language processing.

Components of NLP

Natural Language Understanding



Natural Language Generation

Study of Human Languages

Discipline	Problems	Tools
Linguists	How phrases and sentences can be formed with words? What curbs the possible meaning for a sentence?	Intuitions about well-formedness and meaning. Mathematical model of structure. For example, model theoretic semantics, formal language theory.
Psycholinguists	How human beings can identify the structure of sentences? How the meaning of words can be identified? When does understanding take place?	Experimental techniques mainly for measuring the performance of human beings. Statistical analysis of observations.
Philosophers	How do words and sentences acquire the meaning? How the objects are identified by the words? What is meaning?	Natural language argumentation by using intuition. Mathematical models like logic and model theory.
Computational Linguists	How can we identify the structure of a sentence How knowledge and reasoning can be modeled? How we can use language to accomplish specific tasks?	Algorithms Data structures Formal models of representation and reasoning. AI techniques like search & representation methods.

Computational Models

- **Knowledge Driven:** Rely on explicitly coded linguistic knowledge, often expressed as **handcrafted grammar rules**. Acquiring and encoding such knowledge is difficult.
- **Data Driven:** Presume the existence of a large amount of data and usually employ some **machine learning techniques** to learn syntactic patterns. The amount of human effort is less and performance is dependent on the quantity of data.

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7 A and B

Department of Information Science & Engineering

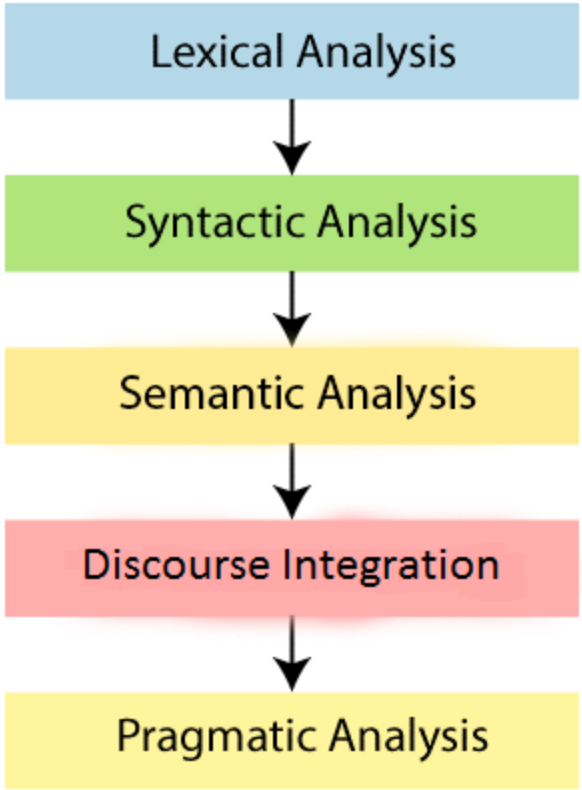
Aca. Year: ODD SEM /2021-22

Language and Knowledge

OVERVIEW

- Language is the medium of expression in which knowledge is deciphered.
- Language, being a medium of expression, is the outer form of the content it expresses. The same content can be expressed in different languages.
- The meaning of one language is written in the same language.

Phases of NLP



Lexical Analysis

- It is Simplest Level, which involves in Analysis of words.
- Words are the most fundamental unit of any natural language text. Word level processing requires morphological knowledge.
- Morphological- knowledge about the structure and formation of words from basic units.
- The rules for forming words from morphemes are language specific.
- This phase scans the source code as a stream of characters and converts it into meaningful lexemes. It divides the whole text into paragraphs, sentences, and words.

Syntactic Analysis

- It consists a sequence of words as a unit, usually a sentence, and finds its structure.
- It decomposes a sentence into its constituents and identifies how they relate to each other. It captures grammaticality or non-grammaticality of sentences by looking at constraints like word order, number and case agreement.
- This level of processing requires syntactic knowledge, i.e. knowledge about how words are combined to form larger units such as phrases and sentences.

Semantic Analysis

- It is associated with the meaning of the language.
- It is concerned with creating meaningful representation of linguistic inputs.
- The general idea of semantic interpretation is to take natural language sentences and map them onto some representation of meaning.
- Defining meaning components is difficult as grammatically valid sentences can be meaningless.

Contd..

- Consider the example:

Kabir and Ayan are married. *(1.1a)*

Kabir and Suha are married. *(1.1b)*

- Syntactic structure and compositional semantics fail to explain these interpretations.
- We make use of pragmatic information.

Discourse Analysis

- Higher Level Analysis.
- Attempts to interpret the structure and meaning of even larger units. (paragraph and documents)
- It requires the resolution of anaphoric references and identification of discourse structure.
- It also requires discourse knowledge i.e, meaning of a sentence is determined by preceding sentences.

Contd..

- Pragmatic knowledge may be needed for resolving anaphoric references.

The district administration refused to give the trade union permission for the meeting because they feared violence. *(1.2a)*

The district administration refused to give the trade union permission for the meeting because they oppose government. *(1.2b)*

Pragmatic Analysis

- Highest level of processing.
- Deals with purposeful use of sentences in situations.
- It requires the knowledge of the world.
- It helps you to discover the intended effect by applying a set of rules that characterize cooperative dialogues.
- **For Example:** "Open the door" is interpreted as a request instead of an order.

CHALLENGES OF NLP

NLP Difficult

- Some factors that make NLP difficult are problems of representation and interpretation.
- Language computing requires precise representation of content.
- Natural Languages are highly ambiguous and vague.
- The greatest source of difficulty in natural language is identifying its semantics.
- Words alone do not make a sentence. Instead words as well as their syntactic and semantic relations give meaning to a sentence.

Contd..

- When we process written text or spoken utterances, we have access to underlying mental representation.
- For example, most news papers and TV channels use 9/11 to refer to the terrorist act on the World Trade Center in the USA in 2004.
- The only way the machine can learn the meaning of a specific word in a message is by considering the context or domain knowledge.
- The context of the word is defined by co-occurring word. Ex-while

Contd..

- Idioms, metaphor and ellipses add more complexity to identify the meaning of the written text.
- Example:- *The old man finally kicked the bucket.*
- Quantifier-scoping is another problem. (the, each, etc.)
- The ambiguity of natural languages is another difficulty.

Types of Ambiguity

Lexical Ambiguity

- Lexical Ambiguity exists in the presence of two or more possible meanings of the sentence within a single word.
- Ambiguity arises at word level.

Example:

- Manya is looking for a **match**.
- In the above example, the word match refers to that either Manya is looking for a partner or Manya is looking for a match. (Cricket or other match)

Solution:

Part-of-speech tagging and word sense disambiguation

Types of Ambiguity

Syntactic Ambiguity

- Syntactic Ambiguity exists in the presence of two or more possible meanings within the sentence.

Example:

- I saw the girl with the binocular.
- In the above example, did I have the binoculars? Or did the girl have the binoculars?

Solution:

Probabilistic parsing

Types of Ambiguity

Referential Ambiguity

- Referential Ambiguity exists when you are referring to something using the pronoun.

Example: Kiran went to Sunita. She said, "I am hungry."

- In the above sentence, you do not know that who is hungry, either Kiran or Sunita.

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7 A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Language and Grammar

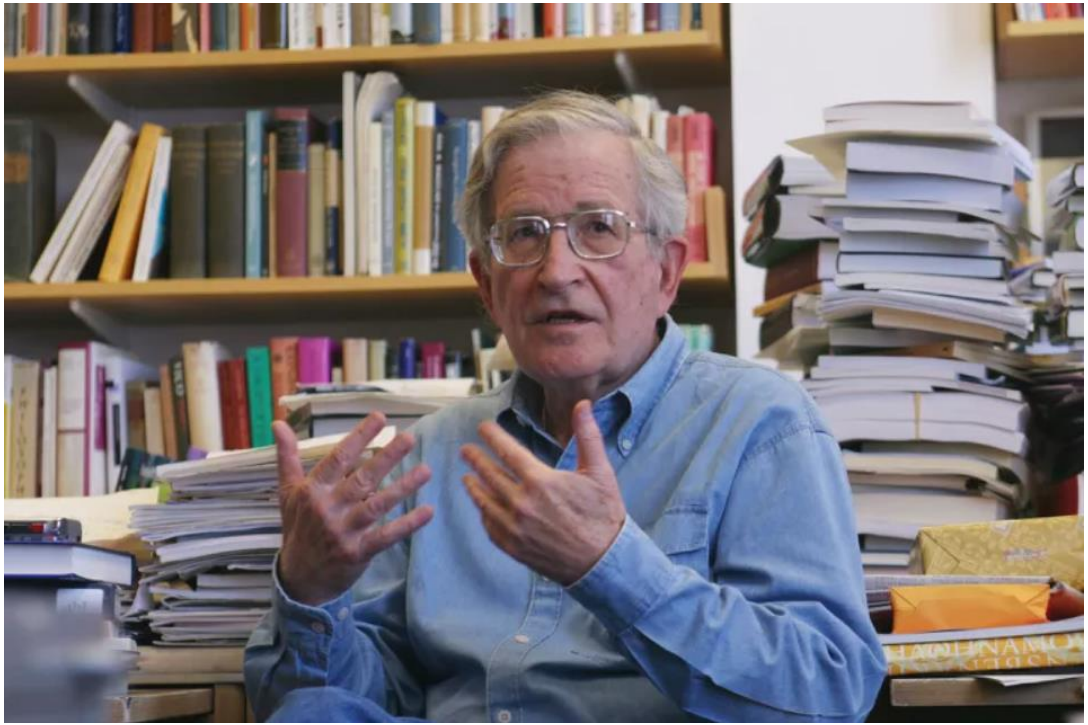
OVERVIEW

- Automatic processing of language requires the rules and exceptions of a language to be explained to the computer.
- Grammar consists of a set of rules that allows us to parse and generate sentences in a language.
- Rules relate information to coding devices at the language level not at the world-knowledge level.
- The world knowledge affects both the coding and the coding convention blurs the boundary between syntax and semantics.

Grammars

- Transformational Grammar (Chomsky 1957)
- Lexical functional Grammar (Kaplan and Bresnan 1982)
- Government and binding (Chomsky 1981)
- Generalized Phase Structure Grammar
- Dependency Grammar
- Paninian Grammar
- Tree-adjoining Grammar (Joshi 1985)
- Grammars focus on **derivations** and **relationships**.

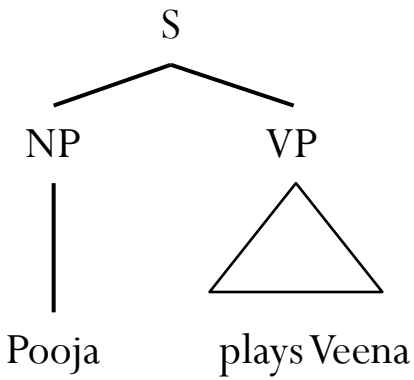
Noam Chomsky

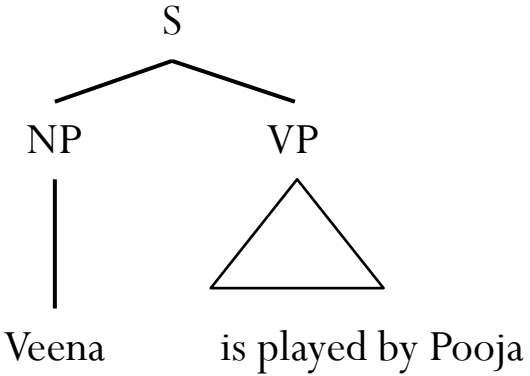


- The greatest contribution to grammar comes from Noam Chomsky, who proposed a hierarchy of formal grammar based on level of complexity.
- These grammars use phrase structure rules.
- Generative grammar basically refers to any grammar that uses a set of rules to specify or generate all and only grammatical sentences in a language.

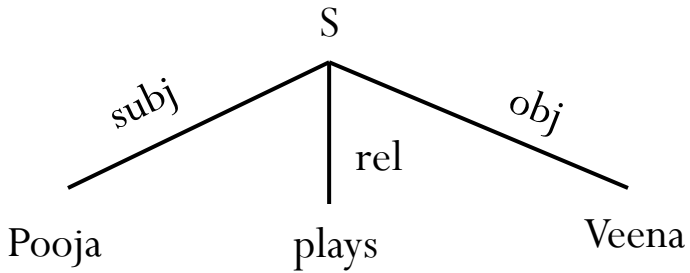
Surface and Deep Structure

- Chomsky proposed a complex system of transformational grammar based on Syntactic Structures.





Deep Structure



Transformational Grammar

It has three components:

- Phrase structure grammar
- Transformational rules
- Morphophonemic rules-these rules match each sentence representation to a string of phonemes.

Phrase Structure Grammar

It consists of rules that generate natural language sentences and assign a structural description to them.

- S → NP + VP
- VP → V + NP
- NP → Det + Noun
- V → Aux + Verb
- Det → the, a, an, …
- Verb → catch, write, eat, …
- Noun → police, snatcher, …
- Aux → will, is, can, …

Tranformational Grammar

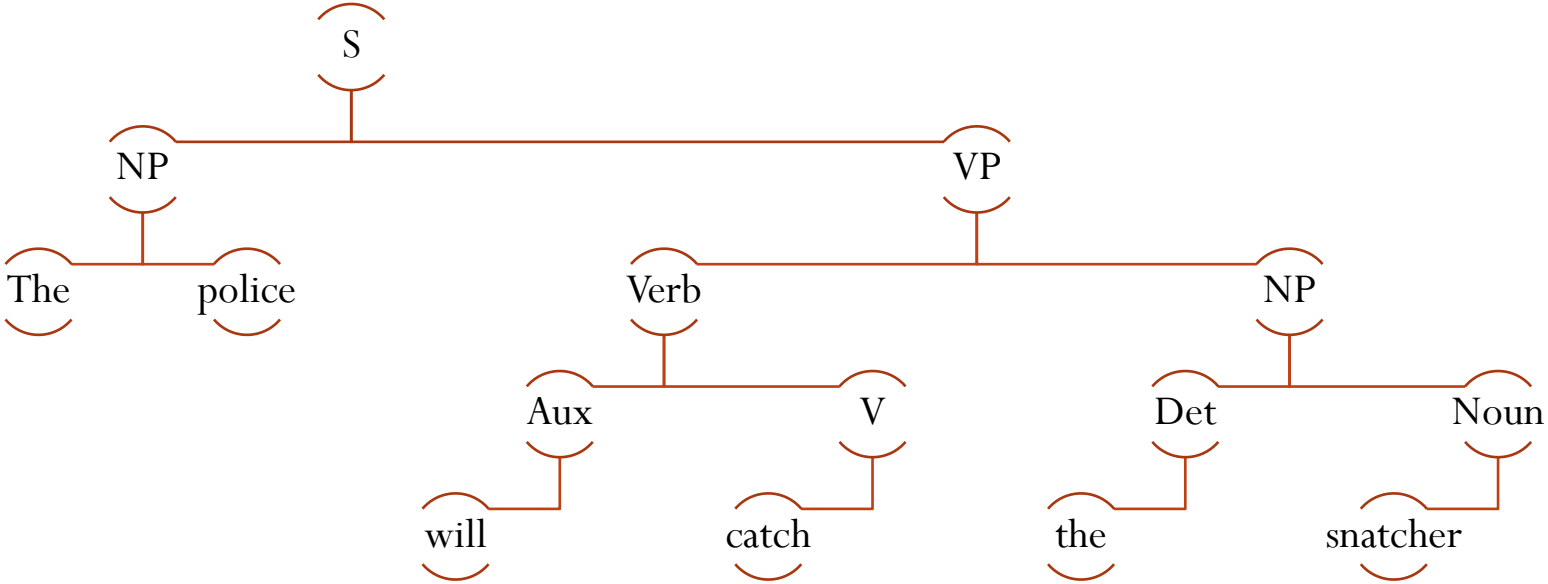
- It is a set of transformational rules, which transform one phrase-maker into another phrase-marker.
- These rules are applied on the terminal string generated by phrase structure rules.
- They are heterogeneous and may have more than one symbol on their left hand side.
- It is used to transform one surface representation into another. Example: active sentence into passive.
- $NP_1 - Aux - V - NP_2 \rightarrow NP_2 - Aux + be + en - V - by + NP_1$

Contd..

- Transformational rules can be obligatory or optional.
- An obligatory transformation is one that ensures agreement in number of subject and verb, etc.
- Optional transformation modifies the structure of a sentence while preserving its meaning.

Morphophonemic rules

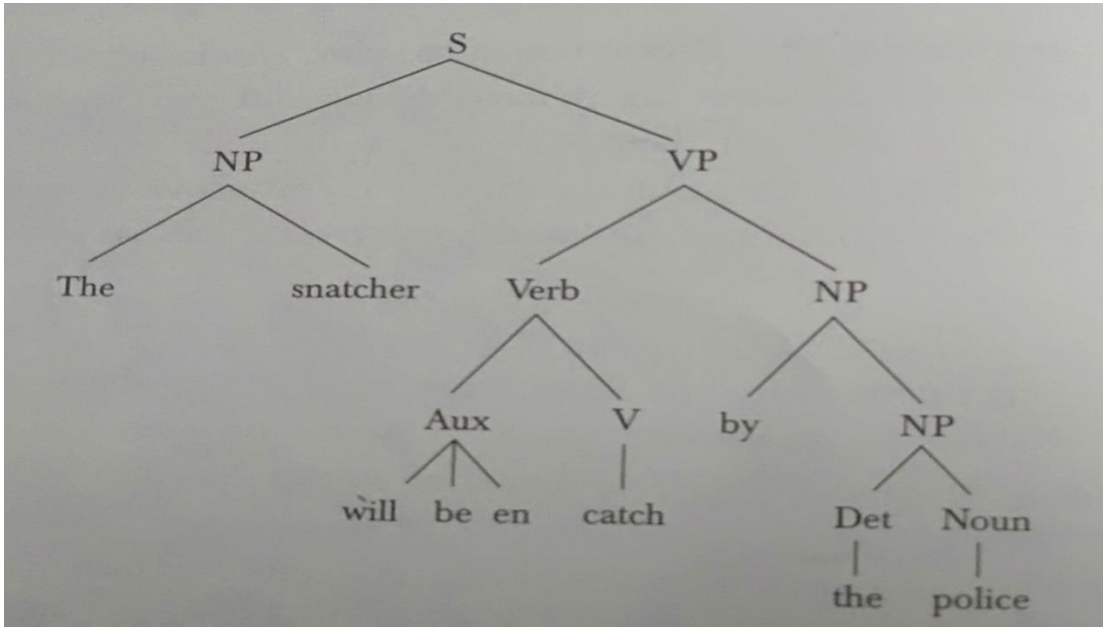
They match each sentence representation to a string of phonemes. Consider the sentence:
The police will catch the snatcher.



Parse Structure

Contd..

- Passive transformation rules will convert the sentence into:
- The + culprit + will + be + en + catch + by + police.

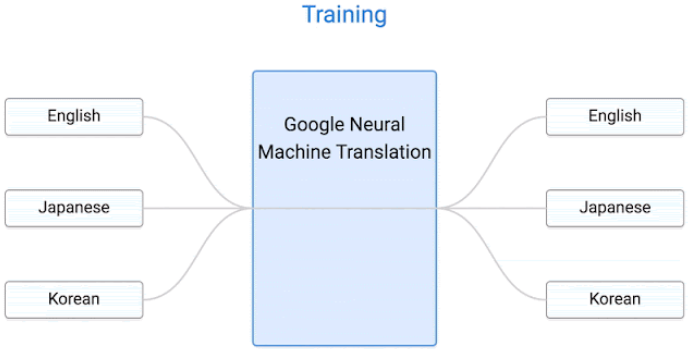


PROCESSING INDIAN LANGUAGES

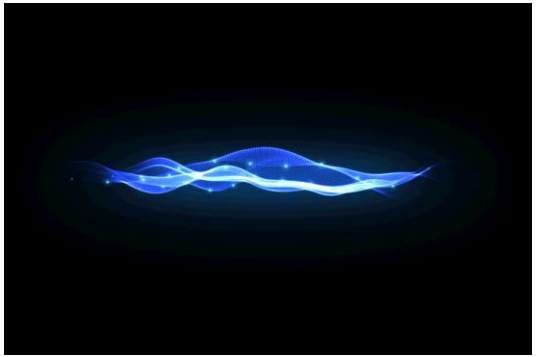
Differences between Indian languages and English.

- Indic scripts have a non-linear structure.
- Indian languages have SOV (Subject-Object-Verb) as the default sentence structure.
- Indian languages have a free word order.
- Spelling standardization is more subtle in Hindi than in English.
- Indian languages have a rich set of morphological variants.
- Indian languages make extensive and productive use of complex predicates (CPs).
- Indian languages use post-position case markers instead of prepositions.
- Indian languages use verb complexes consisting of verbs.

NLP Applications

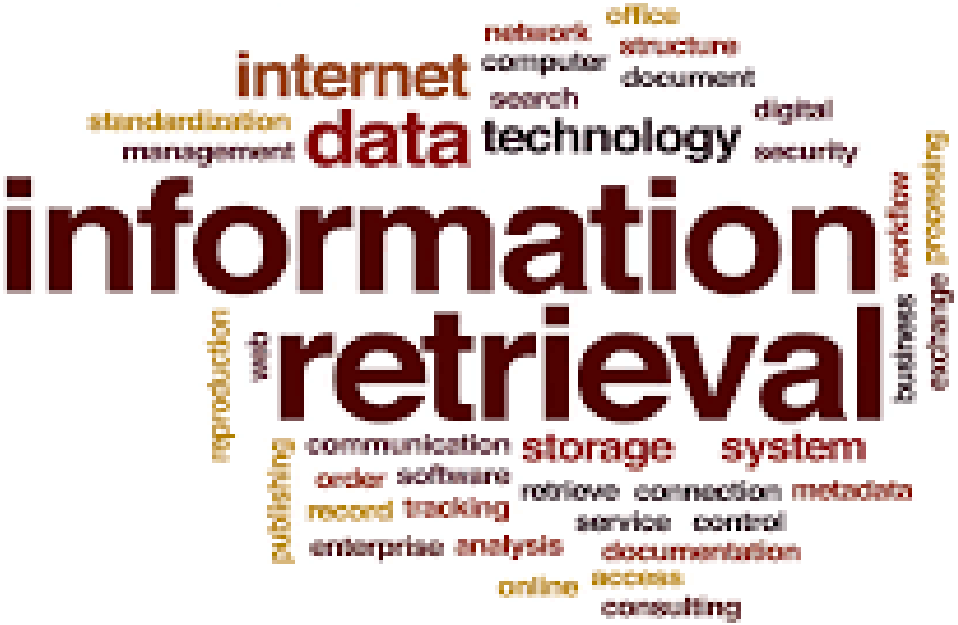


Speech Recognition

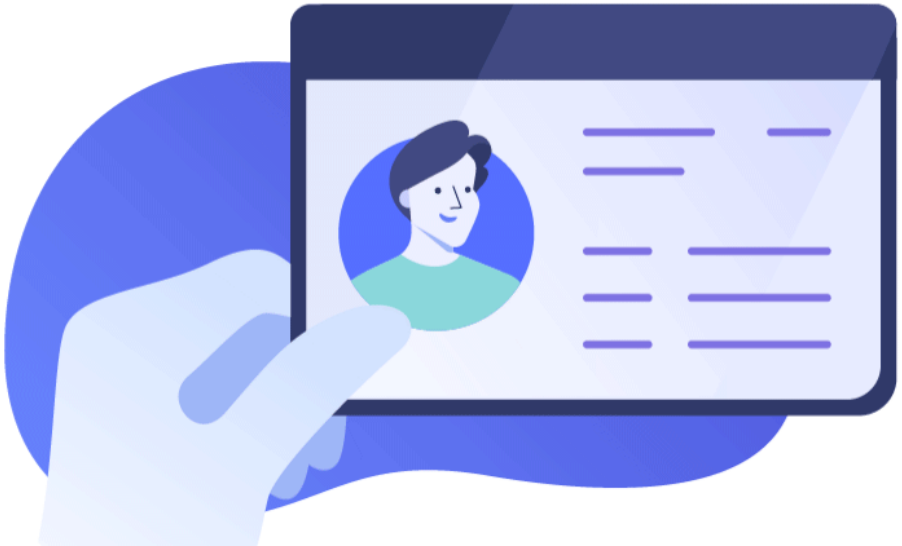


Speech Synthesis

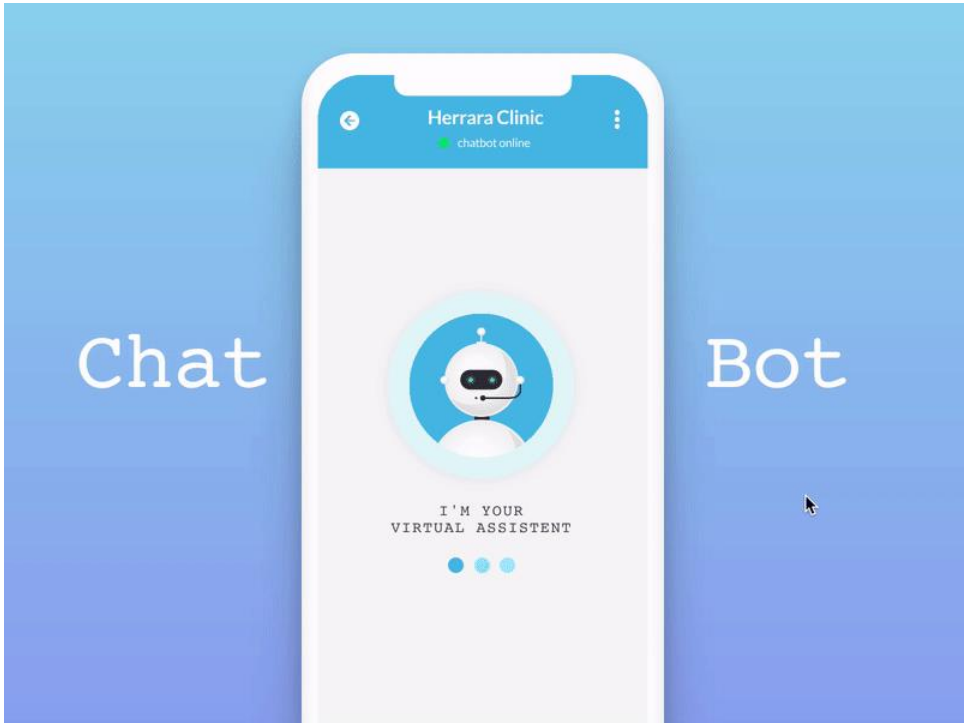
Information Retrieval



Information Extraction



Question Answering



Text Summarization

Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Peter

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7B

Department of Information Science & Engineering

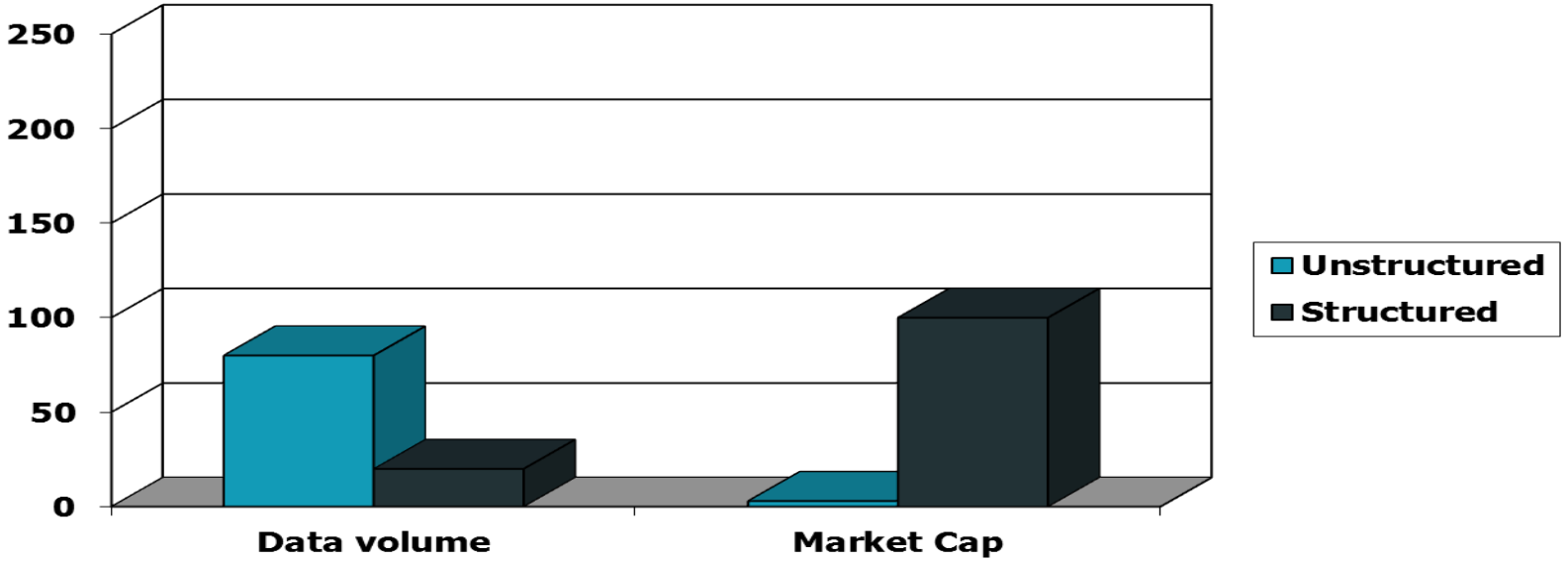
Aca. Year: ODD SEM /2021-22

Information Retrieval

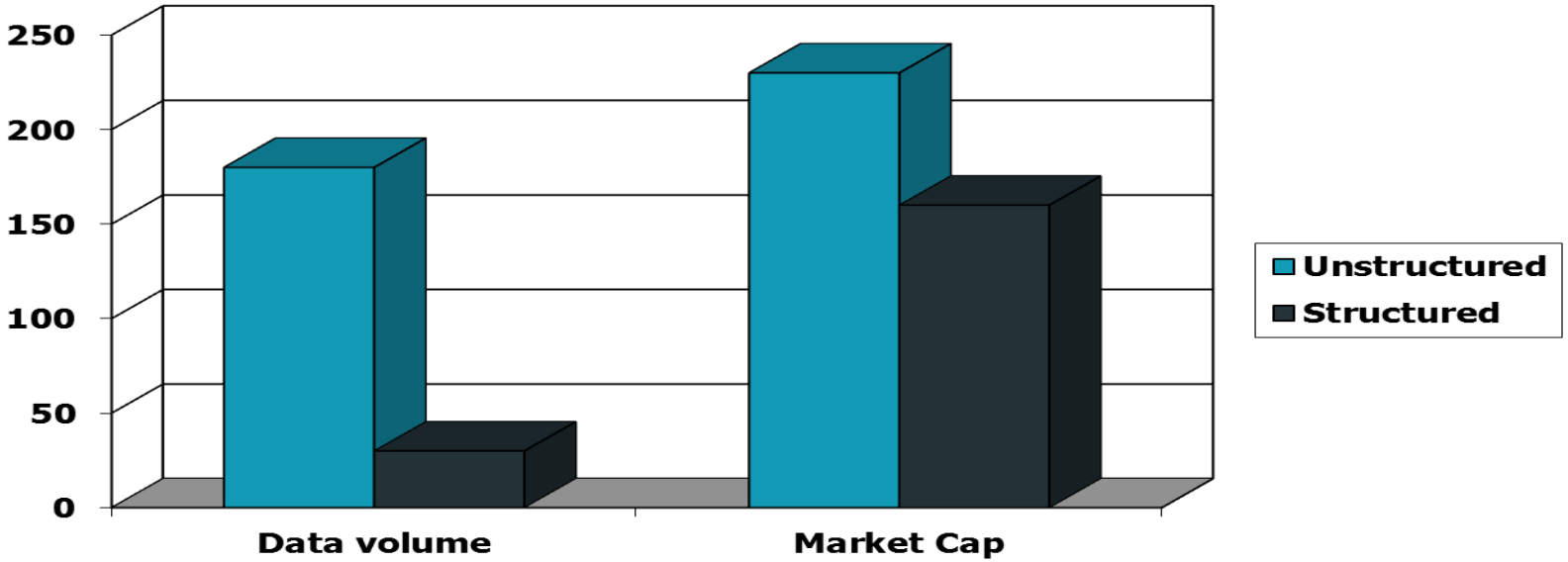
Information Retrieval

- Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).
  - These days we frequently think first of web search, but there are many other cases:
    - E-mail search
    - Searching your laptop
    - Corporate knowledge bases
    - Legal information retrieval

Unstructured (text) vs. structured (database) data in the mid-nineties



Unstructured (text) vs. structured (database) data today

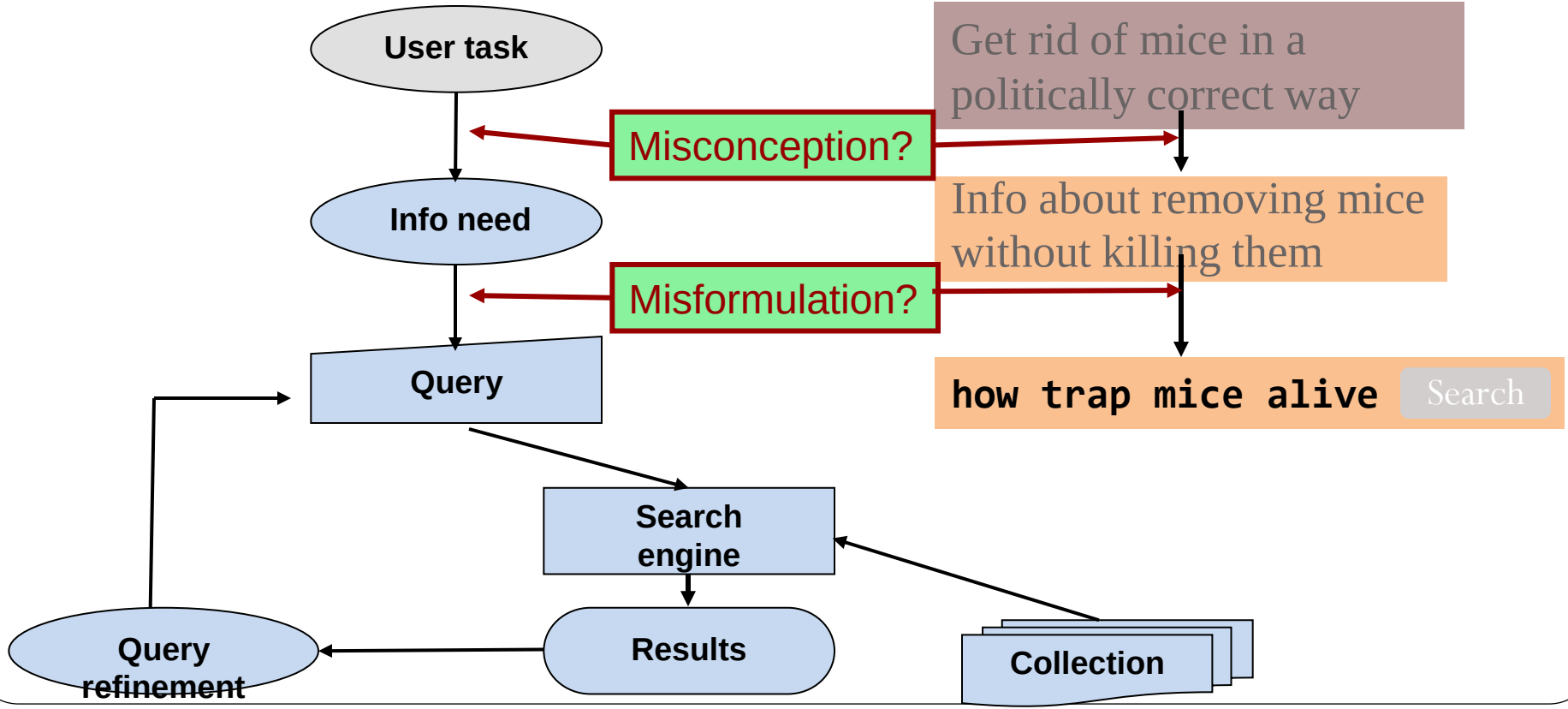


Basic assumptions of Information Retrieval

- Collection: A set of documents
  - Assume it is a static collection for the moment

- Goal: Retrieve documents with information that is relevant to the user's information need and helps the user complete a task

The classic search model



Major Issues in IR

- Representation of Document.
- Creates problems during retrieval due to polysemy, homonymy and synonymy.
- Polysemy involves the phenomenon of a lexeme with multiple meaning.
- Homonymy is an ambiguity in which words that appear the same have unrelated meanings.
- Synonymy creates problem when a document is indexed with one term and query contains a different term and two terms share a common meaning.

Issues in IR

- Inappropriate characterization of queries by the user.
- Many reasons for the vagueness and inaccuracy of the user's queries, say for instance, her lack of knowledge of the subject.
- User may fail to include relevant terms in the query or may include irrelevant terms.
- This problem can be dealt with modifying or expanding the queries.

How good are the retrieved docs?

- *Precision* : Fraction of retrieved docs that are relevant to the user's information need
- *Recall* : Fraction of relevant docs in collection that are retrieved

Major Goal of IR

- Search document in a manner relevant to the query, understanding what constitutes relevance is also an important issue.
- Relevance is subject in nature. (Saracevic 1991).
- It is considered as a binary concept, whereas it is in fact a continuous function.
- Different relevance frameworks: system, communication, psychological and situational frameworks.
- The most inclusive is the situational framework, which is based on the cognitive view of the information seeking process.
- It considers the importance of situation, context, multi-dimensionality and time.

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Language Modeling

Overview

- Domain of the language is vast. It presents infinite number of sentences to the reader.
- To handle such a large number of sentences, we have to create a model of the domain, which can subsequently be simplified and handled computationally.
- Two approaches for language modeling:

1. Define a grammar that can handle the language.
2. Capture the patterns in a grammar language statistically.

Introduction

Why and how to we model a language?

Understand and generate natural languages from a computational viewpoint.

First approach, Just take a language, try to understand every word and sentence of it and then come to a conclusion.

Second approach, study the grammar of various languages, compare them and possible arrive at reasonable models.

Model

- A model is a description of some complex entity or process.
- A language model is thus a description of language.
- Natural language is a complex entity and in order to process it through a computer based program, we need to build a representation of it.
- Language modeling can be viewed as problem of grammar inference or problem of probability estimation.

Grammar-based language model

- It uses the grammar of a language to create its model.
- It attempts to represent the syntactic structure of language.
- Hand-coded rules – linguistic unit.
- Example: sentence consists of noun phrase and verb phrase.
- It utilizes this structure and also relationships between these structures.

Statistical Language modelling

- This approach creates a language model by training it from a corpus.
- Training corpus needs to be large.
- "*Statistical langauge modelling is the attempt to capture regularities of natural language for the purpose of improving the performance of various natural language applications*."
- SLM is one of the fundamental tasks in NLP applications such as speech recognition, spelling correction, handwritten recognition and machine translation.

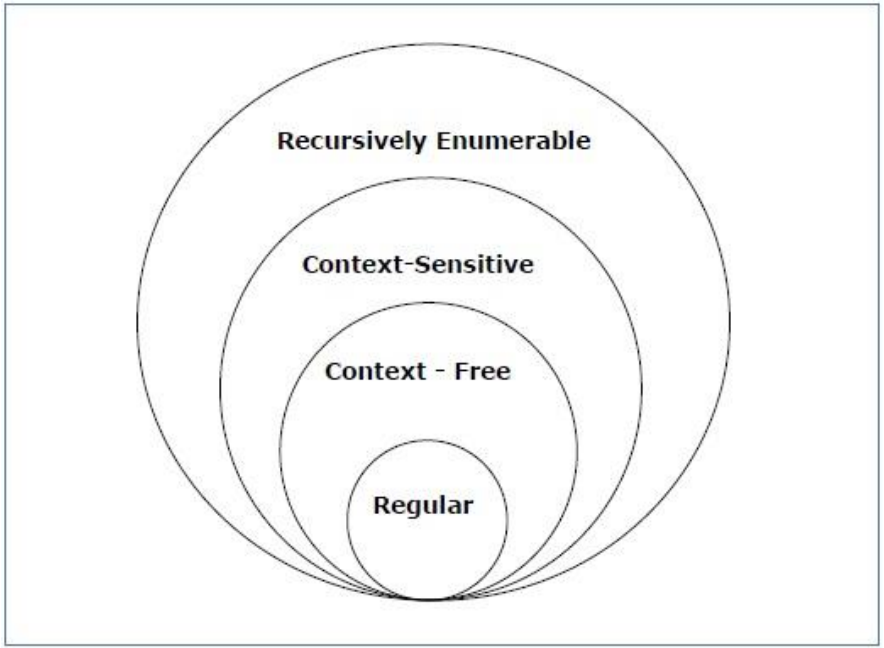
Various Grammar Based Language Models

Generative Grammars

- We can generate sentences in a language if we know a collection of words and rules in that language.
- Only those sentences that can be generated as per rules are grammatical.
- If we have a complete set of rules that can generate all possible sentences in a language, those rules provide a model of that language.
- Language is a relation between the sound and its meaning. So, Models should learn to deal with the meaning of its sentences.

Contd..

Hierarchical Grammar



Type - 3 Grammar

- **Type-3 grammars** generate regular languages. Type-3 grammars must have a single non-terminal on the left-hand side and a right-hand side consisting of a single terminal or single terminal followed by a single non-terminal.
- The productions must be in the form **X → a or X → aY**
- where **X, Y ∈ N** (Non terminal) and **a ∈ T** (Terminal)
- The rule **S → ε** is allowed if **S** does not appear on the right side of any rule.
- Example
- X → ε
- X → a | aY
- Y → b

Type - 2 Grammar

- **Type-2 grammars** generate context-free languages.
- The productions must be in the form **A → γ** where **A ∈ N** (Non terminal) and **γ ∈ (T ∪ N)*** (String of terminals and non-terminals).
- These languages generated by these grammars are be recognized by a non-deterministic pushdown automaton.

Example

S → X a

X → a

X → aX

X → abc

X → ε

Type - 1 Grammar

- **Type-1 grammars** generate context-sensitive languages. The productions must be in the form **α A β → α γ β** where **A ∈ N** (Non-terminal) and **α, β, γ ∈ (T ∪ N)*** (Strings of terminals and non-terminals)
- The strings **α** and **β** may be empty, but **γ** must be non-empty.
- The rule **S → ε** is allowed if S does not appear on the right side of any rule. The languages generated by these grammars are recognized by a linear bounded automaton.

Example

AB → AbBc

A → bcA

B → b

Type - 0 Grammar

- **Type-0 grammars** generate recursively enumerable languages. The productions have no restrictions. They are any phase structure grammar including all formal grammars.
- They generate the languages that are recognized by a Turing machine.
- The productions can be in the form of $\alpha \rightarrow \beta$ where α is a string of terminals and nonterminals with at least one non-terminal and α cannot be null. β is a string of terminals and non-terminals.

Example

$S \rightarrow ACaB$

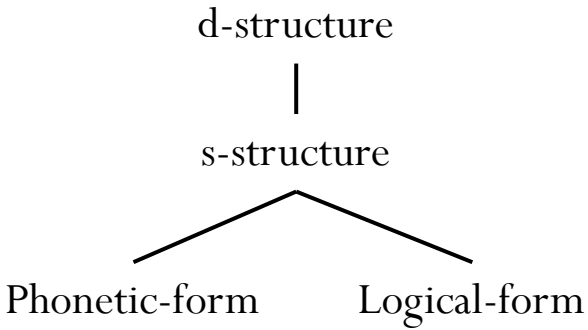
$Bc \rightarrow acB$

$CB \rightarrow DB$

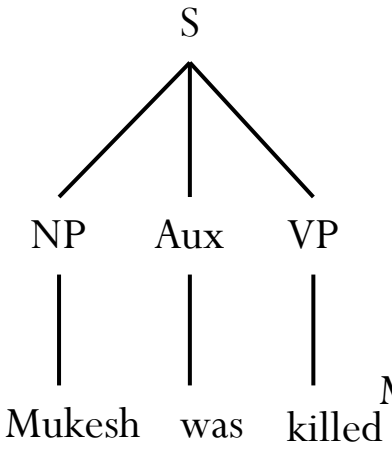
$aD \rightarrow Db$

Government and Binding (GB)

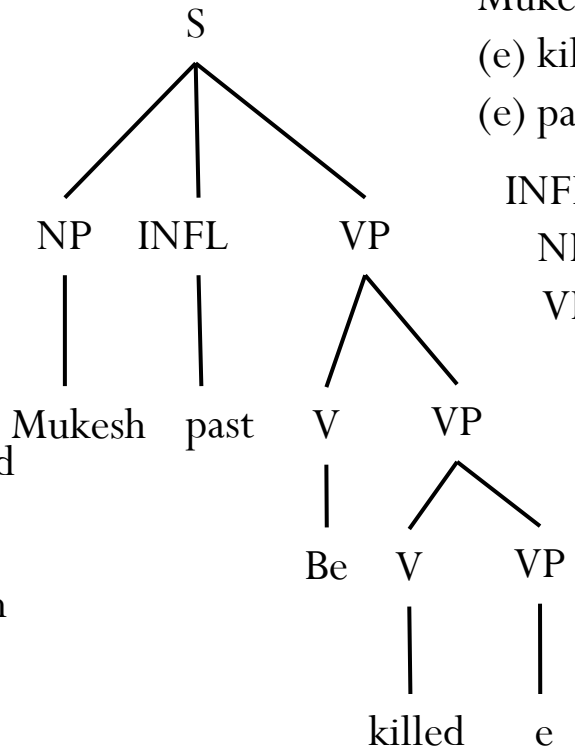
- Transformational Grammars assume two levels of sentences one at surface level and another at deep root level.
- GB theories have renamed them as s-level and d-level and identified by two more levels of representation called phonetic form and logical form.



Example



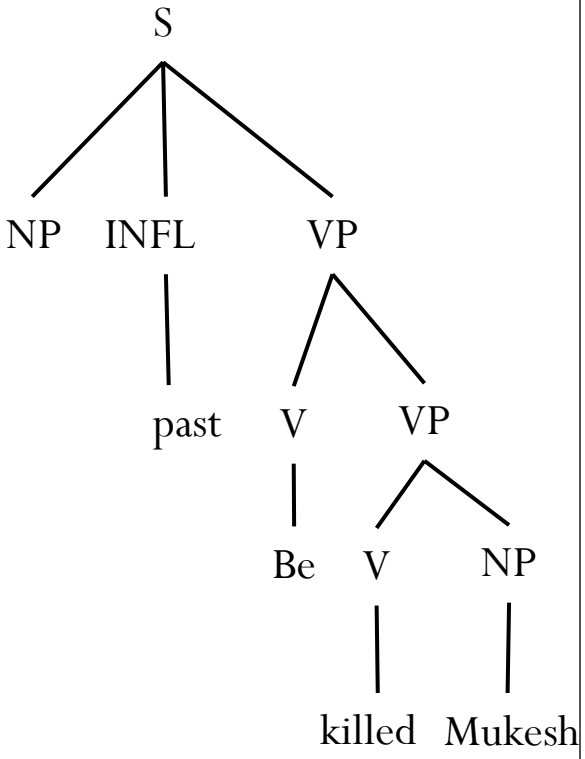
TG Representation



Surface Structure of sentence in GB

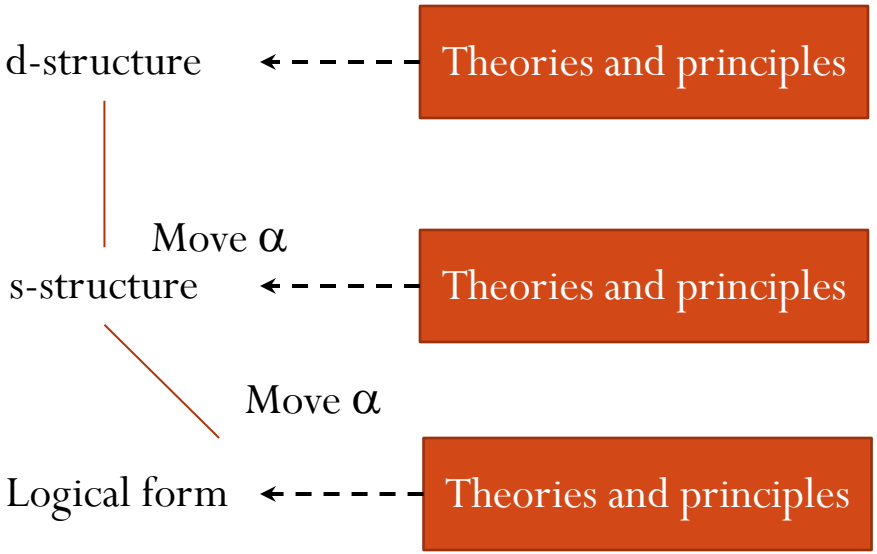
Mukesh was killed
(e) killed Mukesh
(e) past kill Mukesh

INFL: Inflection
NP: Noun phrase
VP: Verb phrase
e: empty



Deep Structure of sentence in GB

Components of GB



Example

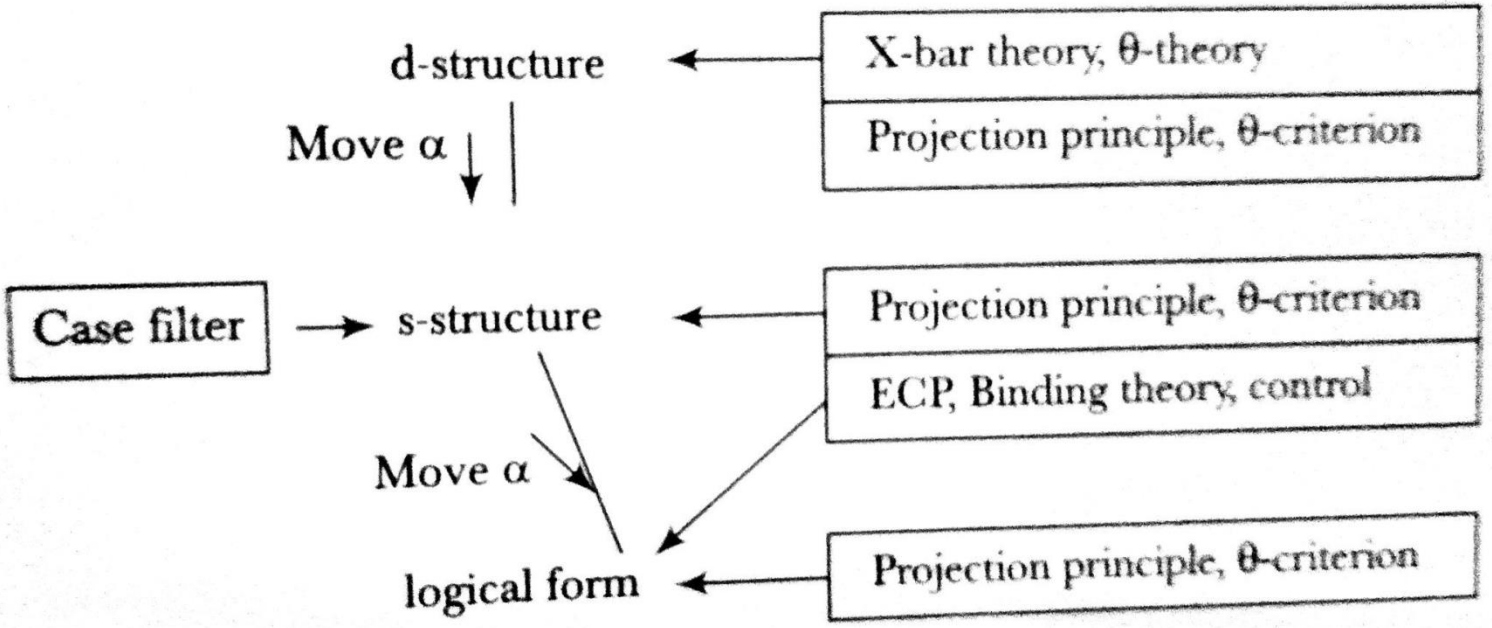
- Two countries are visited by most travellers.

LF1: $[_S$ Two countries are visited by $[_{NP}$ most travellers$]]$

LF2: Applying Move α

$[_{NP}$ Most travellers$_i$ $]$ $[_S$ two countries are visited by $e_i]$

Organization of GB

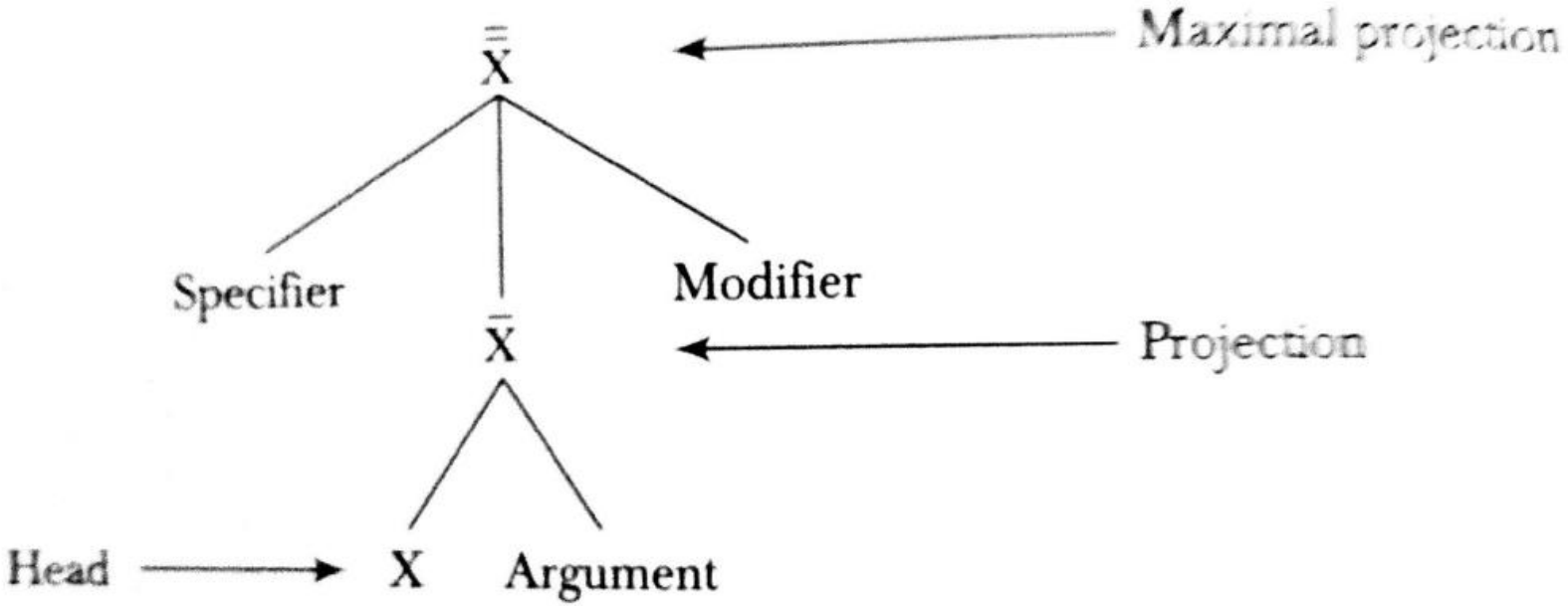


$\bar{X}$- Theory

$\bar{X}$ Theory

- $\bar{X}$ theory is the central concepts in GB.
- It defines phrase structures and sentence structure both as maximal projections.
- Noun Phrase (NP), Verb Phrase (VP), Adjective Phrase (AP) and Prepositional Phrase (PP) are maximal projections of noun (N), verb (V), adjective (A) and preposition (P).
- X={N, V, A, P}

General Phrase and Sentential Structure



NP Structure

1. NP: the food in a dhaba

$[_{NP}\ the\ [_{N}\ food\ ]_{PP}\ [in\ a\ dhabha\ ]]$

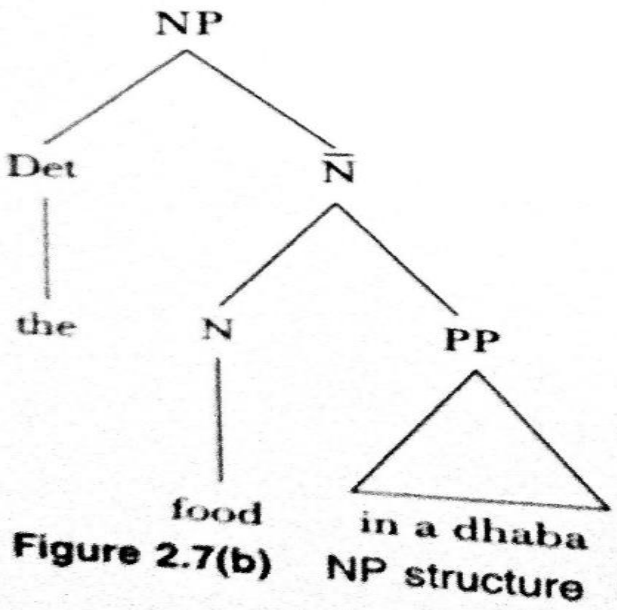
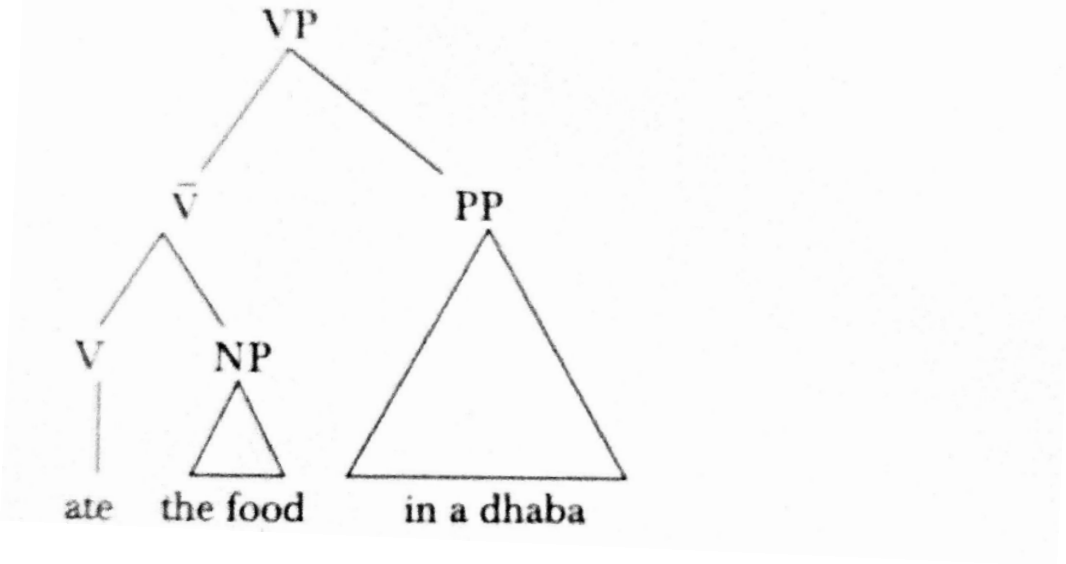


Figure 2.7(b) NP structure

VP Structure

2. VP: ate the food in a dhaba

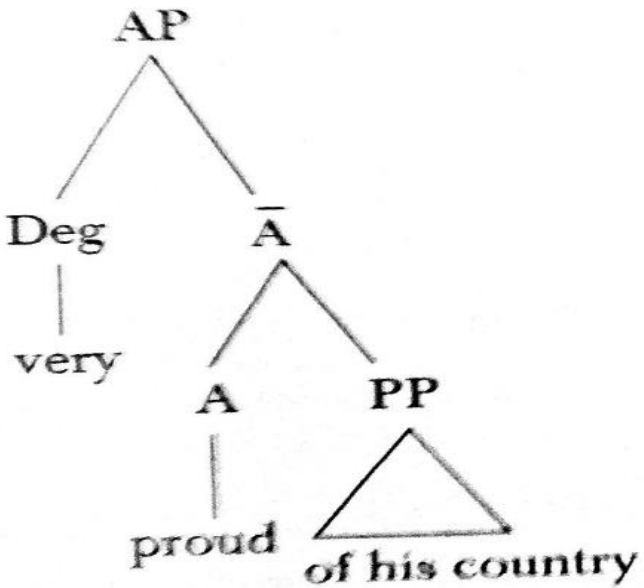
$[_{VP}[_{\bar{V}}[_{V}\ ate\ ]][_{NP}\ the\ food\ ]][_{PP}\ in\ a\ dhaba\ ]]$



AP Structure

3. AP: very proud of his country

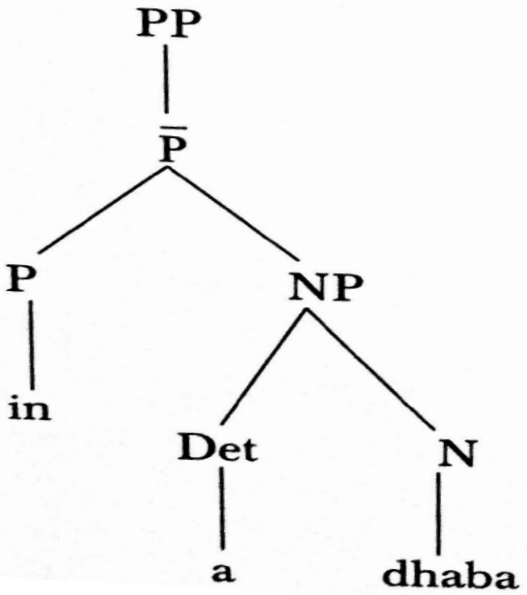
$[_{AP}\,[_{Deg}\,very\,]\,[_{\bar{A}}\,[_{A}\,proud\,]\,[_{PP}\,of\,his\,country\,]]]$



PP Structure

4. PP: in a dhaba

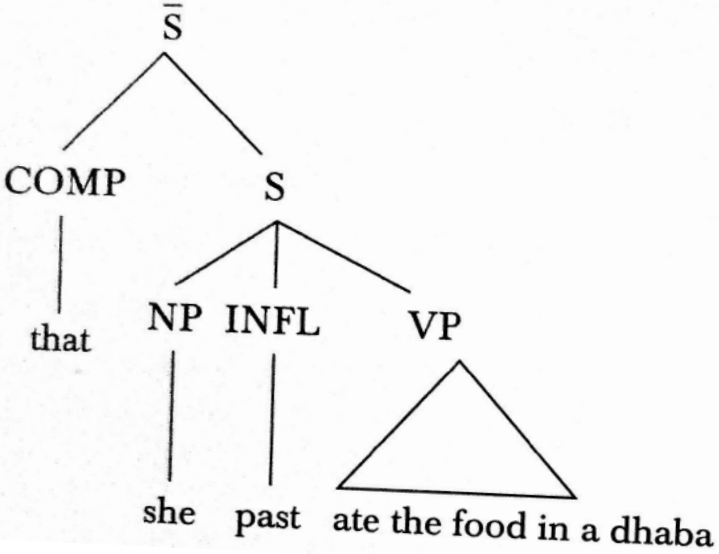
$$[_{PP}\,[_{\bar{P}}\,[_{P}\,in]\,[_{NP}\,[_{\mathrm{Det}}\,a]\,[_{N}\,dhabha]]]]$$



Maximal Projection

5. S: that she ate the food in a dhaba

$[_{\bar{S}}[_{COMP}\ that][_{S}[_{Det}\ she][_{INFL}\ past][_{VP}\ ate\ the\ food\ in\ a\ dhaba]]]$



Sub-categorization

- GB does not consider traditional phrase structure as an appropriate device for defining language constructs.
- It places the burden of ascertaining **well-formedness** to sub-categorization frames for heads.
- In principle, any maximal projection can be the argument of a head, but sub-categorization is used as a filter to permit various heads to select a certain subset of the range of maximal projections.
- It ensures well-formed structures even at the sentence level.
- As '**verb**' is not the head of the sentence, it cannot sub-categorize for '**subject NP**', while it can perfectly sub-categorize for '**object NP**'.

Projection Principle

- Basic notion in GB.
- Places a constraint on the three syntactic representations and their mapping from one to the other.
- The principle states that representations at all syntactic levels are projections from the lexicon.
- Lexical properties of categorical structure must be observed at each level.
- Example: Suppose ‘the object’ is not present at d-level then another NP cannot take this position at s-level.

Theta Theory or Theory of Thematic Relations

- Sub-categorization only places a restriction on syntactic categories which a head can accept.
- GB puts another restriction on the lexical heads through which it assigns certain roles to its arguments.
- These roles are pre-assigned and cannot be violated at any syntactical level as per the projection principle.

Theta-role

- Theta roles are certain thematic roles from which a head can select. These are mentioned in lexicon.
- Example:- The verb 'eat' can take arguments with θ–roles '(Agent, Theme)'.
- Agent is a special type of role which can be assigned by a head to outside arguments whereas other roles are assigned within its domain.
- **Mukesh ate food**
- **Theta-Criterion states that 'each argument bears one and only one θ–role, and each θ–role is assigned to one and only one argument.'**

C-command and Government

- C-command defines the scope of maximal projection. It is a basic mechanism through which many constraints are define on Move α.
- If any word or phrase (α or β) falls within scope of and is determined by a maximal projection, we say that it is dominated by the maximal projection.
- If there are two structures α and β related in such a way that 'every maximal projection dominating α dominates β'.
- α C-commands β

Government

α governs β iff:

α C-commands β

Movement, Empty Category and Co-Indexing

In GB the active to passive transformation is the result of NP movement.

What did Mukesh eat?

[Mukesh INFL eat what]

Empty Category

- When projection principle is applied to some cases of movement leads to existence of an abstract entity called empty category.
- Four types of empty categories:
- Wh-trace -a, -p
- NP trace +a, -p
- Small 'pro' -a, +p
- Big 'PRO' +a, +p

Anaphoric (+a or –a)
Pronominal (+p or –p)

Co-Indexing

- Co-indexing is the indexing of the subject NP and AGR at d-structure which are preserved by Move α operations at s-structure.
- When an NP movement takes place, a trace of the movement is created by having an indexed empty category (e_i) from the position at which the movement began to the corresponding indexed NP, i.e. NP_i.
- All A-positions at s-level are also freely indexed.
- Positions assigned θ–roles are called θ–positions, while others are $\bar{\theta}$–positions.

Binding Theory

α binds β iff:

α C-commands β, and

α and β are co-indexed

[e_i INFL kill Mukesh]

[$Mukesh_i$ was killed (by e_i)]

Mukesh was killed

Binding theory can be given as follows:

(a) An anaphor (+a) is bound in its governing category.

(b) A pronominal (+p) is free in its governing category.

(c) An R-expression (-a,-p) is free.

Example

A: Mukesh$_i$ knows himself$_i$

B: Mukesh$_i$ believes that Amrita knows him$_i$

C: Mukesh believes that Amrita$_j$ knows Nupur$_k$

Similar rules apply on empty categories also:

NP-trace: +a, -p: Mukesh$_i$ was killed e$_i$

wh-trace: -a, -p: Who$_i$ does he$_i$ like e$_i$

Empty Category Principle (ECP)

- Let us define Proper Government:

α properly governs β iff:

α governs β and α is lexical (i.e. N, V, A, or P) or

α locally $\bar{A}$ – binds β

The ECP says 'A trace must be properly governed'

Bounding and Control Theory

- There are many types of constraints on Move α.
- In English, the long distance movement for complement clause can be explained by bounding theory if NP and S are taken to be bounding nodes.
- The theory says that the application of Move may not cross more than one bounding node.
- The theory of control involves syntax, semantics and pragmatics.

Case Theory and Case Filter

- Deals with Distribution of NPs.
- NP must be assigned a case.
- English has nominative, objective, genitive etc. cases which are assigned to NPs at particular positions.

Case Filter: An NP is ungrammatical if it has phoentic content or it is an argument and is not case-marked.

Case Filters restrict the movement of NP at a position which has no case argument.

Lexical Functional Grammar

- LFG represents sentences at two syntactic levels-

Constituent structure – c-structure

Functional structure – f-structure

LFG is a formalism that is both computationally and linguistically motivated and provides precise algorithms for linguistic issues it can handle.

C-structure: derived from usual phrase and sentence structure syntax as in CFG.

F-structure: is the final product which encodes the information obtained from phrase and sentence structure rules and functional specifications.

Example 2.5

She saw stars in the sky.

CFG rules to handle this sentence are:

$$S \rightarrow NP\ VP$$
$$VP \rightarrow V\ \{NP\}\ \{NP\}\ PP^*\ \{S'\}$$
$$PP \rightarrow P\ NP$$
$$NP \rightarrow Det\ N\ \{PP\}$$
$$S' \rightarrow Comp\ S$$

where

S: sentence
P: preposition
S′: clause
{ } optional
V: verb
N: noun
Comp: complement
*: Phrase can appear any number of times including blank.

When annotated with functional specifications, the rules become:

Rule 1: $S \rightarrow$ NP VP
$\uparrow \text{subj} = \downarrow$ $\uparrow = \downarrow$

Rule 2: $VP \rightarrow V$ $\{NP\}$ $\{NP\}$ PP^* $\{S'\}$
$\uparrow \text{obj} = \downarrow$ $\uparrow \text{obj } 2 = \downarrow$ $\uparrow (\downarrow \text{case}) = \downarrow$ $\uparrow \text{comp} = \downarrow$

Rule 3: $PP \rightarrow P$ NP
$\uparrow \text{obj} = \downarrow$

Rule 4: $NP \rightarrow \{\text{Det}\}$ N $\{PP\}$
$\uparrow \text{Adjunct} = \downarrow$

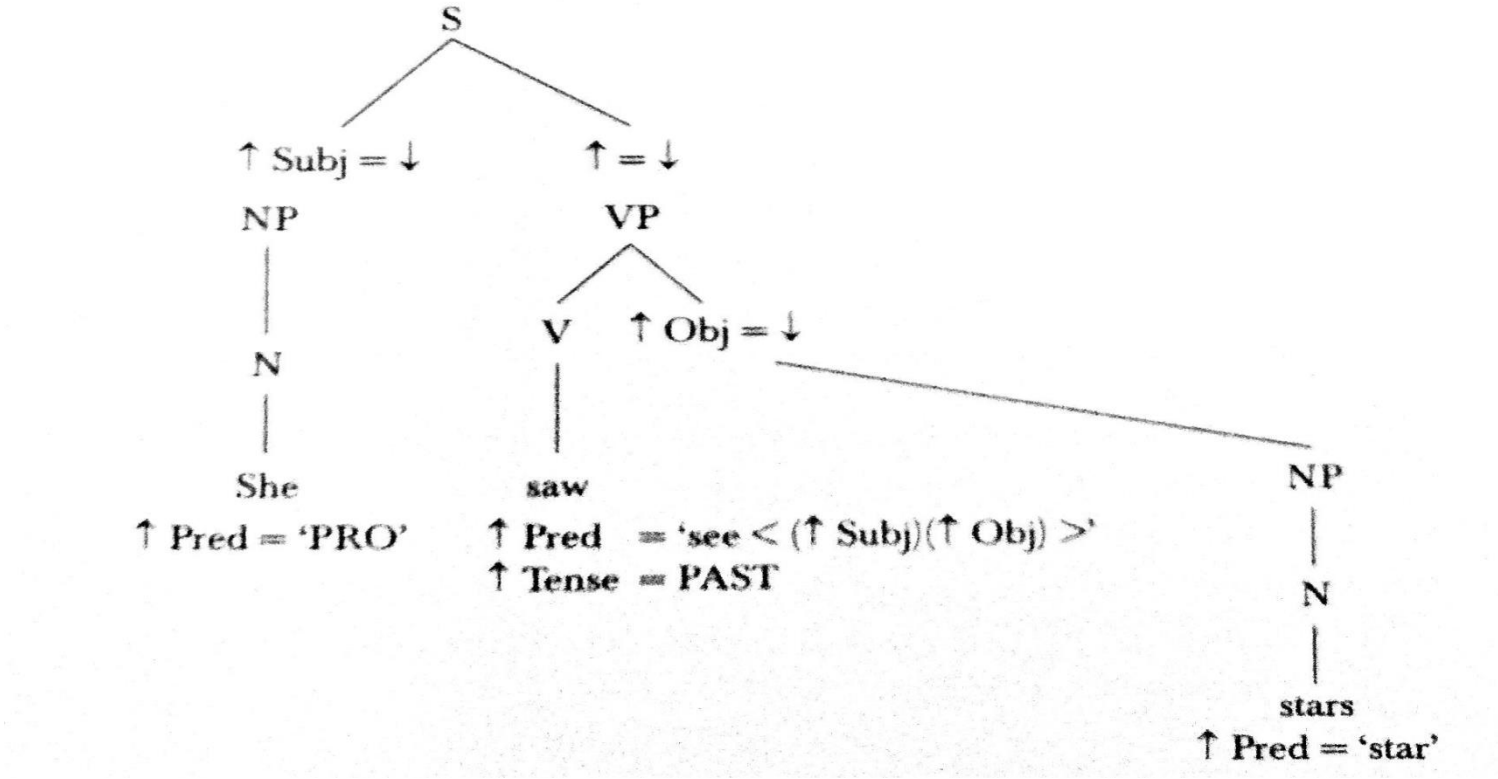
Rule 5: $S' \rightarrow$ Comp S
$\uparrow = \downarrow$

Lexical Entries

She saw stars.

She N	(↑ Pred) = 'PRO'
	(↑ Pers) = 3
	(↑ Num) = SG
	(↑ Gen) = FEM
	(↑ Case) = NOM
Saw V	↑ Pred = 'see < (↑ Subj) (↑ Obj) >'
	(↑ Tense = PAST)
Stars N	↑ Pred = 'Star'
	↑ Pers = 3
	↑ Num = PL

C-Structure


S
↑ Subj = ↓
↑ = ↓
NP
VP
N
V
↑ Obj = ↓
She
saw
NP
↑ Pred = 'PRO'
↑ Pred = 'see < (↑ Subj)(↑ Obj) >'
↑ Tense = PAST
N
stars
↑ Pred = 'star'

F-structure

$$
\begin{bmatrix}
\text{subj} & \begin{bmatrix} \text{Pers} & 3 \\ \text{Num} & \text{SG} \\ \text{Gen} & \text{FEM} \\ \text{Case} & \text{NOM} \\ \text{Pred} & \text{'PRO'} \end{bmatrix} \\
\text{obj} & \begin{bmatrix} \text{Pers} & 3 \\ \text{Num} & \text{PL} \\ \text{Pred} & \text{'Star'} \end{bmatrix} \\
\text{Pred} & \text{'see'} \langle (\uparrow \text{subj}) \ (\uparrow \text{obj}) \rangle
\end{bmatrix}
$$

Lexical Rules in LFG

Active: Tara ate the food.

Passive: The food was eaten by Tara.

Active: ↑ Pred = 'eat < (↑ Subj) (↑ Obj) >'

Passive: ↑ Pred = 'eat < (↑ Obl_{ag}) (↑ Subj) >'

Active: Tara gave a pen to Monika.

Passive: Tara gave Monika a pen.

Active: ↑ Pred = 'give < (↑ Subj) (↑ Obj_2) (↑ Obj) >'

Passive: ↑ Pred = 'give < (↑ Subj) (↑ Obj) (↑ Obl_{go})>'

Active: तारा हँसी

Taaraa hansii

Tara laughed

Causative: मोनिका ने तारा को हँसाया

Monika ne Tara ko hansaayaa

Monika Subj Tara Obj laugh-cause-past

Monika made Tara to laugh.

Active: ↑ Pred = 'Laugh <↑ Subj>'

Causative: ↑ Pred = 'cause <(↑ Subj) (↑ Obj) (Comp)>'

wh-movement

Which picture does Tara like–most?

The f-structure can be represented as follows:

$$\begin{bmatrix} \text{Focus} & \begin{bmatrix} \text{Obl}_{th} & \begin{bmatrix} \text{pred} & \text{'PRO'} \\ \text{Refl} & + \end{bmatrix} \\ \text{Pred} & \text{'picture } \langle(\text{Obl}_{th})\rangle\text{'} \end{bmatrix} \\ \text{Subj} & [\text{pred} \quad \text{'Tara'}] \\ \text{Obj} & [\quad] \\ \text{Pred} & \text{'like } \langle(\uparrow\text{Subj}) \ (\uparrow\text{Obj})\rangle\text{'} \end{bmatrix}$$

Paninian Framework

- Panini in 500BC.
- Asian Languages are SOV ordered.
- Inflectionally rich.
- Inflections provide important syntactic and semantic cues for language analysis and understanding.

Example

(a) माँ बच्चे को खाना देती है।

Maan Bachche ko khanaa detii hai

Mother child to food give–(s)

Mother gives food to the child.

(b) बच्चे को माँ खाना देती है।

Bachche ko Maan khanaa detii hai

Child to mother food give–(s)

Mother gives food to the child.

Example

उसने खाना खाया।	उसने खाना खा लिया।
Usne khanaa khaayaa	*Usne khaanaa kha liyaa*
He (Subj) food ate	He (Subj) food eat taken
He ate food	He ate food (completed the action)
वह चला	वह चल दिया
	He move given
He moved	He moved (started the action)

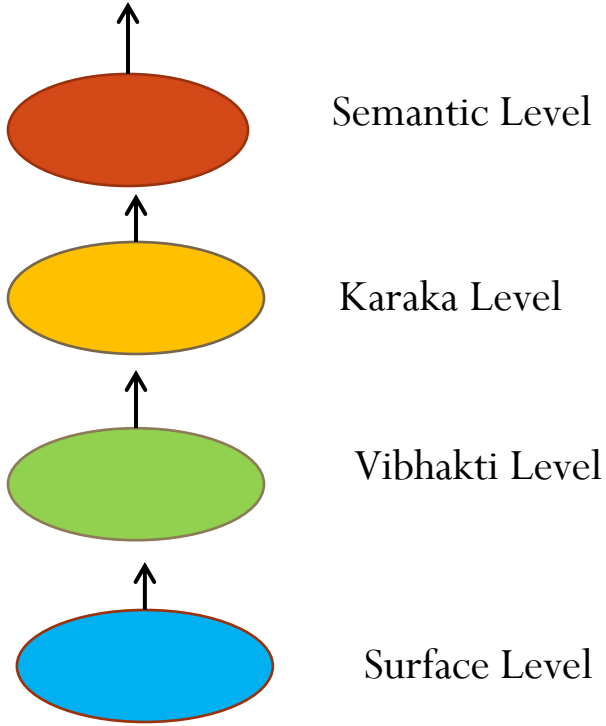
Example

In Indian languages, the nouns are followed by post-positions instead of prepositions. They generally remain as separate words in Hindi, except in the case of pronouns.

रेखा	के	पिता
Rekha	ke	pita
Rekha	of	father
Father	of	Rekha

उसके	पिता
Uske	*pita*
Her (His)	father

Layered Representation in PG



Karaka Theory

माँ बच्ची को आँगन में हाथ से रोटी खिलाती है। (2

Maan bachchi ko aangan mein haath se rotii khilaatii hei

Mother child-to courtyard-in hand-by bread feed (s).

The mother feeds bread to the child by hand in the courtyard.

माँ ने थाली से खाना उठाकर बच्चे को दिया।

Maan ne thaali se khana uthakar bachche ko diyaa

Mother-Karta plate from Apaadan food taking-up child-to gave.

The mother gave food to the child taking it up from the plate.

Issues in Paninian Grammar

(i) Computational implementation of PG, and

(ii) Adaptation of PG to Indian, and other similar languages.

(iii) Another difficulty arises when mapping between the Vibhakti and the semantic relation is not one to one.

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Statistical Language Model

Overview

- It is a probability distribution P(s) over all possible word sequences.
- n-gram model.

n-gram Model

- The goal of statistical language model is to estimate the probability of a sentence.
- Chain Rule:

$$P(s) = P(w_1, w, w_3, \ldots, w_n)$$
$$= P(w_1)\, P(w_2/w_1)\, P(w_3/w_1\, w_2)\, P(w_4/w_1\, w_2\, w_3) \ldots P(w_n/w_1\, w_2 \ldots w_{n-1}))$$
$$= \prod_{i=1}^{n} P(w_i/h_i)$$

Contd..

- To calculate sentence probability, we need to calculate the probability of a word, given the sequence of words preceding it.

$$P\left(w_i / h_i\right) = P(w_i / w_{i-n+1} \ldots w_{i-1})$$

- bi-gram and tri-gram estimate

$$P(s) \approx \prod_{i=1}^{n} P(w_i / w_{i-1})$$

$$P(s) \approx \prod_{i=1}^{n} P(w_i / w_{i-2} \cdot w_{i-1})$$

Maximum Likelihood Estimation

$$P(w_i / w_{i-n+1}, \ldots, w_{i-1}) = \frac{C(w_{i-n+1}, \ldots, w_{i-1}, w_i)}{\sum_{w} C(w_{i-n+1}, \ldots, w_{i-1}, w)}$$

$$P(w_i / w_{i-n+1}, \ldots, w_{i-1}) = \frac{C(w_{i-n+1}, \ldots, w_{i-1}, w_i)}{C(w_{i-n+1}, \ldots, w_{i-1})}$$

Example

Training set:

The Arabian Knights
These are the fairy tales of the east
The stories of the Arabian knights are translated in many languages

Bi-gram model:

$P(\text{the}/\langle s\rangle) = 0.67$	$P(\text{Arabian}/\text{the}) = 0.4$	$P(\text{knights}/\text{Arabian}) = 1.0$
$P(\text{are}/\text{these}) = 1.0$	$P(\text{the}/\text{are}) = 0.5$	$P(\text{fairy}/\text{the}) = 0.2$
$P(\text{tales}/\text{fairy}) = 1.0$	$P(\text{of}/\text{tales}) = 1.0$	$P(\text{the}/\text{of}) = 1.0$
$P(\text{east}/\text{the}) = 0.2$	$P(\text{stories}/\text{the}) = 0.2$	$P(\text{of}/\text{stories}) = 1.0$
$P(\text{are}/\text{knights}) = 1.0$	$P(\text{translated}/\text{are}) = 0.5$	$P(\text{in}/\text{translated}) = 1.0$
$P(\text{many}/\text{in}) = 1.0$		
$P(\text{languages}/\text{many}) = 1.0$		

Finding Probabilities

Test sentence(s): The Arabian knights are the fairy tales of the east.

$$P(\text{The}/\langle s\rangle) \times P(\text{Arabian}/\text{the}) \times P(\text{Knights}/\text{Arabian}) \times P(\text{are}/\text{knights}) \times P(\text{the}/\text{are}) \times P(\text{fairy}/\text{the}) \times P(\text{tales}/\text{fairy}) \times P(\text{of}/\text{tales}) \times P(\text{the}/\text{of}) \times P(\text{east}/\text{the})$$

$$= 0.67 \times 0.5 \times 1.0 \times 1.0 \times 0.5 \times 0.2 \times 1.0 \times 1.0 \times 1.0 \times 0.2$$

$$= 0.0067$$

Add-one Smoothing

$$P(w_i / w_{i-n+1}, \cdots, w_{i-1}) = \frac{C(w_{i-n+1}, \cdots, w_{i-1}, w_i)}{C(w_{i-n+1}, \cdots, w_{i-1}) + V}$$

Good-Turing Smoothing

$$f^* = (f+1)\frac{n_{f+1}}{n_f}$$

Example, Consider that the number of n-grams that occur 4 times is 25,108 and the number of n-grams that occur 5 times is 20,542. Then the smoothed count for 5 will be

$$\frac{20542}{25108} \times 5 = 4.09$$

Caching Technique

- Frequency of n-gram is not uniform across the text segments or corpus.
- The cache model combines the most recent n-gram frequency with the standard n-gram model to improve its performance locally.

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

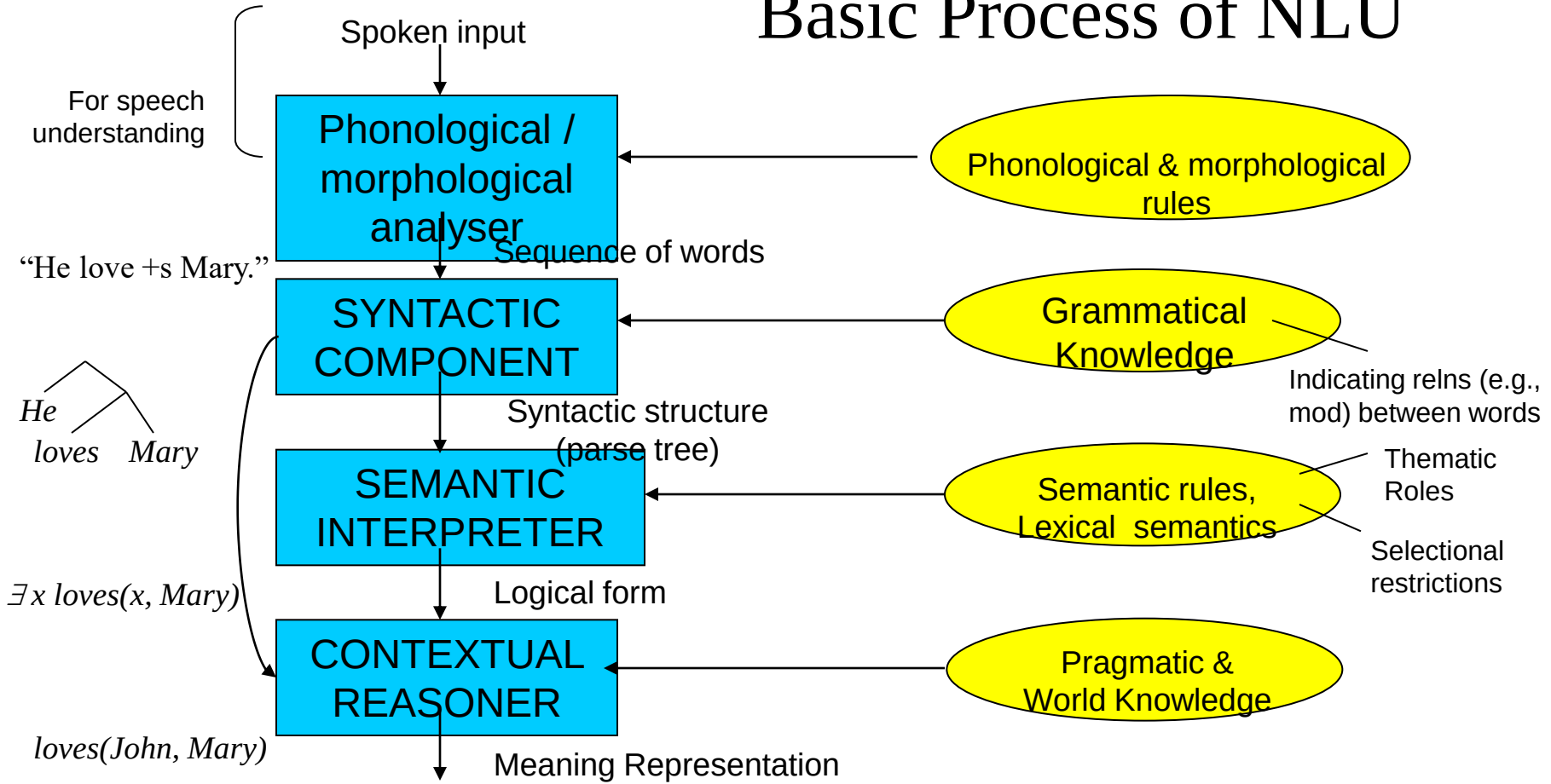
Regular Expressions

Natural Language Understanding

- Associate each input (acoustic signal/character string) with a meaning representation.

- Carried out by a series of components:
  - Each component acts as a translator from one representation to another
  - In general, each component adds successively 'richer' information to the output

Basic Process of NLU



Representations and Algorithms for NLP

- Representations: formal models used to capture linguistic knowledge
- Algorithms manipulate representations to analyze or generate linguistic phenomena
- Simplest often produce best performance but….the 80/20 Rule and "low-hanging fruit"

NLP Representations

- State Machines
  - FSAs, FSTs, HMMs, ATNs, RTNs
- Rule Systems
  - CFGs, Unification Grammars, Probabilistic CFGs
- Logic-based Formalisms
  - 1st Order Predicate Calculus, Temporal and other Higher Order Logics
- Models of Uncertainty
  - Bayesian Probability Theory

NLP Algorithms

- Most are parsers or transducers: accept or reject input, and construct new structure from input
  - State space search
    - Pair a partial structure with a part of the input
    - Spaces too big and 'best' is hard to define
  - Dynamic programming
    - Avoid recomputing structures that are common to multiple solutions

Today

- Review some of the simple representations and ask ourselves how we might use them to do interesting and useful things
  - Regular Expressions
  - Finite State Automata
- How much can you get out of a simple tool?

Regular Expressions

- Can be viewed as a way to specify:
  - Search patterns over text string
  - Design of a particular kind of machine, called a Finite State Automaton (FSA)

- These are really equivalent

Uses of Regular Expressions in NLP

- As grep, perl: Simple but powerful tools for large corpus analysis and 'shallow' processing
  - What word is most likely to begin a sentence?
  - What word is most likely to begin a question?
  - In your own email, are you more or less polite than the people you correspond with?
- With other unix tools, allow us to
  - Obtain word frequency and co-occurrence statistics
  - Build simple interactive applications (e.g., Eliza)
- Regular expressions define regular languages or sets

Regular Expressions

- **Regular Expression**: Formula in algebraic notation for specifying a set of strings

- **String**: Any sequence of alphanumeric characters
  - Letters, numbers, spaces, tabs, punctuation marks

- **Regular Expression Search**
  - Pattern: specifying the set of strings we want to search for
  - Corpus: the texts we want to search through

Simple Example

- Sequence of simple characters:

RE	DESCRIPTION	USES
/This/	Matches the string “This”	Finding the word “this” starting a sentence
/this/	Matches the string “this”	Finding the word “this” internal to a sentence.
/[Tt]his/	Matches either “This” or “this” -- disjunction	Finding the word “this” anywhere in a sentence.

Some Examples

RE	Description	Uses
/./	Wild card; Any char except <cr>	A non-blank line?
/a/	Any ‘a’	Line with words?
/[ab]/	A choice	
/[a-z]/	l.c. char (range)	Common noun?
/[A-Z]/	u.c. char	Proper noun?
/[^?.!]/	Neg of set	Not S-final punc

RE	Description	Uses?
/a*/	Zero or more a's	Optional doubled modifiers
/a+/	One or more a's	Non-optional...
/a?/	Zero or one a's	Optional...
/cat|dog/	'cat' or 'dog'	Words modifying pets
/^cat\.$/	A line that contains only cat. ^anchors beginning, $ anchors end of line.	??
/\bun\B/	Beginnings of longer strings	Words prefixed by 'un'

RE	E.G.
/pupp(y|ies)/	Morphological variants of ‘puppy’
/ (.+)ier and \1ier /	happier and happier, fuzzier and fuzzier

Optionality and Repetition

- /[Ww]oodchucks?/ matches woodchucks, Woodchucks, woodchuck, Woodchuck
- /colou?r/ matches color or colour
- /he{3}/ matches heee
- /(he){3}/ matches hehehe
- /(he){3,} matches a sequence of at least 3 he's

Operator Precedence Hierarchy

1. Parentheses	()
2. Counters	* + ? {}
3. Sequence of Anchors	the ^my end$
4. Disjunction	|

Examples

/moo+/

/try|ies/

/and|or/

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

/the/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with

southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

  /the/

  /[tT]he/

The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

  /the/

  /[tT]he/

*The recent attempt by the police to retain **the**ir current rates of pay has not ga**the**red much favor with the sou**the**rn factions.*

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

  /the/

  /[tT]he/

  /\b[tT]he\b/

The recent attempt by <u>the</u> police to retain their current rates of pay has not gathered much favor with <u>the</u> southern factions.

A Simple Exercise

- Write a regular expression to find all instances of the determiner "the":

  /the/

  /[tT]he/

  /\b[tT]he\b/

  /(^|[^a-zA-Z])[tT]he[^a-zA-Z]/

  The recent attempt by the police to retain their current rates of pay has not gathered much favor with the southern factions.

The Two Kinds of Errors

- The process we just went through was based on fixing errors in the regular expression
  - Errors where some of the instances were missed (judged to not be instances when they should have been) – False negatives
  - Errors where the instances were included (when they should not have been) – False positives
- This is pretty much going to be the story of the rest of the course!

Substitutions (Transductions)

- Sed or ‘s’ operator in Perl
  - s/regexp1/pattern/
  - s/I am feeling (.++)/You are feeling \1?/
  - s/I gave (.+) to (.+)/Why would you give \2 \1?/

ELIZA: Substitutions Using Memory

User: Men are all alike.

ELIZA: IN WHAT WAY

s/.* all .*/IN WHAT WAY/

They're always bugging us about something or other.

ELIZA: CAN YOU THINK OF A SPECIFIC EXAMPLE

s/.* always .*/CAN YOU THINK OF A SPECIFIC EXAMPLE/

User: My boyfriend says I'm depressed.

ELIZA: I AM SORRY TO HEAR YOU ARE DEPRESSED

s/.* I'm (depressed|sad) .*/I AM SORRY TO HEAR YOU ARE \1/

Using RE's: Examples

- Predictions from a news corpus:
  - Which candidate for Governor of California is mentioned most often in the news? Is going to win?
  - What stock should you buy?
  - Which White House advisers have the most power?
- Language use:
  - Which form of comparative is more frequent: 'oftener' or 'more often'?
  - Which pronouns are conjoined most often?
  - How often do sentences end with infinitival 'to'?
  - What words most often begin and end sentences?
  - What's the most common word in your email? Is it different from your neighbor?

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

FINITE-STATE AUTOMATA

Introduction

- Pieces are set up on a playing board; dice are thrown or a wheel is spun, and a number is generated at random.
- Based on the number appearing on the dice, the pieces on the board are rearranged specified by the rules of the game.
- Consider all possible positions of the pieces on the board and call them states. The state in which the game begins is termed the initial state.
- The state corresponding to the winning position is termed the final state.
- When the game is over this will be the final state.

- Now consider a very simple machine with an input device, a processor, some memory and an output device. The machine starts in the initial state.
- It checks the input and goes to next state, which is completely determined by the prior state and the input. If all goes well, the machine reaches final state and terminates.
- If the machine gets an input for which the next state is not specified, and it gets stuck at a non-final state, we say the machine has failed or rejected the input.

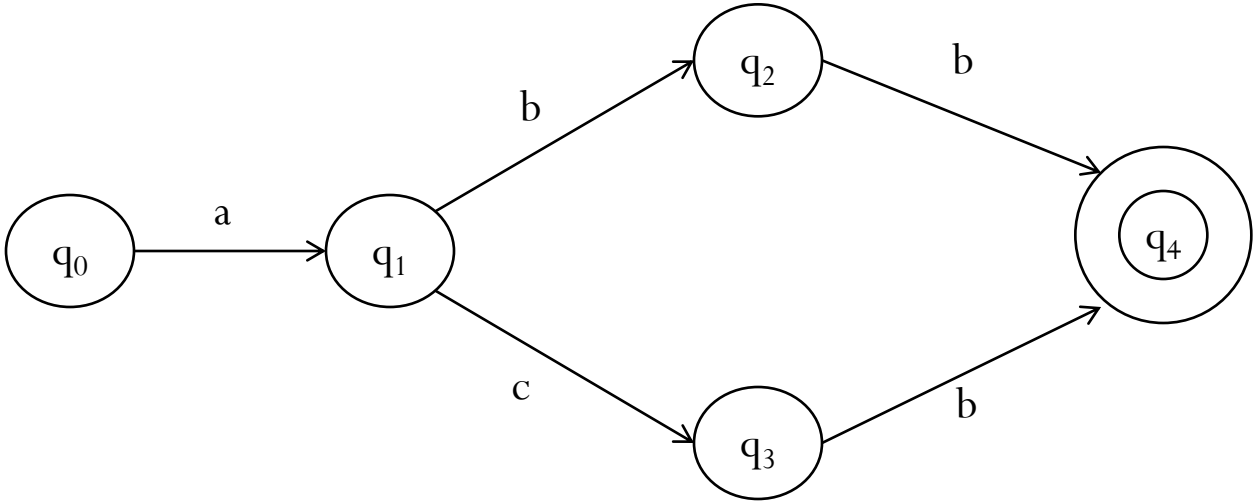
Finite Automation Properties

- A finite set of states, one of which is designated the initial or start state, and one or more of which are designated as the final states.
- A finite alphabet set, Σ, consisting of input symbols.
- A finite set of transitions that specify for each state and each symbol of the input alphabet, the state to which it next goes.

EXAMPLE

- Suppose $\Sigma = \{a, b\}$, the set of states $= \{q_0, q_1, q_2, q_3, q_4\}$ with q_0 being the start state and q_4 the final state, we have the following rules of transition:
- From state q_0 and with input a, go to state q_1.
- From state q_1 and with input b, go to state q_2.
- Form state q_1 and with input a, go to state q_3.
- Form state q_2 and with input b, go to state q_4.
- Form state q_3 and with input b, go to state q_4.

DFA



Finite State Automata

FSA has been used in wide variety of areas:

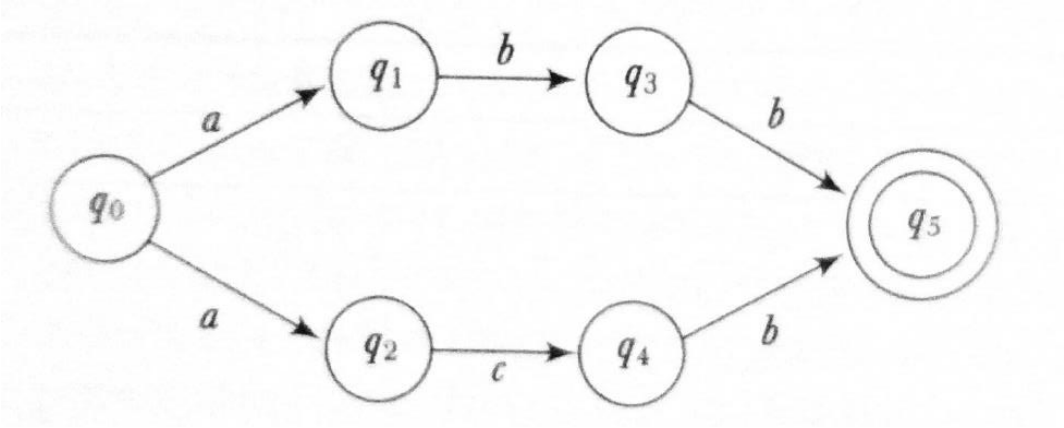
- Lingusitics
- Electrical engineering
- Computer Science
- Mathematics
- Logic

Deterministic Finite State Automaton

A DFA is defined as a 5-tuple $(Q, \Sigma, \delta, S, F)$ where

- Q is a set of states
- Σ is an alphabet
- δ is a transition function
- The transition function defines mapping from $Q \times \Sigma$ to Q.
- That is for each state q and symbol a, there is at most one transition possible.
- In NFA maps $Q \times (\Sigma \cup \{\varepsilon\})$ to a subset of the power set of Q.
- That is for each state there can be more than one transition on a given symbol, each leading to different state.

NFA



- A path is a sequence of transitions beginning with the start state.
- A path leading to one of the final states is a successful path.
- The FSAs encode regular languages.

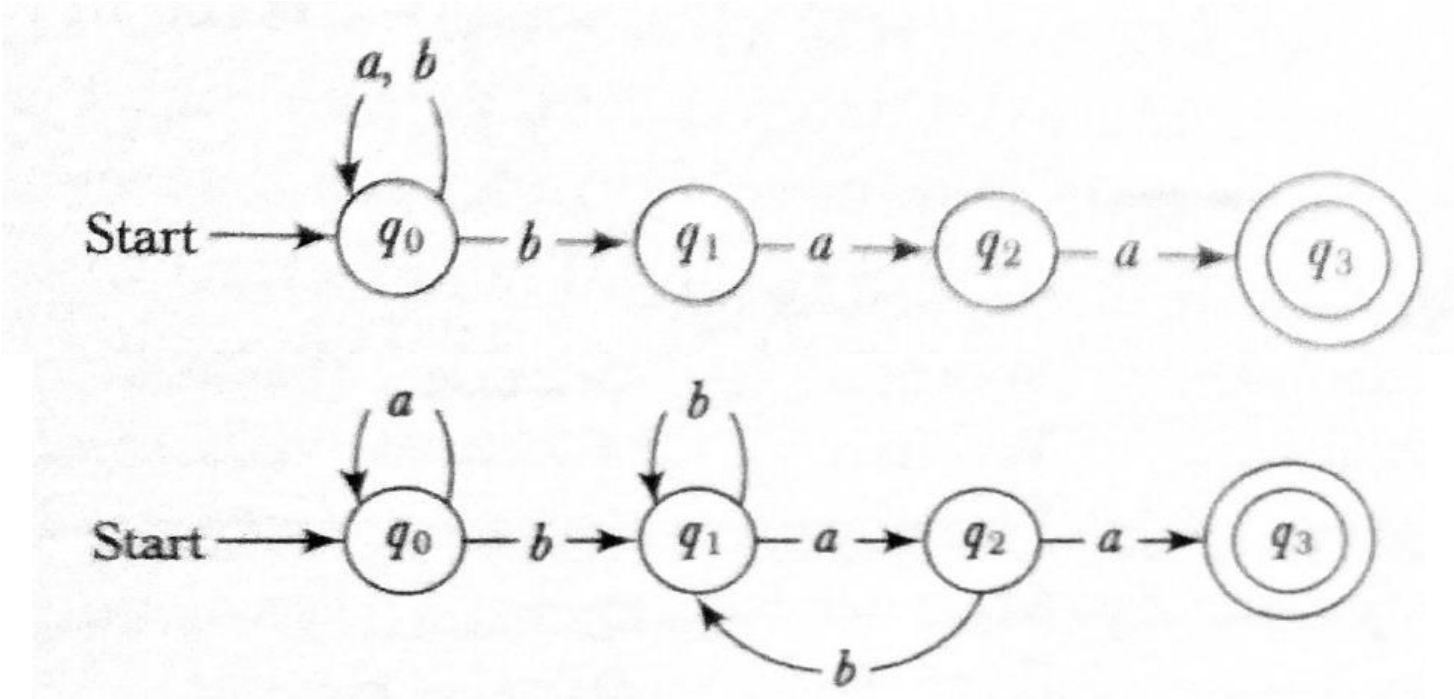
State Transition Table for DFA

		Input		
State		***a***	***b***	*c*
start:	q_0	q_1	ϕ	ϕ
	q_1	ϕ	q_2	q_3
	q_2	ϕ	q_4	ϕ
	q_3	ϕ	q_4	ϕ
final:	q_4	ϕ	ϕ	ϕ

Transition Table for NFA

State		Input	
		a	b
start:	q_0	$\{q_0\}$	$\{q_0, q_1\}$
	q_1	$\{q_2\}$	ϕ
	q_2	$\{q_3\}$	ϕ
final:	q_3	ϕ	ϕ

NFA for the regular expression /(a|b)*baa$/



References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Morphological Parsing

Introduction

- Morphology is a sub discipline of linguistics.
- It studies word structure and the formation of words from smaller units.
- The goal of the morphological parsing is to discover the morphemes that build a given word.
- Morphemes are the smallest meaning-bearing units in a language.
- Example: bread, eggs.

Contd..

- There are two broad classes of morphemes: **stems** and **affixes**.
- The stem is the main morpheme. i.e. the morpheme that contains the central meaning.
- Affixes modify the meaning given by the stem. It is divided into prefix, suffix, infix and circumfix.
- Prefixes are morphemes which appear before a stem. Ex: bewaqt
- Suffixes are morphemes which applied at end of the stem. Ex: ghodhon
- Circumfix are morphemes that may be applied to either end of the stem.
- Infixes are morphemes that appear inside a stem.

A Sample Lexicon Entry

Word form	Category	Root	Gender	Number	Person
Ghodhaa	noun	GhoDaa	masculine	singular	3rd
Ghodhii	-do-	-do-	feminine	-do-	-do-
Ghodhon	-do-	-do-	masculine	plural	-do-
Ghodhe	-do-	-do-	-do-	-do-	-do-

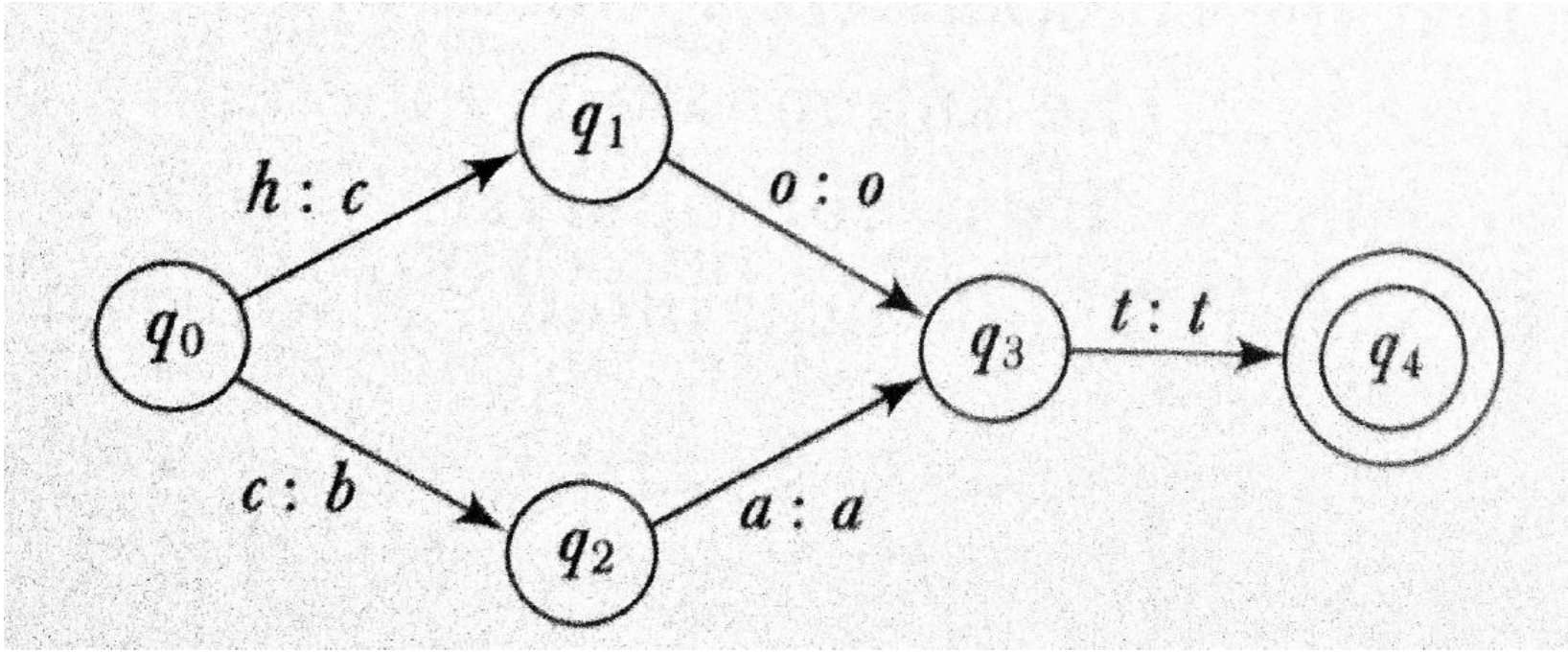
A Morphological Parser

- Lexicon: A lexicon list stems and affixes together with basic information about them.
- Morphotactics: There exists certain ordering among the morphemes that constitute a word. They cannot be arranged arbitrarily. This deals with the ordering of morphemes.
- Orthographic rules: These are spelling rules that specify the changes that occur when two given morphemes combine. Example: easy to easier.

Stemming algorithms

- (i) Suffix removal:- This steps removes predefined endings from words.
- (ii) Recording:- This steps adds predefined endings to the output of the first step.

Finite State Transducer



Two Step Morphological Parser

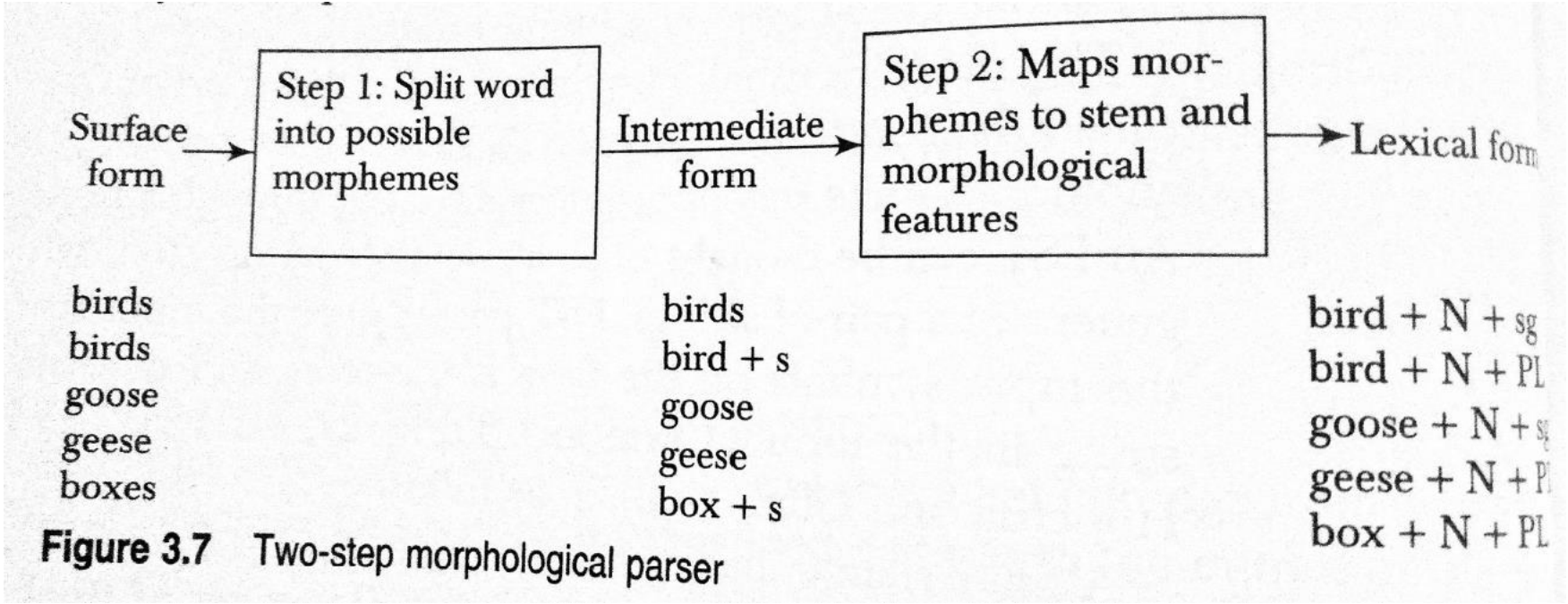
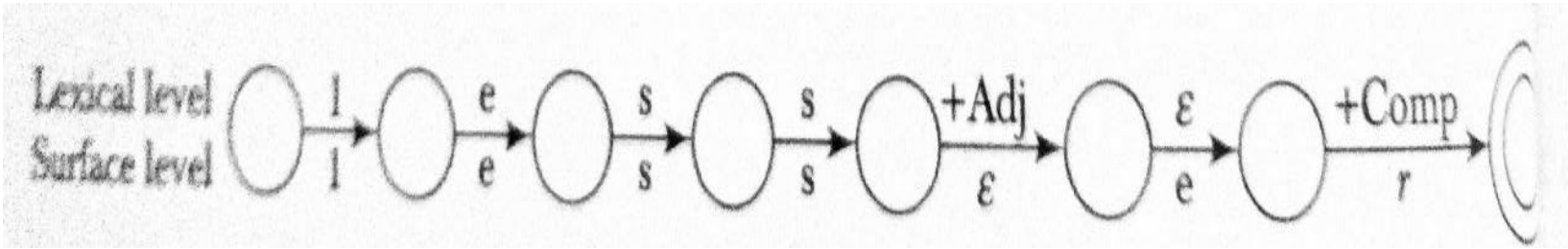
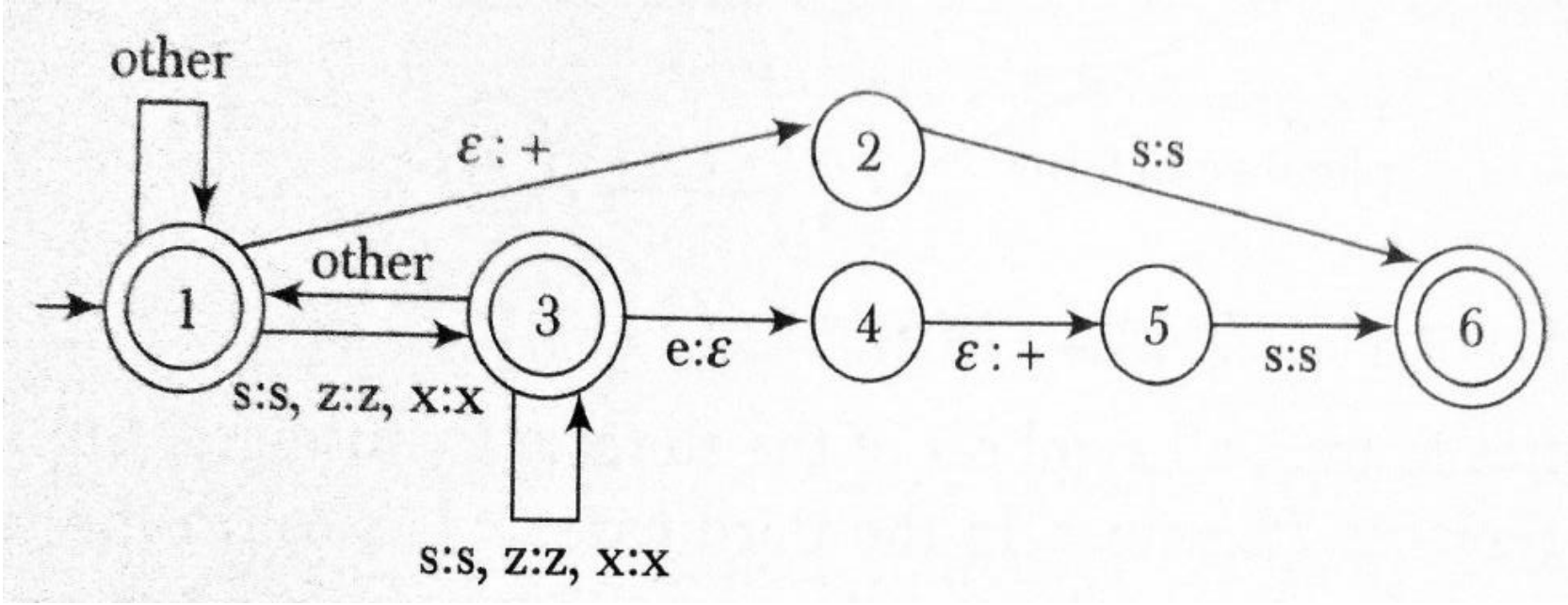


Figure 3.7 Two-step morphological parser

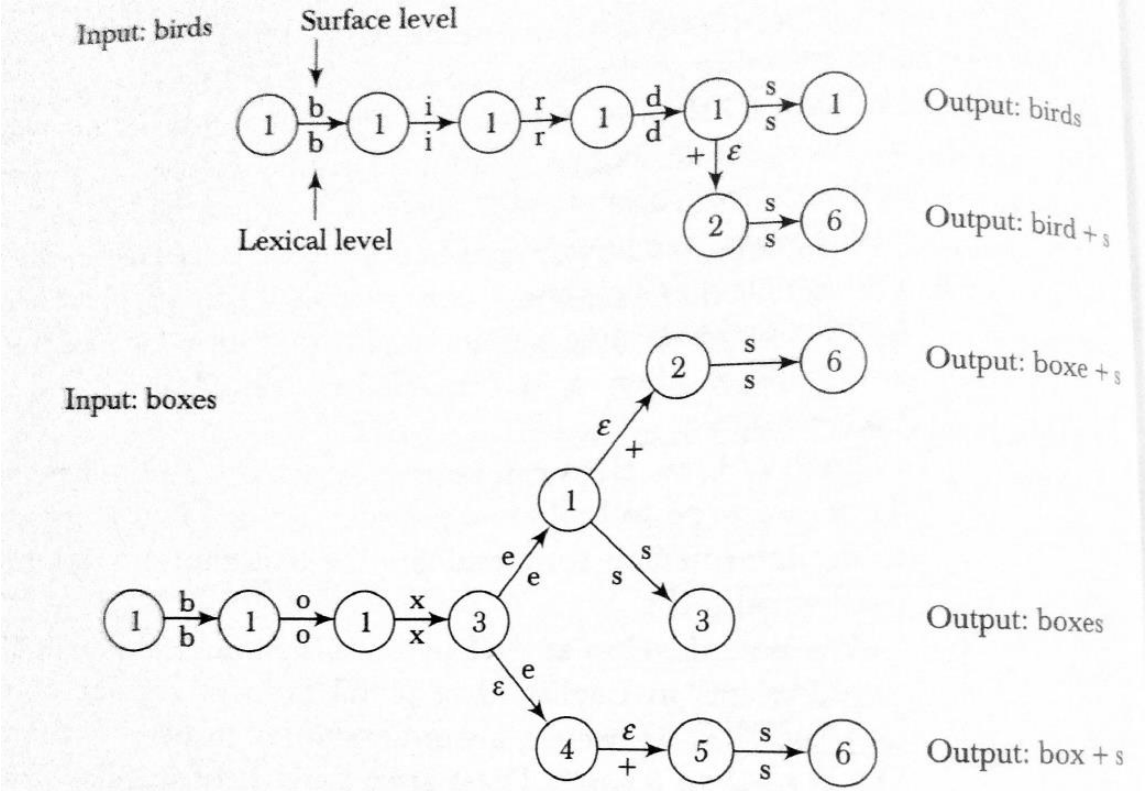
Simple FST



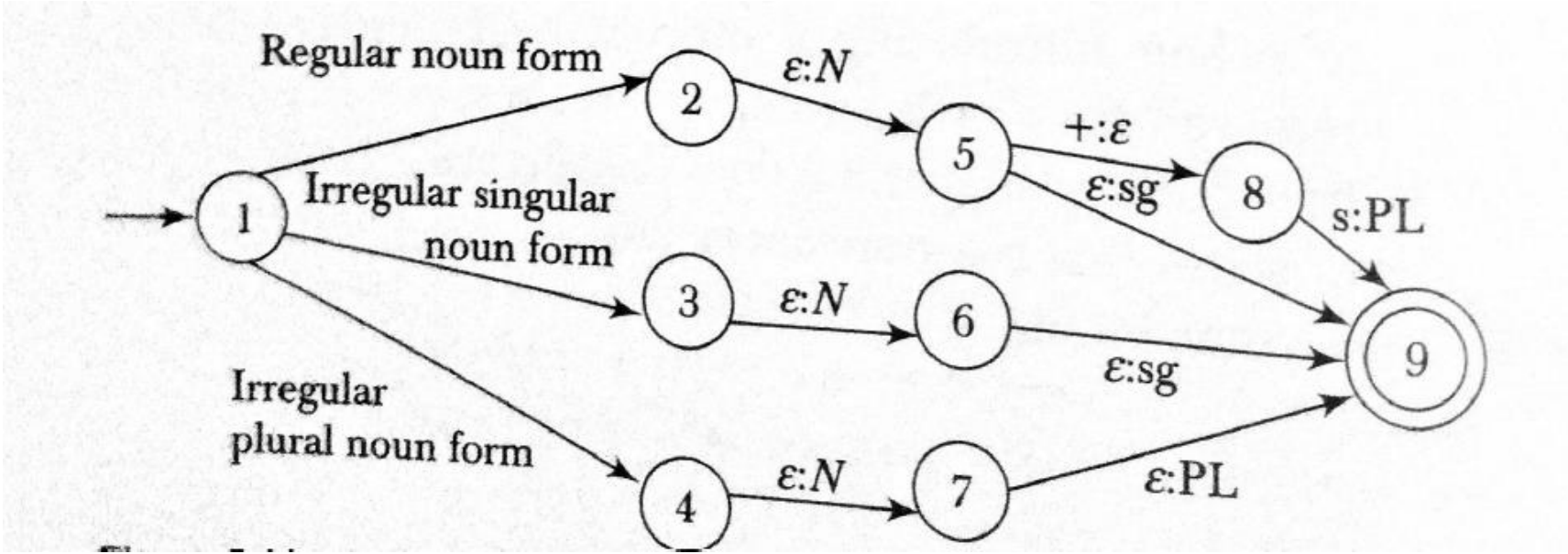
Simplified FST, Mapping English nouns to the intermediate form



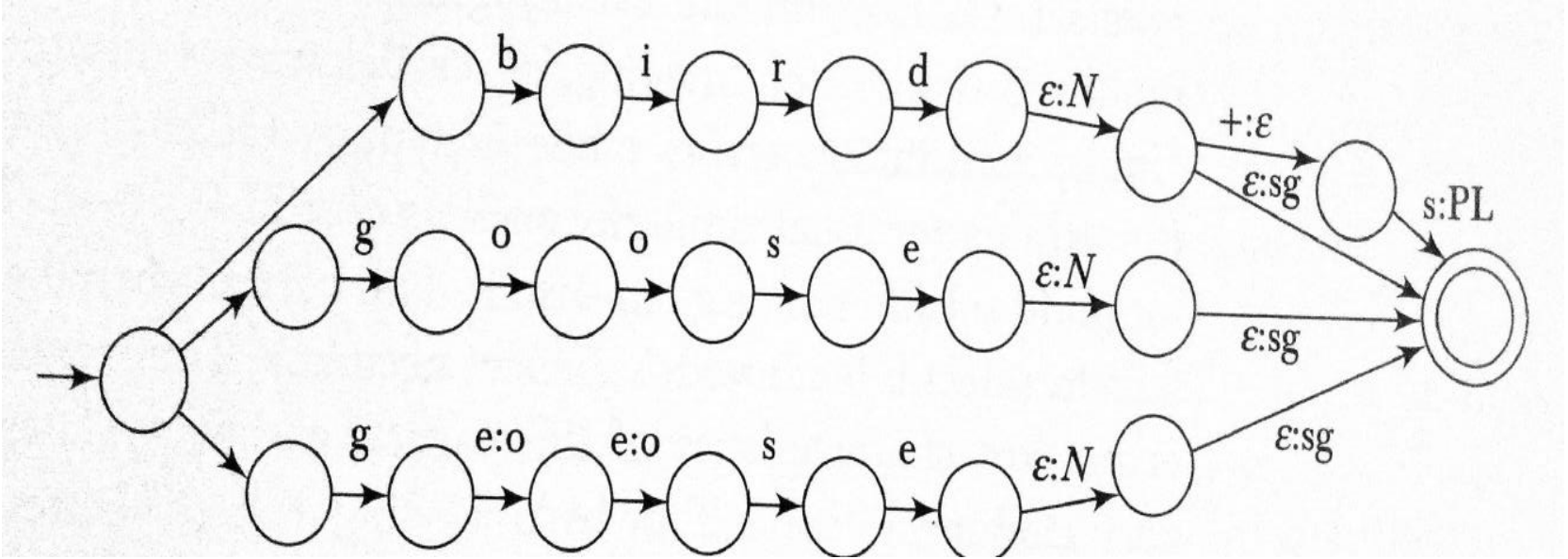
Possible sequences of states



Transducer for Step2



A transducer mapping nouns to their stem



References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

SPELLING ERRO DETECTION AND CORRECTION

Introduction

- In computer base information systems, error of typing and spelling constitute a very common source of variation between strings.
- These errors have been widely investigated. All investigations agree that single character omission, insertion, substitution and reversal are the most common typing mistakes.
- Damearu(1964) reported that over 80% of the typing errors were single-error misspellings:
- (1)substitution of a single letter.
- (2)omission of a single letter.
- (3)insertion of a single letter.
- (4)transposition of tw0 adjacent letters.

Contd..

- Shafer and Hardwick (1968) found that the most common type of single character error was substitution, followed by omission of a letter, and then insertion of a letter.
- Single character omission occurs when a single character is missed, e.g. when ‘concept’ is accidentally typed as ‘concpt’.
- Insertion error refers to the presence of an extra character in a word, e.g. when ‘error’ is misspell as ‘errorn’.
- Substitution error occurs when a wrong letter is typed in place of the right one, as in ‘errpr’ where ‘p’ appears in place of ‘o’.
- Reversal refers to a situation in which the sequence of characters is reversed, e.g., ‘aer’ instead of ‘are’.

Contd..

- Optical character recognition (OCR) and other automatic reading devices introduce errors of substitution, deletion, and insertion but not of reversal.
- OCR errors are usually grouped into five classes: substitution, multi-substitution, space deletion or insertion and failures.
- OCR substitution errors are caused due to visual similarity such as c→ e, 1→l, r→n.
- Multi-substitution – e.g. m→rn.
- These errors can be corrected using 'context' or by using linguistic structures.

Contd..

- Speech recognition deal with strings of phonemes and attempt to match a spoken utterance with a dictionary of known utterances.
- Spelling errors are mainly phonetic, where the misspelled word is pronounced in the same way as the correct word.
- Spelling errors belong to one of the two distinct categories:
- non-word: error results in a word that does not appear in a given lexicon. Detection can be done using n-gram and dictionary lookup.
- real word errors: error results in actual words of the language. It occurs due to typographical mistakes or spelling errors.

Contd..

- Spelling correction consists of detecting and correcting errors.
- Error detection is the process of finding misspelled words.
- Error correction is the process of suggesting correct words to a misspelled one.
- These sub-problems are addressed in two ways:
- Isolated-error detection and correction.
- Context-dependent error detection and correction.

Isolated-error detection and correction

- Each word is checked separately, independent of its context.
- Detecting whether or not a word is correct seems simple-There are number of problems associated with this simple strategy:
- The strategy requires the existence of a lexicon containing all correct words. Such lexicon will take long time to compile and occupy a lot of space.
- Some languages are highly productive. It is impossible to list all the correct words of such languages.
- This strategy fails when spelling error produces a word that belongs to the lexicon, e.g., when 'theses' is written in place of 'these'.
- The larger the lexicon, the more likely it is that an error goes undetected, because the chance of a word being found is greater in a large lexicon.

Context dependent error detection and correction

- It utilizes the context of a word to detect and correct errors.
- Grammatical analysis is required and it is more complex and language dependent.
- The list of candidate words must first be obtained using an isolated method before making a selection depending on the context.

Spelling Correction Algorithms

- **Minimum Edit Distance**: The minimum edit distance b/w two strings is the minimum number of operations required to transform one string into another.
- **Similarity Key techniques**: The basic idea is to change a given string into a key such that similar strings will change into the same key. Example: SOUNDEX.
- **n-gram based techniques**: It is used for both non-word and real-word error detection.
- **Neural nets**: These have the ability to do associative recall based on incomplete and noisy data. They can be trained to adapt to specific spelling error patterns. Computationally expensive.
- **Rule-based techniques**: A set of rules derived from knowledge of a common spelling error pattern is used to transform misspelled words into valid words.

Minimum Edit Distance

- The minimum edit distance is the number of insertions, deletions and substitutions required to change one string into another.
- For example: the minimum edit distance between ‘tutor’ and ‘tumour’ is 2. we substitute m for t and insert u before r.

t u **t** o - r

t u **m** o u r

MED Algorithm

Input: Two strings, X and Y.

Output: The minimum edit distance between X and Y.

```
m ← length(X)
n ← length(Y)
for i=0 to m do
        dist[i,0] ← i
for j=0 to n do
        dist[0,j] ← j
for i = 1 to m do
    for j = 1 to n do
        dist[i,j] = MIN{ dist[i-1,j]+insert_cost,
                         dist[i-1,j-1]+subst_cost(Xi,Yi),
                         dist[i,j-1]+delete_cost}
```

Problem

	#	t	u	m	o	u	r
#							
t							
u							
t							
o							
r							

References

- https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm
- https://www.javatpoint.com/nlp

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Part of Speech Tagging

New Topic: Syntax

- Up until now we have been dealing with individual words and simple-minded (though useful) notions of what sequence of words are likely.
- Now we turn to the study of how words
  - Are clustered into classes
  - Group with their neighbors to form phrases and sentences
  - Depend on other words
- Interesting notions:
  - Word order
  - Consittuency
  - Grammatical relations
- Today: syntactic word classes – part of speech tagging

What is a word class?

- Words that somehow 'behave' alike:
  - Appear in similar contexts
  - Perform similar functions in sentences
  - Undergo similar transformations

Why do we want to identify them?

- Someone say
  - Refuse
  - Project
  - Compact
  - Content
  - Discount
- Why do we want to identify them?
  - Pronunciation (desert/desert)
  - Stemming
  - Semantics
  - More accurate N-grams
  - Simple syntactic information

How many word classes are there?

- A basic set:
  - N, V, Adj, Adv, Prep, Det, Aux, Part, Conj, Num
- A simple division: open/content vs. closed/function
  - Open: N, V, Adj, Adv
  - Closed: Prep, Det, Aux, Part, Conj, Num
- Many subclasses, e.g.
  - eats/V $\Rightarrow$ eat/VB, eat/VBP, eats/VBZ, ate/VBD, eaten/VBN, eating/VBG, …
  - Reflect morphological form & syntactic function

How do we decide which words go in which classes?

- Nouns denote people, places and things and can be preceded by articles? But…

  My typing is very bad.

  *The Mary loves John.

- Verbs are used to refer to actions and processes
  - But some are closed class and some are open

  I will have emailed everyone by noon.

- Adjectives describe properties or qualities, but

  a cat sitter, a child seat

- Adverbs include locatives (here), degree modifiers (very), manner adverbs (gingerly) and temporals (today)
  - Is Monday a temporal adverb or a noun?
- Closed class items (Prep, Det, Pron, Conj, Aux, Part, Num) are easier, since we can enumerate them….but
  - Part vs. Prep
    - George eats up his dinner/George eats his dinner up.
    - George eats up the street/*George eats the street up.
  - Articles come in 2 flavors: definite (the) and indefinite (a, an)

- Conjunctions also have 2 varieties, coordinate (and, but) and subordinate/complementizers (that, because, unless,…)
- Pronouns may be personal (I, he,…), possessive (my, his), or wh (who, whom,…)
- Auxiliary verbs include the copula (be), do, have and their variants plus the modals (can, will, shall,…)

- And more…
  - Interjections/discourse markers
  - Existential there
  - Greetings, politeness terms

Tagsets

- What set of parts of speech do we use?
- Most tagsets implicitly encode fine-grained specializations of 8 basic parts of speech (POS, word classes, morphological classes, lexical tags):
  - Noun, verb, pronoun, preposition, adjective, conjunction, article, adverb
- These categories are based on morphological and distributional similarities and not, as you might think, semantics.
- In some cases, tagging is fairly straightforward (at least in a given language), in other cases it is not.

Distribution of Tags

- Parts of speech follow the usual frequency-based distributional behavior
  - Most word types have only one part of speech
  - Of the rest, most have two
  - A small number of word types have lots of parts of speech
  - Unfortunately, the word types with lots of parts of speech occur with high frequency (and words that occur most frequently tend to have multiple tags)

Distribution of Tags – Brown

- To see the problem:
  - 11.5% of English words in the Brown corpus are ambigjous
  - 40% of tokens in the Brown corpus are ambignous

Unambiguous	(1 tag)	35,340	
Ambiguous	(2-7 tags)	4,100	
	2 tags	3,760	
	3 tags	264	
	4 tags	61	
	5 tags	12	
	6 tags	2	
	7 tags	1	(“still”)

The Brown Corpus

- The Brown Corpus of Standard American English was the first of the modern, computer readable general corpora. (Compiled at Brown University)
- Corpus consists of 1 million words of American English text printed in 1961.
- For a long time, Brown and LOB (British) corpora were the only easily available online, so many studies have been done on these corpora.
- Studying the same data allows comparison of findings without having to take into consideration possible variation caused by the use of different data.
- But…?
- Tagged version of Brown is available.

Tagsets

- There are various standard tagsets to choose from; some have a lot more tags than others
- The choice of tagset is based on the application
- Accurate tagging can be done with even large tagsets

So how do we choose a Tagset?

- http://www.comp.leeds.ac.uk/amalgam/tagsets/tagmenu.html

- Brown Corpus (Francis & Kucera '82), 1M words, 87 tags.
  - http://www.comp.leeds.ac.uk/amalgam/tagsets/brown.html
- Penn Treebank: hand-annotated corpus of *Wall Street Journal*, 1M words, 45-46 tags
  - http://www.comp.leeds.ac.uk/amalgam/tagsets/upenn.html

Tagsets

- How do tagsets differ?
  - Degree of granularity
  - Idiosyncratic decisions, e.g. Penn Treebank doesn't distinguish to/Prep from to/Inf, eg.
  - I/PP want/VBP to/TO go/VB to/TO Zanzibar/NNP ./.
  - Don't tag it if you can recover from word (e.g. do forms)

What does Tagging do?

1. Collapses distinctions
   - E.g., all personal pronouns tagged as PRP
   - Lexical identity may be completely discarded

2. Introduces distinctions (by reducing ambiguity)
   - E.g., deal tagged with NN or VB

Tagging

- Part of speech tagging is the process of assigning parts of speech to each word in a sentence

- Assume we have
  - A tagset
  - A dictionary that gives you the possible set of tags for each entry
  - A text to be tagged
- Output
  - Single best tag for each word
  - E.g., Book/VB that/DT flight/NN

Part-of-Speech Tagging

- How do we assign POS tags to words in a sentence?
  - Get/V the/Det bass/N
  - Time flies like an arrow.
  - Time/[V,N] flies/[V,N] like/[V,Prep] an/Det arrow/N
  - Time/N flies/V like/Prep an/Det arrow/N
  - Fruit/N flies/N like/V a/DET banana/N
  - Fruit/N flies/V like/V a/DET banana/N
  - The/Det flies/N like/V a/DET banana/N

Just for Fun…

- Using Penn Treebank tags, tag the following sentence from the Brown Corpus:

- The grand jury commented on a number of other topics.

Just for Fun…

- Using Penn Treebank tags, tag the following sentence from the Brown Corpus:

- The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

Why is Tagging Hard?

- Example
  - Book/VB that/DT flight/NN
  - Does/VBZ that/DT flight/NN serve/VB dinner/NN

- Tagging is a type of disambiguation
  - Book can be NN or VB
  - Can I read a book on this flight?

  - That can be a DT or complementizer
  - My travel agent said that there would be a meal on this flight.

Potential Sources of Disambiguation

- Many words have only one POS tag (e.g. is, Mary, very, smallest)
- Others have a single most likely tag (e.g. a, dog)
- But tags also tend to co-occur regularly with other tags (e.g. Det, N)
- In addition to conditional probabilities of words $P(w_1 \mid w_{n-1})$, we can look at POS likelihoods $P(t_1 \mid t_{n-1})$ to disambiguate sentences and to assess sentence likelihoods

Approaches to POS Tagging

- Rule-based Approach
  - Uses handcrafted sets of rules to tag input sentences
- Statistical approaches
  - Use training corpus to compute probability of a tag in a context
- Hybrid systems (e.g. Brill's transformation-based learning)

ENGTWOL Rule-Based Tagger

A Two-stage architecture

- Use lexicon FST (dictionary) to tag each word with all possible POS
- Apply hand-written rules to eliminate tags.

- The rules eliminate tags that are inconsistent with the context, and should reduce the list of POS tags to a single POS per word.

Det-Noun Rule:

- If an ambiguous word follows a determiner, tag it as a noun

ENGTWOL Adverbial-that Rule

Given input "that"

- **If** the next word is adj, adverb, or quantifier, and following that is a sentence boundary, and the previous word is not a verb like "consider" which allows adjs as object complements,
- **Then** eliminate non-ADV tags,
- **Else** eliminate ADV tag

- I consider that odd. (that is NOT ADV)
- It isn't that strange. (that is an ADV)

Does it work?

- This approach does work and produces accurate results.

- What are the drawbacks?
  - Extremely labor-intensive

Statistical Tagging

- Statistical (or stochastic) taggers use a training corpus to compute the probability of a tag in a context.
- For a given word sequence, Hidden Markov Model (HMM) Taggers choose the tag sequence that maximixes

$$P(\text{word} \mid \text{tag}) * P(\text{tag} \mid \text{previous-n-tags})$$

A bigram HMM tagger chooses the tag t_i for word w_i that is most probable given the previous tag, t_{i-1}

$$t_i = \text{argmax}_j \, P(t_j \mid t_{i-1}, w_i)$$

Statistical POS Tagging (Allen95)

- Let's step back a minute and remember some probability theory and its use in POS tagging.
- Suppose, with no context, we just want to know given the word "flies" whether it should be tagged as a noun or as a verb.
- We use conditional probability for this: we want to know which is greater

  PROB(N | flies) or PROB(V | flies)
- Note definition of conditional probability

  PROB(a | b) = PROB(a & b) / PROB(b)

  - Where PROB(a & b) is the probability of the two events a dn b occurring simultaneously

Calculating POS for “flies”

We need to know which is more

- PROB(N | flies) = PROB(flies & N) / PROB(flies)
- PROB(V | flies) = PROB(flies & V) / PROB(flies)

- Count on a Corpus

Doing Better

- Simple Method: Always choose the tag that appears most frequently in the training set – will work correctly about 91% of the time.
- How to do better: Consider more of the context. Knowing "the flies" gives much higher probability of a Noun
- General Equation: find the sequence of tags that maximizes:

$$\text{PROB}(T_1,\ldots T_n \mid w_1,\ldots w_n)$$

Estimating Too Hard

$$PROB(T_1,\ldots T_n \mid w_1,\ldots w_n)$$

Estimating the above takes far too much data. Need to do some reasonable approximations.

Bayes Rule:

$$PROB(A \mid B) = PROB(B \mid A) * PROB(A) / PROB(B)$$

Rewriting:

$$PROB(w_1,\ldots w_n \mid T_1,\ldots T_n) * PROB(T_1,\ldots T_n) / PROB(w_1,\ldots w_n)$$

$PROB(w_1,\ldots w_n \mid T_1,\ldots T_n) * PROB(T_1,\ldots T_n) / PROB(w_1,\ldots w_n)$

Remember we are interested in finding the sequence of tags that maximizes this formula – so we can ignore $PROB(w_1,\ldots w_n)$ since it is always the same

So, we want to find the sequence of tags that maximizes

$$PROB(w_1,\ldots w_n \mid T_1,\ldots T_n) * PROB(T_1,\ldots T_n)$$

This is still too hard to calculate – so we need to make some independence assumptions.

Independence Assumptions

So, we want to find the sequence of tags that maximizes

$$\text{PROB}(T_1,\ldots T_n) * \text{PROB}(w_1,\ldots w_n \mid T_1,\ldots T_n)$$

For Tags – use bigram probabilities

$$\text{PROB}(T_1,\ldots T_n) \approx \pi_{i=1,n}\, \text{PROB}(T_i \mid T_{i-1})$$

$$\text{PROB}(\text{ART N V N}) \approx \text{PROB}(\text{ART} \mid \Phi) * \text{PROB}(\text{N} \mid \text{ART}) * \text{PROB}(\text{V} \mid \text{N}) * \text{PROB}(\text{N} \mid \text{V})$$

For second probability: assume word tag is independent of words around it:

$$\text{PROB}(w_1,\ldots w_n \mid T_1,\ldots T_n) \approx \pi_{i=1,n}\, \text{PROB}(w_i \mid T_i)$$

POS Formula

- Find the sequence of tags that maximizes:

$$\pi_{i=1,n} \text{PROB}(T_i \mid T_{i-1}) * \text{PROB}(w_i \mid T_i)$$

- These probabilities can be estimated from a corpus of text labeled with parts of speech. (See handout which takes us through calculating for whole sequence – to slide 47)

Statistical Tagging (cont.)

- Making some simplifying Markov assumptions, the basic HMM equation for a single tag is:

$$t_i = \text{argmax}_j \, P(t_j \mid t_{i-1}) * P(w_i \mid t_j)$$

  - The function $\text{argmax}_x F(x)$ means "the x such that F(x) is maximized"
  - The first P is the tag sequence probability, the second is the word likelihood given the tag.
- Most of the better statistical models report around 95% accuracy on standard datasets
- But, note you get 91% accuracy just by picking the most likely tag!

Statistical POS Tagging (whole sequence)

- Goal: choose the best sequence of tags T for a sequence of words W in a sentence

$$T' = \arg\max_{T \in \tau} P(T|W)$$

  - By Bayes Rule (giving us something easier to calculate)

$$P(T|W) = \frac{P(T)P(W|T)}{P(W)}$$

$$T' = \arg\max_{T \in \tau} P(T)P(W|T)$$

  - Since we can ignore P(W), we have

Statistical POS Tagging: the Prior

$P(T) = P(t_1, t_2, \ldots, t_{n-1}, t_n)$

By the Chain Rule:

$= P(t_n \mid t_1, \ldots, t_{n-1})\, P(t_1, \ldots, t_{n-1})$

$= \prod_{i=1}^{n} P(t_i | t_1^{i-1})$

Making the Markov assumption:

$\approx P(t_i | t_{i-N+1}^{i-1})$ e.g., for bigrams, $\prod_{i=1}^{n} P(t_i | t_{i-1})$

Statistical POS Tagging: the (Lexical) Likelihood

$P(W|T) = P(w_1, w_2, \ldots, w_n \mid t_1, t_2, \ldots, t_n)$

From the Chain Rule:

$$= \prod_{i=1}^{n} P(w_i|w_1t_1 \ldots w_{i-1}t_{i-1}t_i)$$

Simplifying assumption: probability of a word depends only on its own tag $P(w_i|t_i) \approx \prod_{i=1}^{n} P(w_i|t_i)$

So... $T' = \arg\max_{T \in \tau} \prod_{i=1}^{n} P(t_i|t_{i-1}) \prod_{i=1}^{n} P(w_i|t_i)$

Estimate the Tag Priors and the Lexical Likelihoods from Corpus

- Maximum-Likelihood Estimation
  - For bigrams:

$$P(t_i \mid t_{i-1}) = c(t_{i-1}, t_i) / c(t_{i-1})$$

$$P(w_i \mid t_i) = \frac{c(w_i t_i)}{c(t_i)}$$

Performance

- This method has achieved 95-96% correct with reasonably complex English tagsets and reasonable amounts of hand-tagged training data.

Transformation-Based (Brill) Tagging

A hybrid approach

- Like rule-based taggers, this tagging is based on rules
- Like (most) stochastic taggers, rules are also automatically induced from hand-tagged data

Basic Idea: do a quick and dirty job first, and then use learned rules to patch things up

Overcomes the pure rule-based approach problems of being too expensive, too slow, too tedious etc…

An instance of Transformation-Based Learning.

Transformation-Based Tagging

- Combine rules and statistics…
- Start with a dumb statistical system and patch up the typical mistakes it makes.
- How dumb?
  - Assign the most frequent tag (unigram) to each word in the input

Examples

- Race
  - "race" as NN: .98
  - "race" as VB: .02
- So you'll be wrong 2% of the time, which really isn't bad
- Patch the cases where you know it has to be a verb
  - Change NN to VB when previous tag is TO

TBL tagging algorithm

Input: Tagged corpus and lexicon

1. Label every word with its most likely tag.
2. Examine every possible transformation, and select the one that results in the most improved tagging.
3. Re-tag the data according to the selected rule.

Go to 2 until stopping criterion is reached.

Stopping:

Insufficient improvement over previous pass.

Output: Ranked sequence of transformation rules. These constitute a tagging procedure.

Rules

- Where did that transformational rule come from?
- In principle, the set of possible rules is infinite.
  - Use set of rule templates to define possible rules to try in the search.

Hypothesis Space

- In Brill tagging it's defined by a set of templates of the form
  - Change tag a to tag b when …

The preceding (following) word is tagged z.
The word two before (after) is tagged z.
One of the two preceding (following) words is tagged z.
One of the three preceding (following) words is tagged z.
The preceding word is tagged z and the following word is tagged w.
The preceding (following) word is tagged z and the word two before (after) is tagged w.

- a, b, w and z range over the tags

How?

- Deciding whether or not to accept a transformation depends on the overall change made by the rule.

- If a given tag change rule makes things better (fixes tags that were wrong) should you always accept it?
  - No. It might break things that were right.

Brill Tagging: TBL

- Start with simple (less accurate) rules…learn better ones from tagged corpus
  - Tag each word initially with most likely POS
  - Examine set of transformations to see which improves tagging decisions compared to tagged corpus
  - Re-tag corpus using best transformation
  - Repeat until, e.g., performance doesn't improve
  - Result: tagging procedure (ordered list of transformations) which can be applied to new, untagged text

An Example

The horse raced past the barn fell.

The/DT horse/NN raced/VBN past/IN the/DT barn/NN fell/VBD ./.

1) Tag every word with most likely tag and score

The/DT horse/NN raced/VBD past/NN the/DT barn/NN fell/VBD ./.

2) For each template, try every instantiation (e.g. Change VBN to VBD when the preceding word is tagged NN, add rule to ruleset, retag corpus, and score

3) Stop when no transformation improves score

4) Result: set of transformation rules which can be applied to new, untagged data (after initializing with most common tag)

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

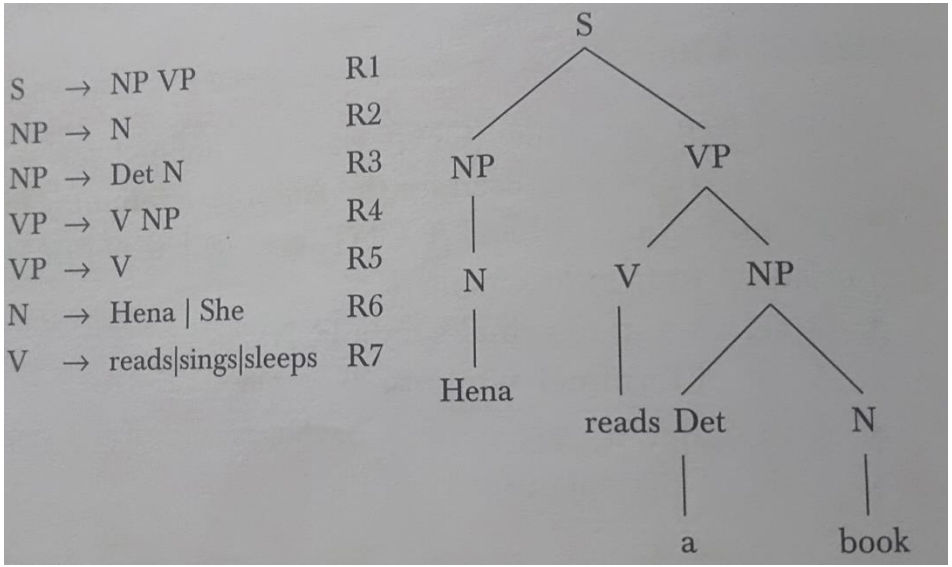
Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

SYNTACTIC ANALYSIS

Overview

- "Syntax" refers to the grammatical arrangement of words in a sentence and the relationship with each other.
- The main objective of syntactic analysis is to find the syntactic structure of the sentence.



- Two important ideas in NLP are:
- **Constituency:** grouping words together.
- **Word Order:** word ordering

Context Free Grammar

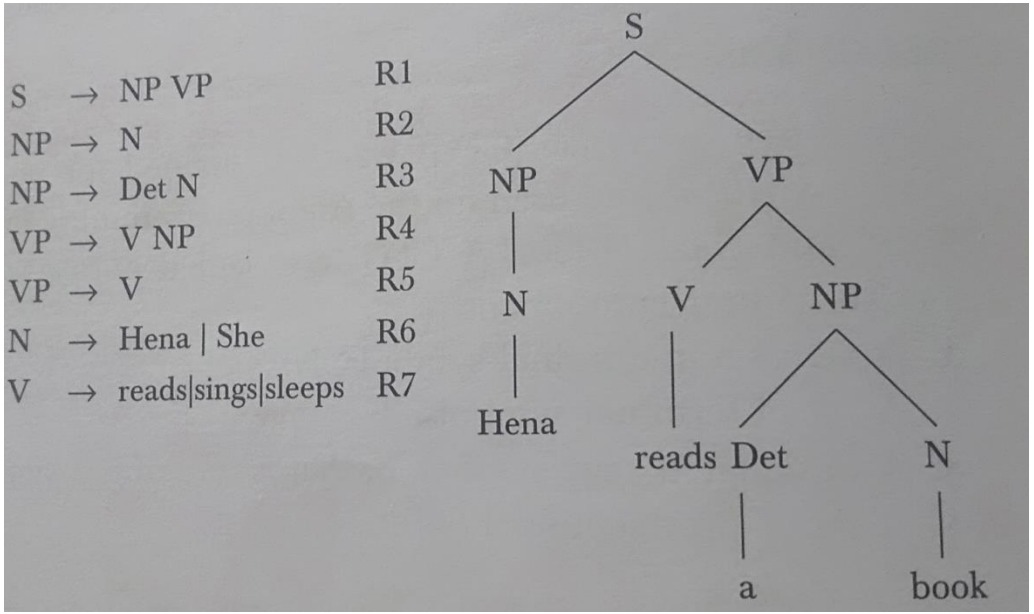
- A CFG consists of four components:

1. A set of non-terminal symbols, N.
2. A set of terminal symbols, T.
3. A designated start symbol, S, that is one of the symbols from N.
4. A set of productions, P, of the form:

$A \rightarrow \alpha$

Example

- Hena reads a book.
- $[_S[_{NP}[_N$ Hena$]][_{VP}[_V$ reads$][_{NP}[_{Det}$ a$][_N$ book$]]]]$



CONSTITUENCY

- Words in a sentence are not tied together as a sequence of part-of-speech.
- Language puts constraints on word order.
- Phrase Level Constructions:

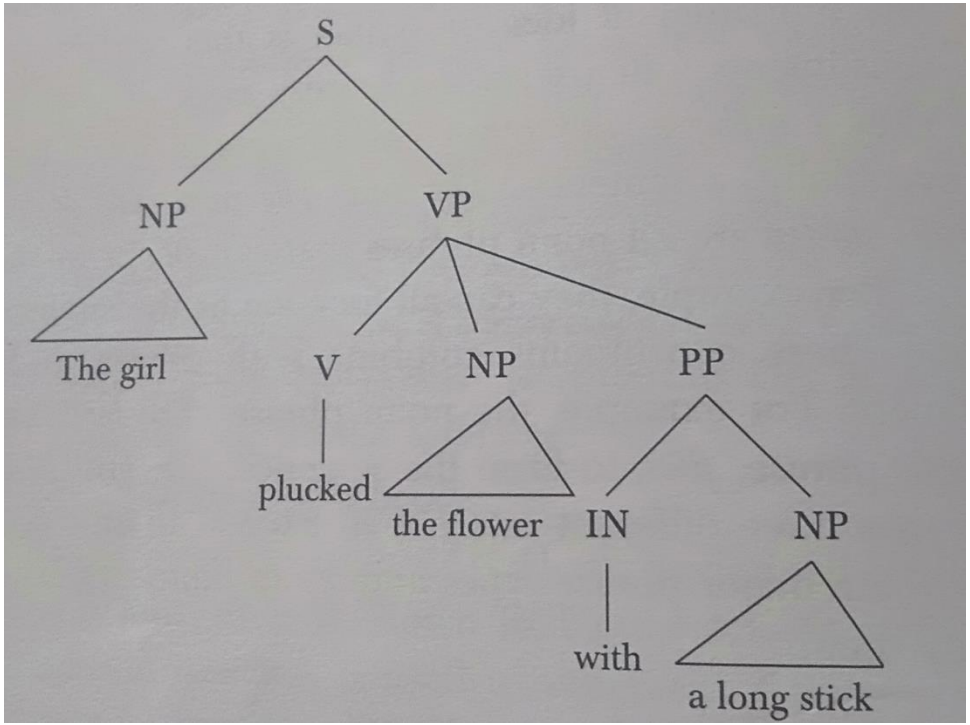
Hena reads a book.

Hena reads a storybook.

Those girls read a book.

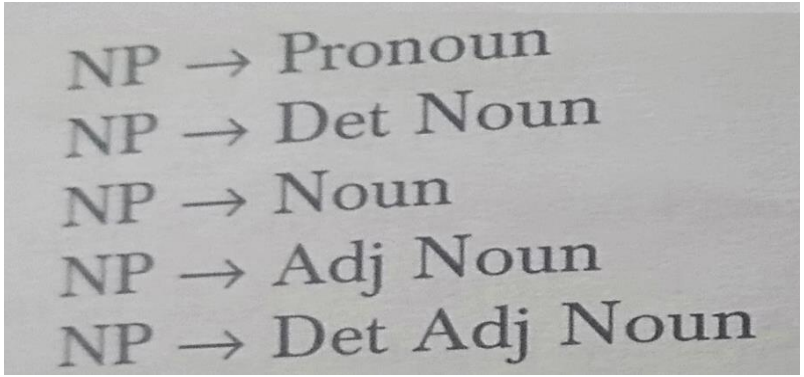
She reads a comic book.

Parse Tree


S
NP
VP
The girl
V
NP
PP
plucked
the flower
IN
NP
with
a long stick

Noun Phrase

- A noun phrase is a phrase whose head is a noun or pronoun, optional accompanied by a set of modifiers.



Examples of Noun Phrase

- NP→(Det) (AP) Noun (PP)

They

The foggy morning

Chilled water

A beautiful lake in Kashmir

Cold banana milk shake

- NP→(Det) (AP) Nom (PP)
- Nom → Noun | Noun Nom

Examples

- A noun phrase can act as a subject, an object or a predicate. The following sentences demonstrate each of these uses.

The foggy damped weather disturbed the match.

I would like a nice cold apple shake.

Kula botanical garden is a beautiful location.

Verb Phrase

- The verb phrase is headed by verb.
- Examples:

Khushbu slept.	[VP → Verb]
The boy kicked the ball.	[VP → Verb NP]
Khushbu slept in the garden.	[VP → Verb PP]
The boy gave the girl a book.	[VP → Verb NP NP]
The boy gave the girl a book with blue cover.	[VP → Verb NP NP PP]

VP → Verb (NP) (NP) (PP)*

Prepositional Phrase

- Prepositional phrases are headed by a prepostion. They consists of a preposition possibly followed by some other constituent, usually a noun phrase.

"We played volleyball on the beach"

- We can have preposition phrase that consists of just a preposition.

"John went outside"

- **PP → Prep (NP)**

Adjective Phrase

- The head of adjective phrase is an adjective. Aps consists of an adjective, which may be preceded by an adverb and followed by a PP.
- Examples:

"Ashish is clever"

"The train is very late."

"My sister is fond of animals."

- **AP → (Adv) Adj (PP)**

Adverb Phrase

- An adverb phrase consists of an adverb, possibly preceded by a degree adverb.
- Example:

"Time passes very quickly"

- AdvP → (Intens) Adv

Sentence Level Constructions

- A sentence can have varying structure. There are four known structures:

1. Declarative Structure.
2. Imperative Structure.
3. yes-no question structure.
4. wh-question structure.

Declarative structure

- Sentences with a declarative structure have a subject followed by a predicate.
- The subject of a declarative sentence is a noun phrase and the predicate is a verb phrase.
- Example: “I like horse riding”
- S → NP VP

Imperative Structure

- Sentences with an imperative structure usually begin with a verb phrase and lack subject. The subject of these types of sentence is implicit and is understood to be 'you'.
- Rule: S → VP
- Examples:

Look at the door.

Give me the book.

Stop talking.

Show me the latest design.

Yes-no question structure

- Sentences with the yes-no question structure ask questions which can be answered using yes or no. These sentences begin with auxiliary verb, followed by a subject NP, followed by a VP.
- Rule: "S → Aux NP VP"
- Examples:

Do you have a red pen?

Is there a vacant quarter?

Is the game over?

Can you show me your album?

Wh-question structure

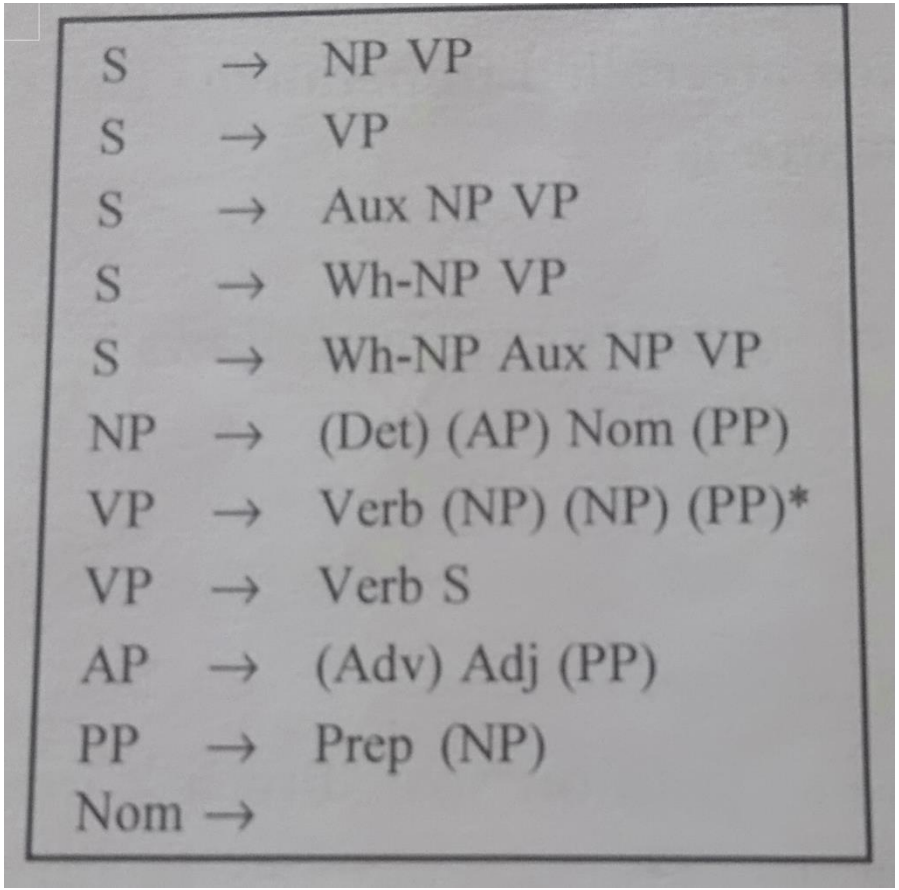
- Sentences with wh-question structure are more complex.
- These sentences begin with a wh-words who, which, where, what, why and how.
- A wh-question may have a wh-phrase as a subject or may include another subject.
- Rule: **S → Wh-NP VP**

Which team won the match?

- Rule: **S → Wh-NP Aux NP VP**

Which cameras can you show me in your shop?

Grammar Rules



S	→	NP VP
S	→	VP
S	→	Aux NP VP
S	→	Wh-NP VP
S	→	Wh-NP Aux NP VP
NP	→	(Det) (AP) Nom (PP)
VP	→	Verb (NP) (NP) (PP)*
VP	→	Verb S
AP	→	(Adv) Adj (PP)
PP	→	Prep (NP)
Nom	→	

Agreement

$$
\begin{aligned}
3sgNP &\rightarrow (Det)\ (AP)\ SgNom\ (PP)\\
Non3sgNP &\rightarrow (Det)\ (AP)\ PlNom\ (PP)\\
SgNom &\rightarrow SgNoun \mid SgNoun\ SgNom\\
PlNom &\rightarrow PlNoun \mid PlNoun\ PlNom\\
SgNoun &\rightarrow Priya \mid lake \mid banana \mid sister \mid \ldots\\
PlNoun &\rightarrow Children \mid \ldots
\end{aligned}
$$

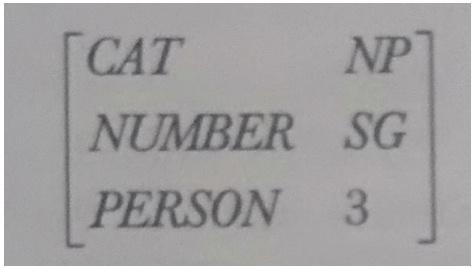
Feature Structures

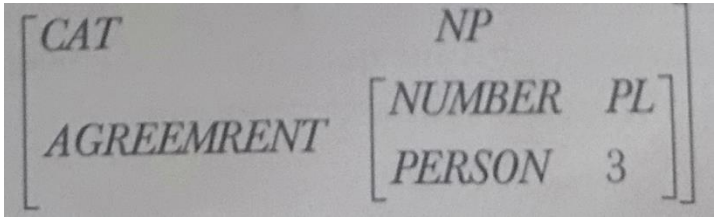
- They are sets of feature-value pairs.
- They can be used to efficiently capture the properties of grammatical categories.

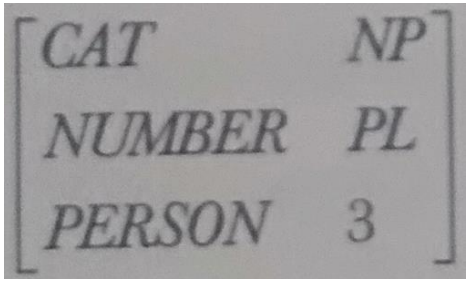
$FEATURE_1 \quad VALUE_1$

- $FEATURE_2 \quad VALUE_2$

$FEATURE_N \quad VALUE_N$







THANK YOU

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

PARSING

Overview

- The task that uses the rewrite rules of a grammar to either generate a particular sequence of words or reconstruct its derivation is termed as **parsing**.
- The following constraints guide the search process:

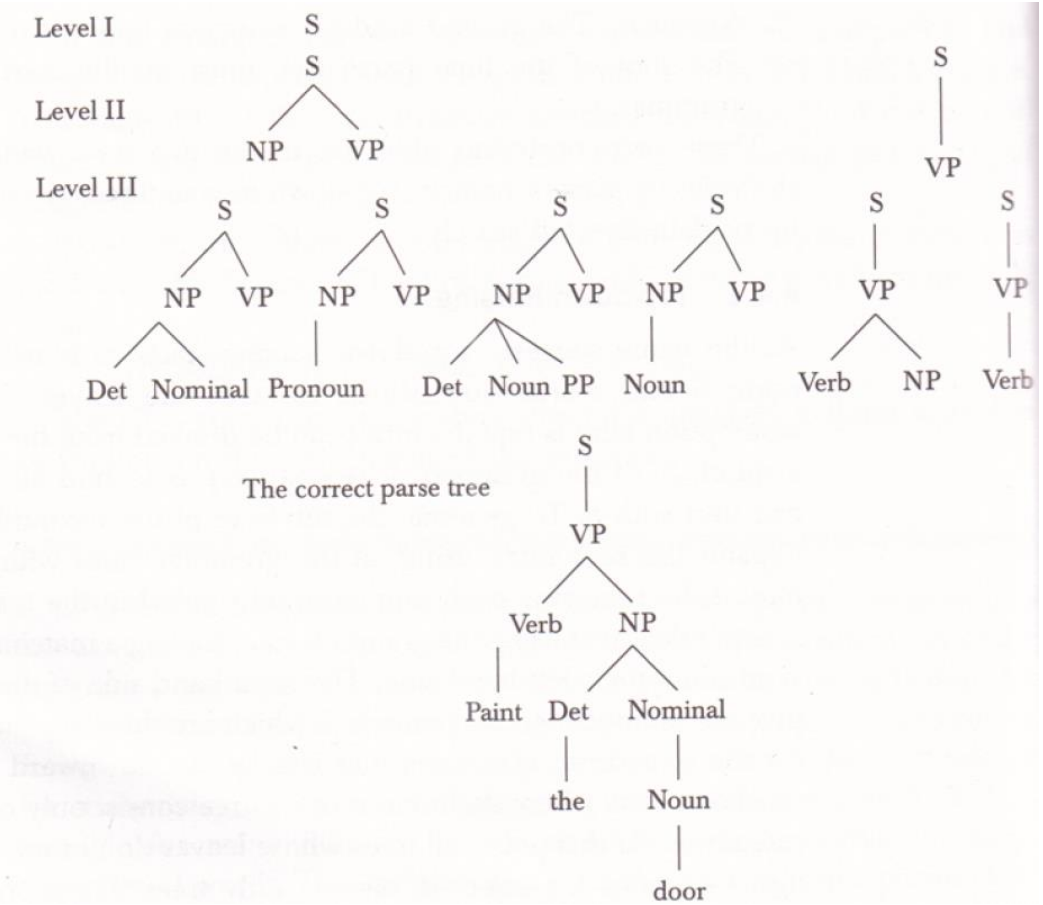
1. Input: Words in the input sentence. A valid parse is one that covers all the words in a sentence. These words must constitute the leaves of the final parse tree.

2. Grammar: The root of the final parse tree must be the start symbol of the grammar.

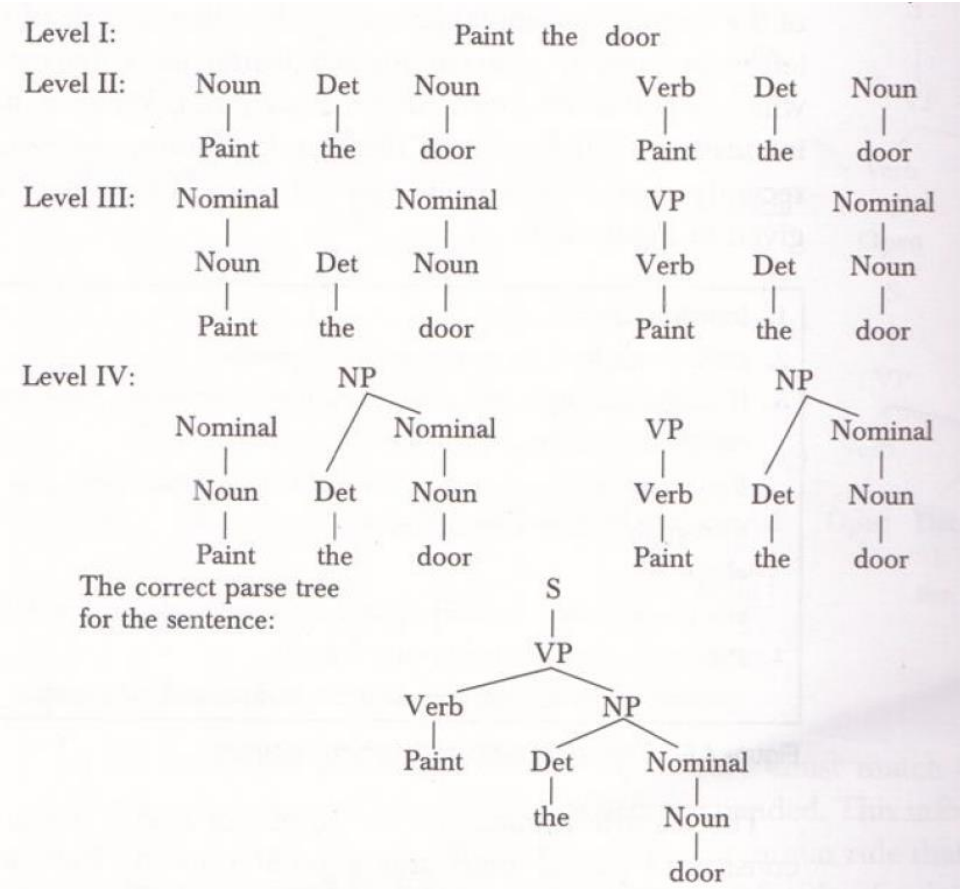
Top-down parsing

S → NP VP	VP → Verb NP
S → VP	VP → Verb
NP → Det Nominal	PP → Preposition NP
NP → Noun	Det → this | that | a | the
NP → Det Noun PP	Verb → sleeps | sings | open | saw | paint
Nominal → Noun	Preposition → from | with | on | to
Nominal → Noun Nominal	Pronoun → she | he | they

Top-down search space



Bottom-up parsing



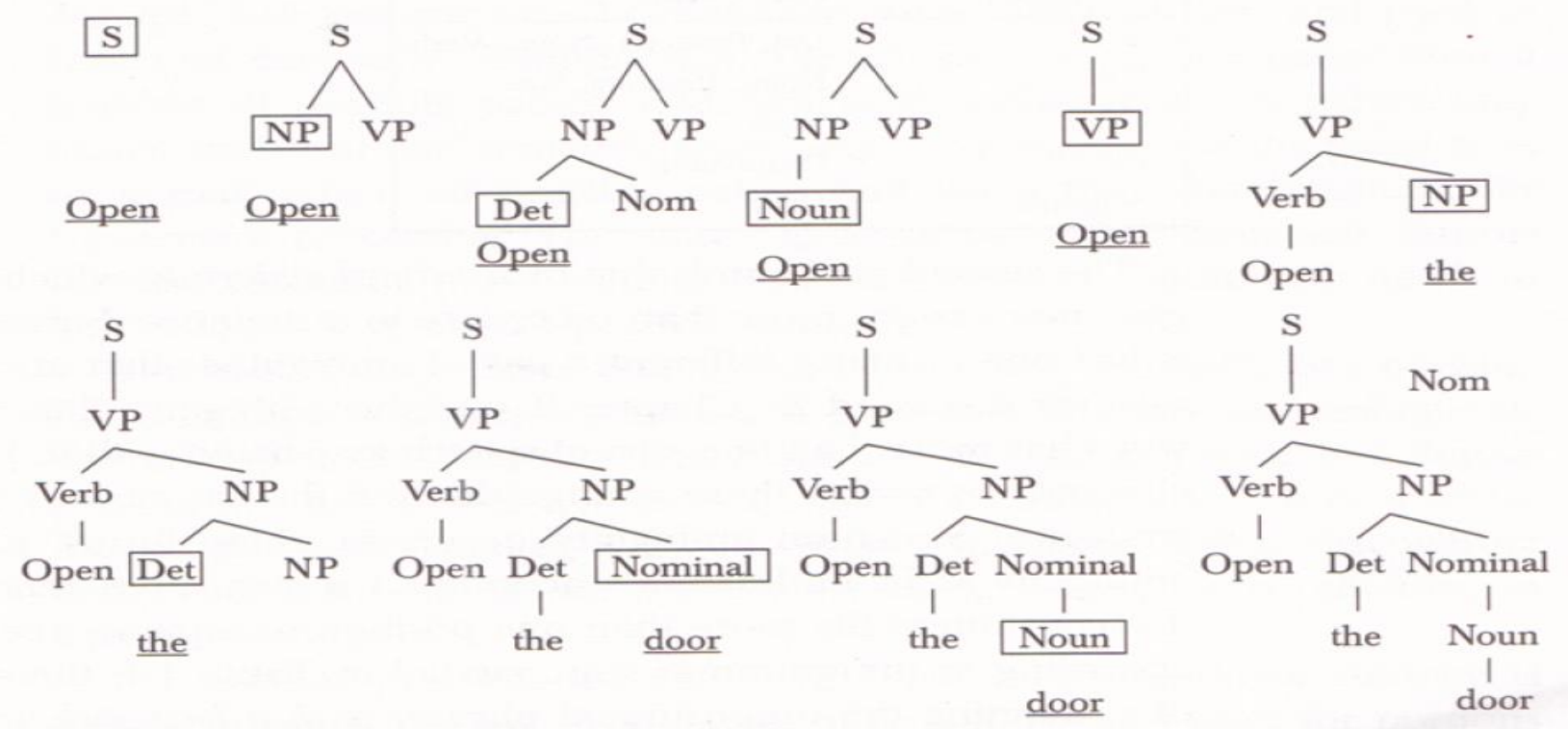
Pros and Cons

- The top-down parser search starts generating trees with the start symbol of the grammar. It never wastes time exploring a tree leading to a different root.
- It wastes time exploring S trees that eventually result in words that are inconsistent with the input.
- The bottom up parser never explores a tree that does not match the input.
- It wastes time generating trees that have no chance of leading to an S-rooted tree.

Basic Top-Down Parser

1. Initialize agenda
2. Pick a state, let it be curr_state, from agenda
3. If (curr_state) represents a successful parse then return parse tree
   else if curr_stat is a POS then
   if category of curr_state is a subset of POS associated with curr_word
   then apply lexical rules to current state
   else reject
   else generate new states by applying grammar rules and push them into agenda
4. If (agenda is empty) then return failure
   else select a node from agenda for expansion and go to step 3.

Derivation using top-down, depth first algorithm



Disadvantages

1. **Left Recursion:** which causes the search to get stuck in an infinite loop.
2. **Structural Ambiguity:** which occurs when a grammar assigns more than one parse to a sentence.
3. **Attachment Ambiguity:** If a constituent fits more than one position in a parse tree.
4. **Coordination Ambiguity:** Occurs when its is nit clear which phrases are being combined with a conjunction like 'and'.
5. **Local Ambiguity:** Occurs when certain parts of a sentence are ambiguous.
6. **Repeated Parsing:** Parser often builds valid trees for portions of the input that it discards during backtracking.

EARLEY PARSER

- It implements an efficient parallel top-down search using dynamic programming.
- It builds a table of sub-trees for each of the constituents in the input.
- The most important component of this algorithm is the Earley Chart that has n+1 entries, where n is the number of words in the input.
- The algorithm makes a left to right scan of input to fill the elements in this chart.

State Information

1. A sub-tree corresponding to a grammar rule.
2. Information about the program made in completing the sub-tree.
3. Position of the sub-tree with respect to input.

A state is represented as a dotted rule and a pair of numbers representing starting position and the position of dot.

$$A \rightarrow X_1 \ldots \bullet C \ldots X_m, [i,j]$$

Algorithm

Earley Parsing
Input: Sentence and the Grammar
Output: Chart

```
chart[0] ← S′ → S, [0,0]
n ← length (sentence) // number of words in the sentence
for i = 0 to n do
    for each state in chart[i] do
    if (incomplete (state) and next category is not a part of speech) then
    predictor (state)
    else if (incomplete (state) and next category is a part of speech)
            scanner (state)
        else
            completer (state)
        end-if
    end-if
end for
end for
return
```

Procedure predictor $(A \rightarrow X_1 \ldots \bullet B \ldots X_m, [i, j])$
for each rule $(B \rightarrow \alpha)$ in G do
insert the state $B \rightarrow \bullet\ \alpha, [j, j]$ to chart $[j]$
End
Procedure scanner $(A \rightarrow X_1 \ldots \bullet B \ldots X_m, [i, j])$
If B is one of the part of speech associated with word$[j]$ then
Insert the state $B \rightarrow$ word $[j] \bullet, [j, j+1]$ to chart $[j+1]$
End
Procedure Completer $(A \rightarrow X_1 \ldots \bullet, [j, k])$
for each $B \rightarrow X1 \ldots \bullet A \ldots, [i, j]$ in chart$[j]$ do
insert the state $B \rightarrow X1 \ldots A\bullet \ldots [i, k]$ to chart$[k]$
End

Predictor

- Generates new states representing potential expansion of the non-terminal in the left-most derivation.
- It is applied to every state that has a non-terminal to the right of the dot, when the category of that non-terminal is different from the part-of-speech.
- If $A \rightarrow X_1 \ldots \bullet C \ldots X_m, [i,j]$ then for every rule of the form $B \rightarrow \alpha$ the operation adds to chart[j], the state:

$$B \rightarrow \bullet \alpha, [j,j]$$

Example

- When the generating state is S → • NP VP, [0,0] the predictor adds the following states to chart[0]:
- NP → • Det Nominal, [0,0]
- NP → • Noun, [0,0]
- NP → • Pronoun, [0,0]
- NP → • Det Noun PP, [0,0]

Scanner

- Scanner is used when a state has a part of speech category to the right of the dot.
- It examines the input to see if the part-of-speech appearing to the right of the dot matches one of the part-of-speech associated with the current input.
- IfYes, then it creates a new state using the rule.
- If the state is A → … • a …, [i,j] and 'a' is associated with w_j then, it adds a → … w_j • [i,j] to chart [j+1].

Completer

- Completer is used when the dot reaches the right end of the rule.
- The presence of such a state signifies successful completion of the parse of some grammatical category.
- If A → … •, [j,k], then the computer adds

B → … A • … [i, k] to chart [k] for all states

B → … A • … , [i, j] in chart [j].

Chart [0] Dummy	S0	S′ → • S start	[0,0] state
	S1	S → • NP VP	
	S2	S → • VP	
	S3	NP → • Det Nominal	
	S4	NP → • Noun	
	S5	NP → • Pronoun	
	S6	NP → Det Noun PP	
	S7	VP → • Verb NP	
	S8	VP → • Verb	
Chart [1]	S9	Noun → paint •	[0,1]
	S10	Verb → paint •	[0,1]
	S11	NP → Noun •	[0,1]
	S12	VP → Verb• NP	[0,1]
	S13	VP → Verb •	[0,1]
	S14	S → NP • VP	[0,1]
	S15	NP → • Det Nominal	[1,1]
	S16	NP → • Noun	[1,1]
	S17	S → VP •	[0,1]
	S18	VP → • Verb NP	[1,1]
	S19	VP → • Verb	[1,1]
Chart [2]	S20	Det → the •	[1,2]
	S21	NP → Det • Nominal	[1,2]
	S22	Nominal → • Noun	[2,2]
	S23	Nominal → • Noun Nominal	[2,2]
Chart [3]	S24	Noun → door •	[2,3]
	S25	Nominal → Noun •	[2,3]
	S26	NP → Det Nominal•	[1,3]
	S27	S → NP • VP	[0,3]
	S28	VP → Verb NP •	[0,3]
	S29	VP → • Verb NP	[3,3]
	S30	VP → • Verb	[3,3]
	S31	S → VP •	[0,3]

CYK Parser

- Cocke-Younger-Kasami is a dynamic programming parsing algorithm.
- It follows a bottom-up approach in parsing. Builds the parse tree incrementally. Each entry in the table is based on the previous entries.
- The CYK algorithm assumes the grammar to be in chomsky normal form (CNF). A CFG is in CNF if all the rules are of only two forms.

$$A \rightarrow BC$$

$$A \rightarrow W, \text{ where } W \text{ is a word.}$$

CYK Algorithm

```
Let w = w1 w2 w3 wi ...wj...wn
    and wij = wi ...wi+j-1
// Initialization step
for i := 1 to n do
    for all rules A→ wi do
        chart [i,1] = {A}
// Recursive step
for j = 2 to n do
    for i = 1 to n-j+1 do
    begin
        chart [i, j] = φ
        for k = 1 to j -1 do
        chart [i, j] := chart[i, j] ∪{A | A → BC is a production and
                        B ∈ chart[i, k] and  C ∈ chart [i+k, j-k]}
    end
if S ∈ chart[1, n] then accept else reject
```

Example

Sentence: "The girl wrote an essay"

	1	2	3	4	5
1	Det → The	NP → Det Noun			S → NP VP
2	Noun → Girl				
3	Verb → wrote		VP → Verb NP		
4	Det → an	NP → Det Noun			
5	Noun → essay				

Probabilistic Parsing

- A statistical parser works by assigning probabilities to possible parses of a sentence and returning the most likely parse as the final one.
- More formally given a grammar G, sentence s and a set of possible parse trees of s which we denote by $\tau(s)$, a probabilistic parser finds the most likely parse $\varphi`$ of s as follows:

$$\begin{aligned}\varphi' &= \operatorname{argmax}_{\varphi \in \tau(s)} P(\varphi \mid s) \\ &= \operatorname{argmax}_{\varphi \in \tau(s)} P(\varphi, s) \\ &= \operatorname{argmax}_{\varphi \in \tau(s)} P(\varphi)\end{aligned}$$

Advantages

1. Probabilistic parser offers is removal of ambiguity for parsing.
2. The search becomes more efficient.

Probabilistic Context Free Grammar

Let us now define PCFG. A PCFG is defined by the pair (G, f), where G is a CFG and f is a positive function defined over the set of rules such that, the sum of the probabilities associated with the rules expanding a particular non-terminal is 1 (Infante-Lopez and Maarten de Rijke 2006).

$$\sum_{\alpha} f(A \rightarrow \alpha) = 1$$

Example

S → NP VP	0.8	Noun → door	0.25
S → VP	0.2	Noun → bird	0.25
NP→ Det Noun	0.4	Noun → hole	0.25
NP → Noun	0.2	Verb → sleeps	0.2
NP → Pronoun	0.2	Verb → sings	0.2
NP → Det Noun PP	0.2	Verb → open	0.2
VP → Verb NP	0.5	Verb → saw	0.2
VP → Verb	0.3	Verb → paint	0.2
VP → VP PP	0.2	Preposition → from	0.3
PP → Preposition NP	1.0	Preposition → with	0.25
Det → this	0.2	Preposition → on	0.2
Det → that	0.2	Preposition → to	0.25
Det → a	0.25	Pronoun→ she	0.35
Det → the	0.35	Pronoun→ he	0.35
Noun → paint	0.25	Pronoun→ they	0.25

Probability Estimation

$$f(S \rightarrow NP\ VP) + f(S \rightarrow VP) = 1$$

$$f(NP \rightarrow Det\ Noun) + f(NP \rightarrow Noun) + f(NP \rightarrow Pronoun) + f(NP \rightarrow Det\ Noun\ PP) = 1$$

$$f(VP \rightarrow Verb\ NP) + f(NP \rightarrow Verb) + f(VP \rightarrow VP\ PP) = 1.0$$

$$f(Det \rightarrow this) + f(Det \rightarrow that) + f(Det \rightarrow a) + f(Det \rightarrow the) = 1.0$$

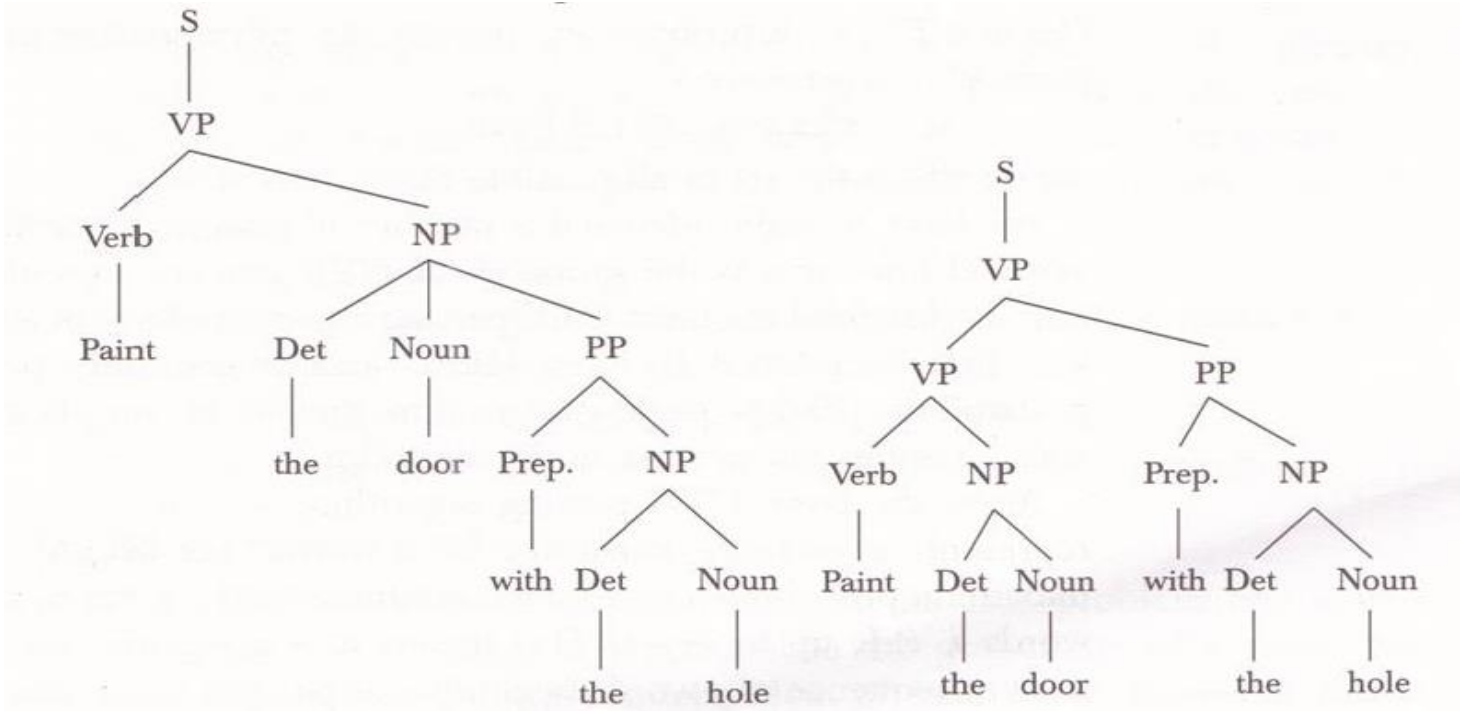
$$f(Noun \rightarrow paint) + f(Noun \rightarrow door) + f(Noun \rightarrow bird) + f(Noun \rightarrow hole) = 1.0$$

Estimating Rule Probability

$$P_{MLE}(A \rightarrow \alpha) = \frac{\text{Count}(A \rightarrow \alpha)}{\sum_{\alpha} \text{Count}(A \rightarrow \alpha)}$$

Rule	Count (A → α)	Count A	MLE estimates
S → VP	2	2	1
NP → Det Noun PP	1	4	0.25
NP → Det Noun	3	4	0.75
VP → Verb NP	2	3	0.66
VP → VP PP	1	3	0.33
Det → the	2	2	1
Noun → hole	2	4	0.5
Noun → door	2	4	0.5
Prep → with	1	1	1
Verb → Paint	1	1	1

Two Parse Trees



Parsing PCFGs

Initialization:

for $i := 1$ to n do

for all rules $A \rightarrow w_i$ do

$\varphi[i,1,A] = P(A \rightarrow w_i)$

Recursive Step:

for $j = 2$ to n do

for $i = 1$ to $n-j+1$ do

begin

$\varphi[i,1,A] = \phi$

for $k = 1$ to $j-1$ do

$$\varphi'[i,j,A] = \max_k \varphi'[i,k,B] \times \varphi'[k,j,C] \times P(A \rightarrow BC),$$

such that $A \rightarrow BC$ is a production rule in grammar

$BP[i,j,A] = \{ k, A, B \}$

end

Indian Languages

The majority of the indian languages are free word order. The order of the sentence can be changed without leading to a grammatically incorrect sentence.

सबा खाना खाती है ।	Saba khana khati hai.
खाना सबा खाती है ।	Khana Saba khati hai.

Contd..

- Extensive and productive use of complex predicates (CPs) is another property that most Indian languages have in common.
- A complex predicate combines a light verb, noun, or adjective to produce a new verb.

(a) सबा आयी ।
(*Saba Ayi.*)
Saba came.

(b) सबा आ गयी ।
(*Saba a gayi.*)
Saba come went.
Saba arrived.

(c) सबा आ पडी ।
Saba a pari.
Saba come fell.
Saba came (suddenly).

Parsing Indian Languages

- Bharti and Sangal described an approach for parsing of Indian languages based on Paninian grammar formalism. It has 2 stages:

1. Is responsible for identifying word groups.
2. Assigning parse structure to the input sentence.

लडकियाँ मैदान में हाकी खेल रही हैं। (4.8)

Ladkiyan maidaan mein hockey khel rahi hein.

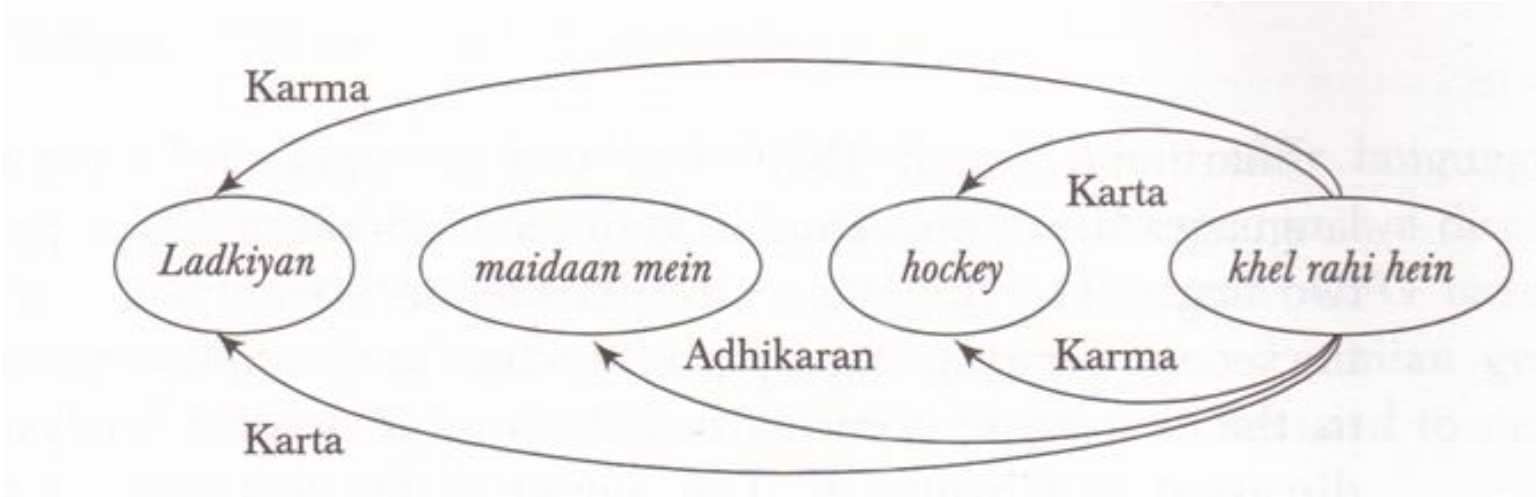
The word *ladkiyan* forms one unit, the words *maidaan* and *mein* are grouped together to form a noun group, and the word sequence *khel rahi hein* forms a verb group.

Karaka Chart

Table 4.6 Default Karaka chart

Karaka (case relations)	Vibhakti (case markers or post-positions)	Necessity
Karta	ϕ	Mandatory
Karma	*Ko* or ϕ	Mandatory
Adhikaran	*Mein* or *par*	Optional
Sampradan	*Ko* or *ke liye*	Optional

Constraint Graph

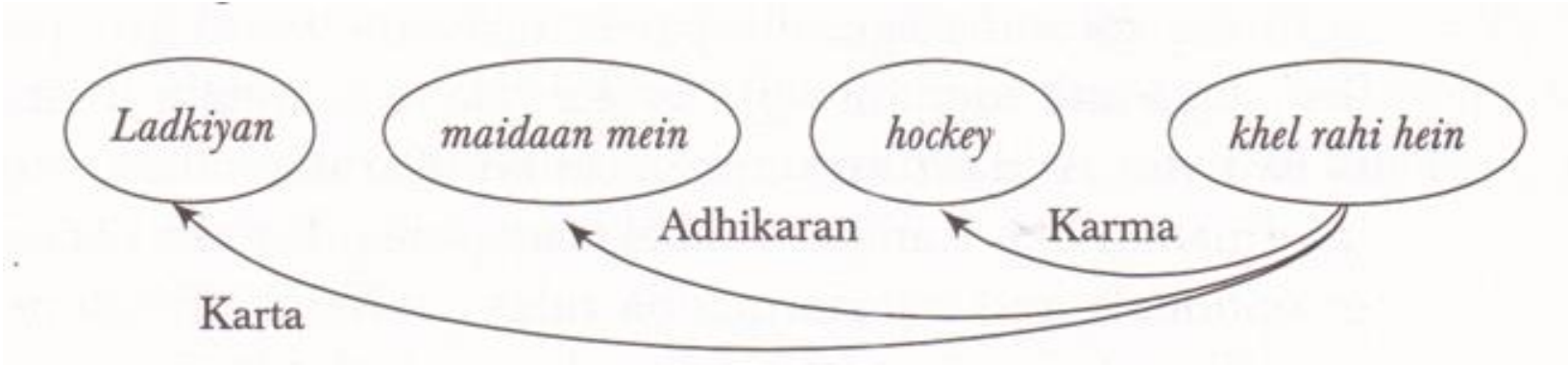


Constraints

Each sub-graph of the constraint graph that satisfies the following constraints yields a parse of the sentence.

1. It contains all the nodes of the graph.
2. It contains exactly one outgoing edge from a verb group for each of its mandatory Karakas. These edges are labelled by the corresponding Karaka.
3. For each of the optional Karaka in Karaka chart, the sub-graph can have at most one outgoing edge labelled by the Karaka from the verb group.
4. For each noun group, the sub-graph should have exactly one incoming edge.

Parse of the sentence



References

1. Bharti, Akshar and Rajeev Sangal, 1990, 'A Karaka-based approach parsing of Indian Languages', Proceedings of the 13th Conference on Computational Linguistics, Association for Computational Linguistics, 3.
2. Chomsky, N., 1957, Syntactic Structures, Mouton, The Hague.

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Subsequence Kernels for Relation Extraction

Overview

- Extracting semantic relationships between entities mentioned in text documents is an important task in natural language processing.
- The various types of relationships that are discovered between mentions of entities can provide useful structured information to a text mining system.
- The task specifies a predefined set of entity types and relation types that are deemed to be relevant to a potential user and that are likely to occur in a particular text collection.
- Example, information extraction from newspaper articles is usually concerned with identifying mentions of people, organizations, locations, and extracting useful relations between them.

Contd..

Two recent approaches to relation extraction that Differ in terms of the kind of linguistic information they use:

1. Each potential relation is represented implicitly as a vector of features, where each feature corresponds to a *word sequence* anchored at the two entities forming the relationship. A relation extraction system is trained based on the subsequence kernel. This kernel is further generalized so that words can be replaced with word classes, thus enabling the use of information coming from POS tagging, named entity recognition, chunking,

or Wordnet.

2. The representation is centered on the shortest *dependency path* between the two entities in the dependency graph of the sentence. Because syntactic analysis is essential in this method, its applicability is limited to domains where syntactic parsing gives reasonable accuracy.

Subsequence Kernels for Relation Extraction

- One of the first approaches to extracting interactions between proteins from biomedical abstracts is that of Blaschke *et al.*, described in [7, 8].
- Their system is based on a set of manually developed rules, where each rule (or frame) is a sequence of words (or POS tags) and two protein-name tokens. Between every two adjacent words is a number indicating the maximum number of intervening words allowed when matching the rule to a sentence.
- An example rule is "*interaction of (3) <P> (3) with (3) <P>*", where '<P>' is used to denote a protein name. A sentence matches the rule if and only if it satisfies the word constraints in the given order and respects the respective word gaps.
- ELCS (Extraction using Longest Common Subsequences) rule representation does not use POS tags, but allows disjunctions of words.
- An example rule learned by this system is "- *(7) interaction (0) [between | of] (5) <P> (9) <P> (17) .*"

Capturing Relation Patterns with a String Kernel

- Both Blaschke and ELCS do relation extraction based on a limited set of matching rules, where a rule is simply a sparse (gappy) subsequence of words or POS tags anchored on the two protein-name tokens. Therefore, the two methods share a common limitation:
- either through manual selection (Blaschke),
- or as a result of a greedy learning procedure (ELCS),
- they end up using only a subset of all possible anchored sparse subsequences.

Contd..

- The feature space is further prunned down by utilizing the following property of natural language statements: when a sentence asserts a relationship between two entity mentions, it generally does this using one of the following four patterns:
- **[FB] F**ore–**B**etween: words before and between the two entity mentions are simultaneously used to express the relationship. Examples: 'interaction of *P*1 with *P*2,' 'activation of *P*1 by *P*2.'
- **[B] B**etween: only words between the two entities are essential for asserting the relationship. Examples: '*P*1 interacts with *P*2,' '*P*1 is activated by *P*2.'
- **[BA] B**etween–**A**fter: words between and after the two entity mentions are simultaneously used to express the relationship. Examples: '*P*1 – *P*2 complex,' '*P*1 and *P*2 interact.'
- **[M]M**odifier: the two entity mentions have no words between them. Examples: *U.S. troops* (a Role:Staff relation), *Serbian general* (Role:Citizen).

A Generalized Subsequence Kernel

- Let $\Sigma 1, \Sigma 2, \ldots, \Sigma k$ be some disjoint feature spaces.
- $\Sigma 1$ could be the set of words, $\Sigma 2$ the set of POS tags, etc.
- Let $\Sigma \times = \Sigma 1 \times \Sigma 2 \times \ldots \times \Sigma k$ be the set of all possible feature vectors, where a feature vector would be associated with each position in a sentence.
- Given two feature vectors $x, y \in \Sigma \times$, let $c(x, y)$ denote the number of common features between x and
- y.

- Let s, t be two sequences over the finite set $\Sigma\times$, and let $|s|$ denote the length of $s = s1 \ldots s|s|$.
- The sequence $s[i:j]$ is the contiguous subsequence $si \ldots sj$ of s. Let $\mathbf{i} = (i1, \ldots, i|\mathbf{i}|)$ be a sequence of $|\mathbf{i}|$ indices in s, in ascending order.
- We define the length $l(\mathbf{i})$ *of the index sequence* $\mathbf{i}$ *in* s as $i|\mathbf{i}| - i1 + 1$. Similarly, $\mathbf{j}$ is a sequence of $|\mathbf{j}|$ indices in t.

- Let $\Sigma U = \Sigma 1 \cup \Sigma 2 \cup \ldots \cup \Sigma k$ be the set of all possible features. We say that the sequence $u \in \Sigma * U$ is a (sparse) subsequence of s if there is a sequence of $|u|$ indices $\mathbf{i}$ such that $uk \in sik$, for all $k = 1, \ldots, |u|$.
- Equivalently, we write $u \prec s[\mathbf{i}]$ as a shorthand for the component-wise '$\in$' relationship between u and $s[\mathbf{i}]$.

Contd..

- Finally, let $Kn(s, t, \lambda)$ (Equation 3.1) be the number of weighted sparse subsequences u of length n common to s and t (i.e., $u \prec s[\mathbf{i}]$, $u \prec t[\mathbf{j}]$), where the weight of u is $\lambda l(\mathbf{i})+l(\mathbf{j})$, for some $\lambda \leq 1$.

$$K_n(s,t,\lambda) = \sum_{u \in \Sigma^n} \sum_{\mathbf{i}: u \prec s[\mathbf{i}]} \sum_{\mathbf{j}: u \prec t[\mathbf{j}]} \lambda^{l(\mathbf{i})+l(\mathbf{j})}$$

Contd..

$$K_n(s,t,\lambda) = \sum_{\mathbf{i}:|\mathbf{i}|=n} \sum_{\mathbf{j}:|\mathbf{j}|=n} \prod_{k=1}^{n} c(s_{i_k}, t_{j_k}) \lambda^{l(\mathbf{i})+l(\mathbf{j})} \qquad (3.2)$$

To enable an efficient computation of *Kn*, we use the auxiliary function *K n* with a definition similar to *Kn*, the only difference being that it counts the length from the beginning of the particular subsequence *u* to the end of the strings *s* and *t*, as illustrated in Equation 3.3:

$$K_n'(s,t,\lambda) = \sum_{u \in \Sigma^n_{\cup}} \sum_{\mathbf{i}:u \prec s[\mathbf{i}]} \sum_{\mathbf{j}:u \prec t[\mathbf{j}]} \lambda^{|s|+|t|-i_1-j_1+2} \qquad (3.3)$$

Computation of subsequence kernel

$$
\begin{aligned}
K_0^{'}(s,t) &= 1, \ for\ all\ s,t \\
K_i^{'}(s,t) &= 0, \ if\ \min(|s|,|t|) < i \\
K_i^{''}(s,\emptyset) &= 0, \ for\ all\ i,s \\
K_i^{''}(sx,ty) &= \lambda K_i^{''}(sx,t) + \lambda^2 K_{i-1}^{'}(s,t) \cdot c(x,y) \\
K_i^{'}(sx,t) &= \lambda K_i^{'}(s,t) + K_i^{''}(sx,t) \\
K_n(s,t) &= 0, \ if\ \min(|s|,|t|) < n \\
K_n(sx,t) &= K_n(s,t) + \sum_j \lambda^2 K_{n-1}^{'}(s,t[1:j-1]) \cdot c(x,t[j])
\end{aligned}
$$

Computing the Relation Kernel

- the input consists of a set of sentences, where each sentence contains exactly two entities.

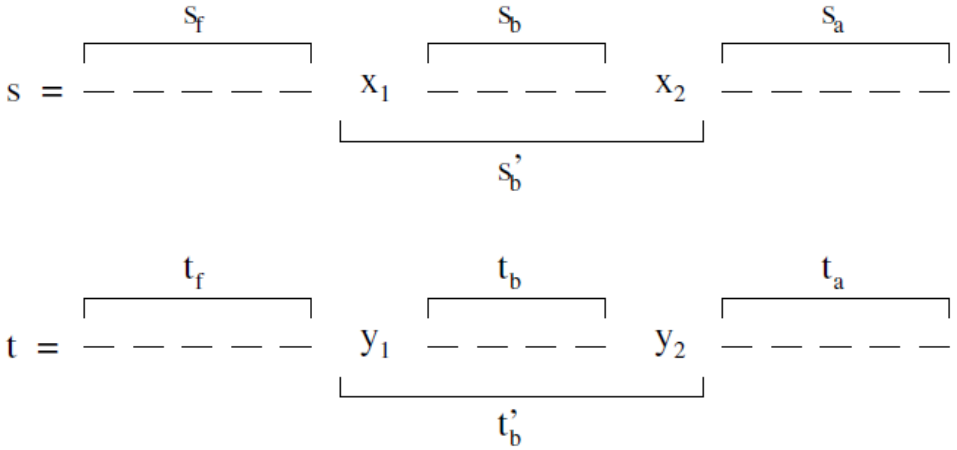


Fig. 3.2. Sentence segments.

Computation of relation kernel.

$$rK(s,t) = fbK(s,t) + bK(s,t) + baK(s,t) + mK(s,t)$$

$$bK_i(s,t) = K_i(s_b, t_b, 1) \cdot c(x_1, y_1) \cdot c(x_2, y_2) \cdot \lambda^{l(s_b^{'}) + l(t_b^{'})}$$

$$fbK(s,t) = \sum_{i,j} bK_i(s,t) \cdot K_j^{'}(s_f, t_f), \quad 1 \leq \mathrm{i},\ 1 \leq \mathrm{j},\ \mathrm{i} + \mathrm{j} < \mathrm{fb}_{\max}$$

$$bK(s,t) = \sum_{i} bK_i(s,t), \quad 1 \leq \mathrm{i} \leq \mathrm{b}_{\max}$$

$$baK(s,t) = \sum_{i,j} bK_i(s,t) \cdot K_j^{'}(s_a^{-}, t_a^{-}), \quad 1 \leq \mathrm{i},\ 1 \leq \mathrm{j},\ \mathrm{i} + \mathrm{j} < \mathrm{ba}_{\max}$$

$$mK(s,t) = \mathbb{1}(s_b = \emptyset) \cdot \mathbb{1}(t_b = \emptyset) \cdot c(x_1, y_1) \cdot c(x_2, y_2) \cdot \lambda^{2+2},$$

A Dependency-Path Kernel for Relation Extraction

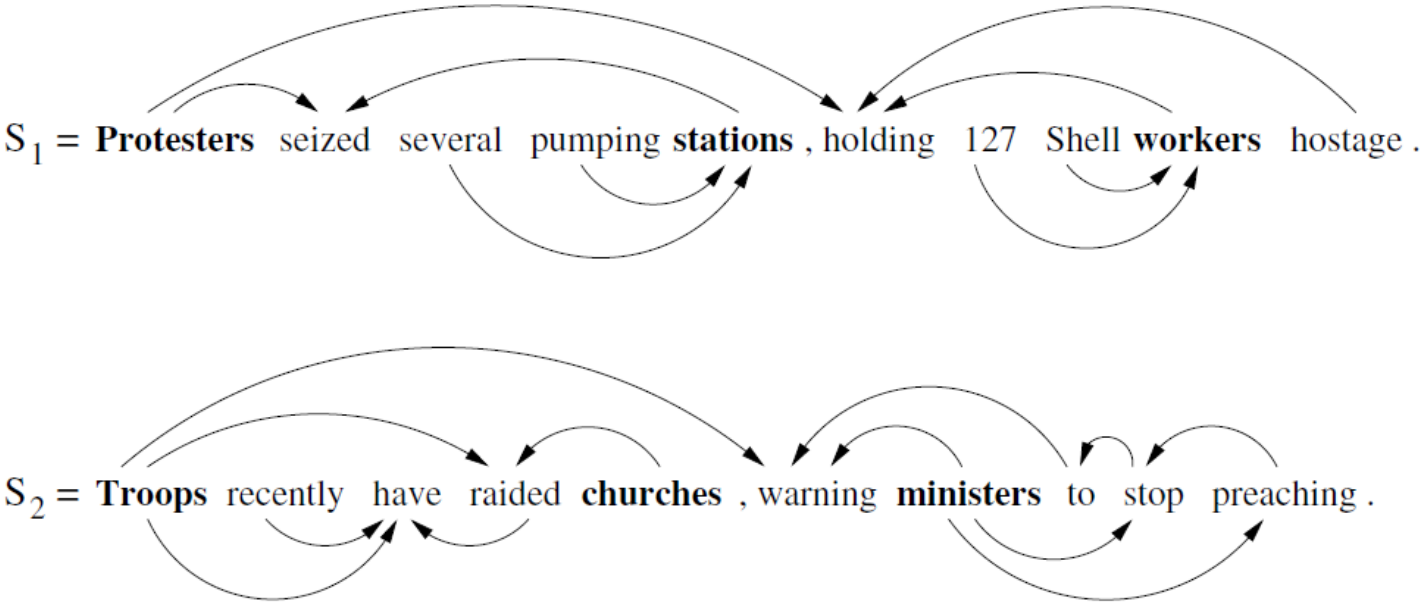


Fig. 3.4. Sentences as dependency graphs.

Contd..

Word-word dependencies are typically categorized in two classes as follows:

- [**Local Dependencies**] These correspond to local predicate-argument (or head-modifier) constructions such as 'troops → raided', or 'pumping → stations' in Figure 3.4.
- [**Non-local Dependencies**] Long-distance dependencies arise due to various linguistic constructions such as coordination, extraction, raising and control. In Figure 3.4, among non-local dependencies are 'troops → warning', or 'ministers → preaching'.

The Shortest Path Hypothesis

- If $e1$ and $e2$ are two entities mentioned in the same sentence such that they are observed to be in a relationship R, then the contribution of the sentence dependency graph to establishing the relationship $R(e1, e2)$ is almost exclusively concentrated in the shortest path between $e1$ and $e2$ in the undirected version of the dependency graph.

Table 3.1. Shortest Path representation of relations.

Relation Instance	Shortest Path in Undirected Dependency Graph
S_1:protesters AT stations	**protesters** → seized ← **stations**
S_1:workers AT stations	**workers** → holding ← protesters → seized ← **stations**
S_2:troops AT churches	**troops** → raided ← **churches**
S_2:ministers AT churches	**ministers** → warning ← troops → raided ← **churches**

Relation Examples

(1) He had no regrets for **his** actions in **Brcko**.

his → actions ← in ← **Brcko**

(2) U.S. **troops** today acted for the first time to capture an alleged Bosnian war criminal, rushing from unmarked vans parked in the northern Serb-dominated **city** of Bijeljina.

troops → rushing ← from ← vans → parked ← in ← **city**

(3) Jelisic created an atmosphere of terror at the **camp** by killing, abusing and threatening the **detainees**.

detainees → killing ← Jelisic → created ← at ← **camp**
detainees → abusing ← Jelisic → created ← at ← **camp**
detainees → threatning ← Jelisic → created ← at ← **camp**
detainees → killing → by → created ← at ← **camp**
detainees → abusing → by → created ← at ← **camp**
detainees → threatening → by → created ← at ← **camp**

Learning with Dependency Paths

- The shortest path between two entities in a dependency graph offers a very condensed representation of the information needed to assess their relationship.
- A dependency path is represented as a sequence of words interspersed with arrows that indicate the orientation of each dependency, as illustrated in Table 3.1.
- These paths, however, are completely lexicalized and consequently their performance will be limited by data sparsity.
- The solution is to allow paths to use both words and their word classes, similar with the approach taken for the subsequence patterns.

$$\begin{bmatrix} \text{protesters} \\ \text{NNS} \\ \text{Noun} \\ \text{PERSON} \end{bmatrix} \times [\rightarrow] \times \begin{bmatrix} \text{seized} \\ \text{VBD} \\ \text{Verb} \end{bmatrix} \times [\leftarrow] \times \begin{bmatrix} \text{stations} \\ \text{NNS} \\ \text{Noun} \\ \text{FACILITY} \end{bmatrix}$$

Fig. 3.6. Feature generation from dependency path.

The set of features can then be defined as a Cartesian product over words and word classes, as illustrated in Figure 3.6 for the dependency path between ‘protesters’ and ‘station’ in sentence *S*1.

In this representation, sparse or contiguous subsequences of nodes along the lexicalized dependency path (i.e., path fragments) are included as features simply by replacing the rest of the nodes with their corresponding generalizations.

- Examples of features generated by Figure 3.6 are
- “protesters → seized ← stations,”
- “Noun → Verb ← Noun,”
- “Person → seized ← Facility,” or “Person → Verb ← Facility.”
- The total number of features generated by this dependency path is $4 \times 1 \times 3 \times 1 \times 4$.

If $\boldsymbol{x} = x1x2\ldots xm$ and $\boldsymbol{y} = y1y2\ldots yn$ are two relation examples, where xi denotes the set of word classes corresponding to position i, then the number of common features between $\boldsymbol{x}$ and $\boldsymbol{y}$ is computed as in Equation 3.4.

$$K(\mathbf{x}, \mathbf{y}) = \mathbb{1}(m = n) \cdot \prod_{i=1}^{n} c(x_i, y_i) \qquad (3.4)$$

- This is a simple kernel, whose computation takes $O(n)$ time.
- If the two paths have different lengths, they correspond to different ways of expressing a relationship – for instance, they may pass through a different number of predicate argument structures. Consequently, the kernel is defined to be 0 in this case.

As an example, let us consider two instances of the Located relationship,and their corresponding dependency paths:

1. '**his** actions in **Brcko**' (**his** → actions ← in ← **Brcko**).

2. '**his** arrival in **Beijing**' (**his** → arrival ← in ← **Beijing**).

Their representation as a sequence of sets of word classes is given by:

1. x = [$x1$ $x2$ $x3$ $x4$ $x5$ $x6$ $x7$], where $x1$ = {his, PRP, Person}, $x2$ = {→}, $x3$ = {actions, NNS, Noun}, $x4$ = {←}, $x5$ = {in, IN}, $x6$ = {←}, $x7$ = {Brcko, NNP, Noun, Location}

2. 2. y = [$y1$ $y2$ $y3$ $y4$ $y5$ $y6$ $y7$], where $y1$ = {his, PRP, Person}, $y2$ = {→}, $y3$ ={arrival, NN, Noun}, $y4$ = {←}, $y5$ = {in, IN}, $y6$ = {←}, $y7$ = {Beijing, NNP, Noun, Location}

- Based on the formula from Equation 3.4, the kernel is computed as

  $K(x, y) = 3 \times 1 \times 1 \times 1 \times 2 \times 1 \times 3 = 18.$

Experimental Evaluation

- The two relation kernels described above are evaluated on the task of extracting relations from two corpora with different types of narrative.
- In both cases, we assume that the entities and their labels are known. All preprocessing steps – sentence segmentation, tokenization,POS tagging, and chunking – were performed using the OpenNLP package.
- The dependency graph that is input to the shortest path dependecy kernel is obtained from two different parsers:
- The CCG parser outputs a list of functor-argument dependencies, from which head-modifier dependencies are obtained using a straightforward procedure.
- Head-modifier dependencies can be easily extracted from the full parse output of Collins' CFG parser, in which every non-terminal node is annotated with head information.

Interaction Extraction from AIMed

- Comparative experiments on the AIMed corpus, which has been previously used for training the protein interaction extraction systems. It consists of 225 Medline abstracts, of which 200 are known to describe interactions between human proteins, while the other 25 do not refer to any interaction. There are 4084 protein references and around 1000 tagged interactions in this dataset.
- The following systems are evaluated on the task of retrieving protein interactions from AIMed:

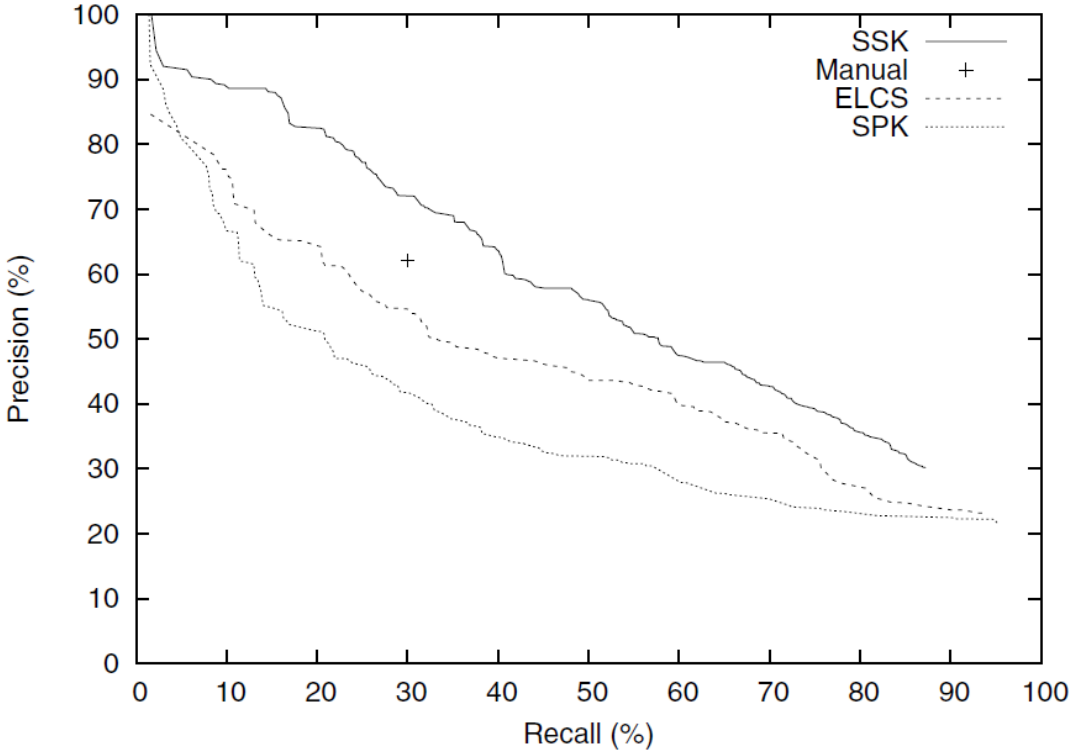
[Manual]: We report the performance of the rule-based system.

[ELCS]: We report the 10-fold cross-validated results as a Precision-Recall (PR) graph.

[SSK]: The subseqeuence kernel is trained and tested on the same splits as ELCS. In order to have a fair comparison with the other two systems, which use only lexical information, we do not use any word classes here.

[SPK]: This is the shortest path dependency kernel, using the head-modifier dependencies extracted by Collins' syntactic parser. The kernel is trained and tested on the same 10 splits as ELCS and SSK.

Precision-Recall curves for protein interaction extractors



Relation Extraction from ACE

- The two kernels are also evaluated on the task of extracting top-level relations from the ACE corpus [12], the version used for the September 2002 evaluation.
- The training part of this dataset consists of 422 documents, with a separate set of 97 documents reserved for testing. This version of the ACE corpus contains three types of annotations: coreference, named entities and relations.
- There are **five types of entities** – Person, Organization, Facility, Location, and Geo-Political Entity – which can participate in **five general, top-level relations**: Role, Part, Located, Near, and Social.
- In total, there are 7,646 intra-sentential relations, of which 6,156 are in the training data and 1,490 in the test data.

Contd..

- Precision and recall values are reported for the task of extracting the five top-level relations in the ACE corpus under two different scenarios:

– **[S1]** This is the classic setting: one multi-class SVM is learned to discriminate among the five top-level classes, plus one more class for the no-relation cases.

– **[S2]** One binary SVM is trained for *relation detection*, meaning that all positive relation instances are combined into one class. The thresholded output of this binary classifier is used as training data for a second multi-class SVM, trained for *relation classification*.

Performance Analysis

Table 3.2. Extraction Performance on ACE.

(Scenario) Method	Precision	Recall	F-measure
(S1) K4	70.3	26.3	38.0
(S1) SSK	73.9	35.2	47.7
(S1) SPK-CCG	67.5	37.2	48.0
(S1) SPK-CFG	**71.1**	**39.2**	**50.5**
(S2) K4	67.1	35.0	45.8
(S2) SPK-CCG	63.7	41.4	50.2
(S2) SPK-CFG	**65.5**	**43.8**	**52.5**

Conclusion

- Mining knowledge from text documents can benefit from using the structured information that comes from entity recognition and relation extraction.
- However, accurately extracting relationships between relevant entities is dependent on the granularity and reliability of the required linguistic analysis.
- We presented two relation extraction kernels that differ in terms of the amount of linguistic information they use.
- Experimental evaluations on two corpora with different types of discourse show that they compare favorably to previous extraction approaches.

References

1. R. J. Mooney, R. C. Bunescu, Mining knowledge from text using information extraction, SIGKDD Explorations (special issue on Text Mining and Natural Language Processing) 7 (1) (2005) 3–10.
2. H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, C. Watkins, Text classification using string kernels, Journal of Machine Learning Research 2 (2002) 419–444.
3. C. D. Fellbaum, WordNet: An Electronic Lexical Database, MIT Press, Cambridge, MA, 1998.
4. L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of the IEEE 77 (2) (1989) 257–286.
5. A. McCallum, D. Freitag, F. Pereira, Maximum entropy Markov models for information extraction and segmentation, in: Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000), Stanford, CA, 2000.
6. J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: Proceedings of 18th International Conference on Machine Learning (ICML-2001), Williamstown, MA, 2001, pp. 282–289.
7. C. Blaschke, A. Valencia, Can bibliographic pointers for known biological data be found automatically? protein interactions as a case study, Comparative and Functional Genomics 2 (2001) 196–206.
8. C. Blaschke, A. Valencia, The frame-based module of the Suiseki information extraction system, IEEE Intelligent Systems 17 (2002) 14–20.
9. R. Bunescu, R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani, Y. W. Wong, Comparative experiments on learning information extractors for proteins and their interactions, Artificial Intelligence in Medicine (special issue on Summarization and Information Extraction from Medical Documents) 33 (2) (2005) 139–155.

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Mining Diagnostic Text Reports by Learning to Annotate Knowledge Roles

Eni Mustafaraj, Martin Hoof, and Bernd Freisleben

Introduction

- Several tasks approached by using text mining techniques, like text categorization, document clustering, or information retrieval, operate on the document level, making use of the so called bag-of-words model.
- Other tasks, like document summarization, information extraction, or question answering, have to operate on the sentence level, in order to fulfill their specific requirements.
- Issues: Data sparsity, Availability of training data, quantity and quality data

Contd..

- In this paper, we present an approach to extract knowledge from text documents containing diagnostic problem solving situations in a technical domain.
- In the proposed approach, we have combined techniques from several areas, including NLP, knowledge engineering, and machine learning to implement a learning framework for annotating cases with knowledge roles.
- The ultimate goal of the approach is to discover interesting problem solving situations that can be used by an experience management system to support new engineers during their working activities.

Contd..

- The experimental results presented in the paper are based on a collection of 500 Microsoft Word documents written in German, amounting to about one million words.
- Several processing steps were required to achieve the goal of case annotation.

(a) transform the documents in an XML format,
(b) extract paragraphs belonging to cases,
(c) perform part-of-speech tagging,
(d) perform syntactical parsing,
(e) transform the results into XML representation for manual annotation,
(f) construct features for the learning algorithm, and
(g) implement an active learning strategy.

Domain Knowledge

- Our domain of interest is predictive maintenance of high voltage rotating electrical machines in the field of power engineering.
- Focus is on textual diagnostic reports.
- Two parties are involved:

The service provider

The customer

Evaluations of the measurements

At $1.9U_N(= 30kV)$, an insulation breakdown occurred on the upper bar of the slot $N°18$, at the slot exit on the NDE side. The breakdown indicates that the bar insulation is seriously weakened. This may be caused by intense discharges due to a malfunction of the slot anti-corona protection.

The measured bypass currents are in a relatively high range indicating a certain surface conductivity. This is due to the fact that the motor was stored in cold area before it was moved to the high voltage laboratory where the temperature and humidity was much higher so that a certain degree of condensation could occur on the surface of the winding.

Domain Concepts

- In some text mining applications, such as text categorization or information retrieval, the goal is often to discover terms specific to the domain that could be used as indices for organizing or retrieving information.
- Domain-specific terms: *insulation*,*discharge*, *slot anti-corona protection*, *conductivity*, or *winding*.

1) The calculated **insulating resistance** values lay in the safe operating area.
2) Compared to the last examination, lower values for the **insulating resistance** were ascertained, due to dirtiness at the surface.

Knowledge Roles

- Knowledge roles are a concept introduced in CommonKADS Schreiber et al.[2000], a knowledge engineering methodology for implementing knowledge based systems.
- Knowledge roles are abstract names that refer to the role a domain concept plays when reasoning about a knowledge task.
- Such tasks are, for example, diagnosis, assessment, monitoring, or planning.

Knowledge Role	**Text Phrase**
Observed Object:	insulating resistance
Symptom:	lower values
Cause:	dirtiness at the surface

Contd..

- It might be argued that one could simply use a combination of keywords to retrieve the information.
- For example, for sentences like that in Figure 4.2, one might write a query as below:

[low | small | high | large] && [value] && [insulating resistance]

- for retrieving symptoms. Or one can search for:

[due to] | [caused by] | [as a result of] . . .

Reasons Why can't be applied

- A large number of words (adjectives, nouns, adverbs, or verbs) can be used to describe changes, and no one can know beforehand which of them is used in the text.
- While verbs are very important for capturing the meaning of a sentence, they also abound in numbers. For example, to express an observation, any of the following verbs can be used: *observe*, *detect*, *show*, *exhibit*, *recognize*, *determine*, *result in*, *indicate*, etc.
- Often, meaning emerges from the relation between different words, instead of the words separately, and this is exactly what we encountered in the diagnostic cases.

Frame Semantics and Semantic Role Labeling

Frame Semantics

- A frame is a "**script-like conceptual structure that describes a particular type of situation, object, or event and the participants involved in it**".
- The structure of a frame contains **lexical units**, **frame elements**, as well as **annotated sentences** for all lexical units that evoke the frame.

Frame with its related components

Frame *Evidence*

Definition: The Support, a phenomenon or fact, lends support to a claim or proposed course of action, the Proposition, where the Domain_of_Relevance may also be expressed.

Lexical units: *argue.v*, *argument.n*, *attest.v*, *confirm.v*, *contradict.v*, *corroborate.v*, *demonstrate.v*, *disprove.v*, *evidence.n*, *evidence.v*, *evince.v*, *from.prep*, *imply.v*, *indicate.v*, *mean.v*, *prove.v*, *reveal.v*, *show.v*, *substantiate.v*, *suggest.v*, *testify.v*, *verify.v*

Frame Elements:

Proposition [PRP]	This is a belief, claim, or proposed course of action to which the Support lends validity.
Support [SUP]	Support is a fact that lends epistemic support to a claim, or that provides a reason for a course of action.
...	

Examples:

And a [$_{SUP}$ sample tested] **REVEALED** [$_{PRP}$ some inflammation].
It says that [$_{SUP}$ rotation of partners] does not **DEMONSTRATE** [$_{PRP}$ independence].

Semantic Role Labeling

- SRL is approached as a learning task.
- For a given target verb in a sentence, the syntactic constituents expressing semantic roles associated to this verb need to be identified and labeled with the right roles.
- SRL systems usually divide sentences word-by-word or phrase-by-phrase and for each of these instances calculate many features creating a feature vector.
- The feature vectors are then fed to supervised classifiers, such as support vector machines, maximum entropy, or memory-based learners.

Frames and Roles for Annotating Cases

- A knowledge task like diagnosis or monitoring is not equivalent to a semantic frame. It is usually divided into several components, which in turn can be regarded equivalent to semantic frames.
- By analyzing the textual episodes of diagnostic evaluations, we noticed that they typically contain a list of observations, explanations based on evidence, and suggestions to perform some activities.
- Thus, we consulted FrameNet for frames like Observation, Change, Evidence, or Activity.

Access

- We initially look for the most frequent verbs in our corpus, and by consulting several sources (since the verbs are in German), such as [15], VerbNet,3 and FrameNet.
- We connect every verb with a frame, and try to map between semantic roles in a frame and knowledge roles we are interested in.
- FrameNet was only used as a lexical resource for consultation, that is, to find out which frames are evoked by certain lexical units, and what the related semantic roles are.

Annotation Example

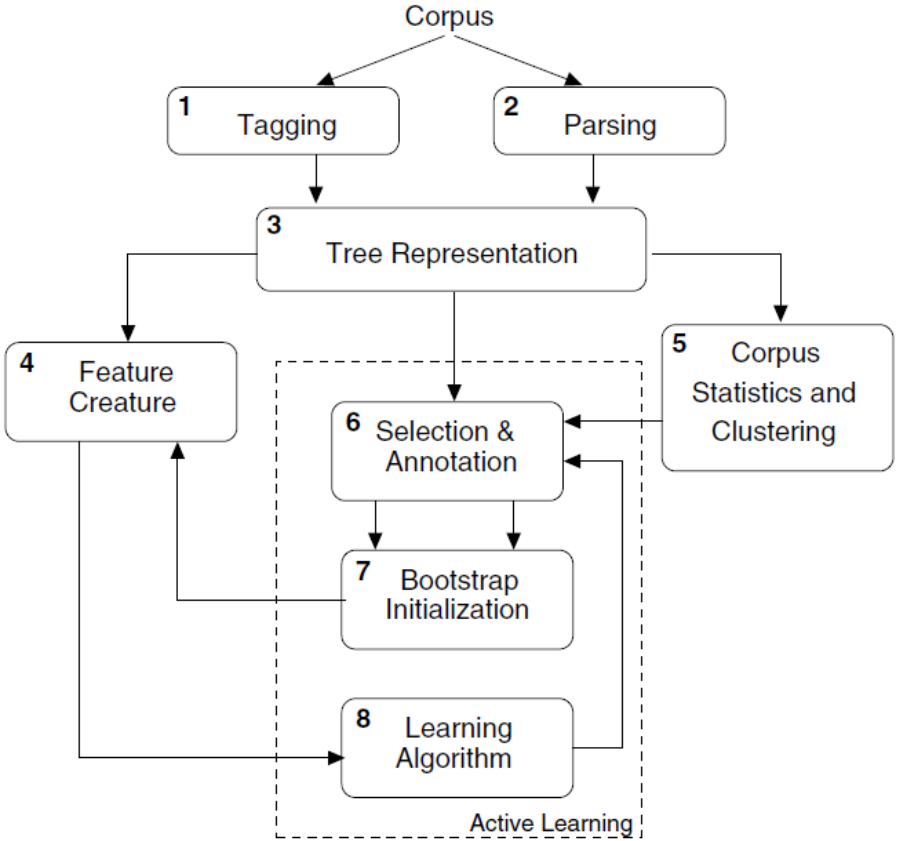
Table 4.1. Some phrases annotated with the role *Cause*.

German Phrase	English Translation	Frequency
Verschmutzungseinflüsse	influences of pollution	10
leitende Verschmutzungen	conducting pollutions	8
Ionisation in Klemmenbereich	ionization in the terminal area	3
äussere Entladungen	external discharges	1

Information Extraction

- Information extraction (IE), often regarded as a restricted form of natural language understanding, predates research in text mining.
- In IE it is usually known in advance what information is needed, and part of text is extracted to fill in slots of a predefined template.
- Usually, methods used by IE have been based on shallow NLP techniques, trying to extract from a corpus different types of syntactic rules that match syntactic roles to semantic categories.

Learning to Annotate Cases with Knowledge Roles



Document Preparation

```
<section title="Measurements">
  <subsection title="Stator_Winding">
    <measurement title="Visual_Control">
      <submeasurement title="Overhang_Support">
        <evaluation>
          Die Wickelkopfabsttzung AS und NS befand sich in einem ...
        </evaluation>
        <action>Keine</action>
      </submeasurement>
...
```

Tagging

- The part-of-speech (POS) tagger (TreeTagger4) used is a probabilistic tagger with parameter files for tagging several languages: **German**, **English**, **French**, or **Italian**.

Es	PPER	es
liegt	VVFIN	liegen
insgesamt	ADV	insgesamt
ein	ART	ein
guter	ADJA	gut
äusserer	ADJA	äuβer
Wicklungszustand	NN	<unknown>
vor	PTKVZ	vor
.	$.	.

Correct and erroneous tagging

an	APR	an
ca.	ADV	ca.
50	CARD	50
%	NN	%

(	$(	(
ca	NE	<unknown>
.	$.	.
20	CARD	20

A handy solution for correcting spelling errors is to use a string similarity function, available in many programming language libraries.

Results of experiments for the contribution of stem information on learning.

Experiment	Recall	Precision
a) no stems (only words)	90.38	92.32
b) only stems	91.29	93.40

Table 4.3. Original words (first row), words composed of stems (second row).

Ableitstrom*werte*, Gesamtstrom*werte*, Isolationswiderstands*werte*, Isolation-sstrom*werte*, Kapazitäts*werte*, Ladestrom*werte*, Strom*werte*n, Verlustfaktoranfangs*wert*, etc.
Ableit-Strom-Wert, Gesamt-Strom-Wert, Isolation-Widerstand-Wert, Isolation-Strom-Wert, Kapazität-Wert, Lade-Strom-Wert, Strom-Wert, Verlustfaktor-Anfang-Wert, etc.

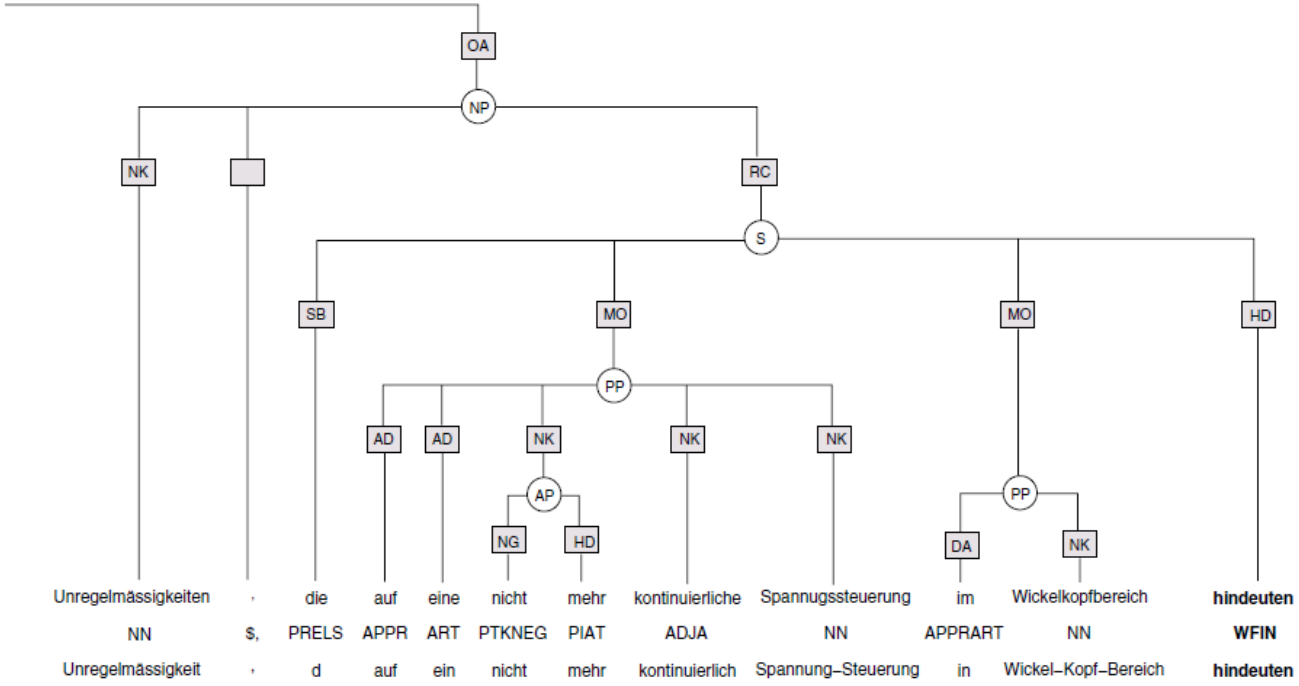
Parsing

- Syntactical parsing is one of the most important steps in the learning framework, since the produced parse trees serve as input for the creation of features used for learning.
- Stanford Parser
- BitPar Parser
- Sleepy Parser

Parsers

- **Stanford Parser** - tackles the task of generating parse trees from unlabeled data independently of the language. For the moment, the parser is distributed with parameter files for parsing English, German, and Chinese.
- **BitPar Parser** – It is composed of two parts, the parser itself and the parameter files (chart rules, lexicon, etc.). It has Robust Performance, extended to triples of domain dependent words.
- **Sleepy Parser** - It has been specifically tuned for the German language, and while it is a statistical parser like the others, it uses different annotation schemas and incorporates grammatical functions (SB–subject, OC–clausal object, MO–modifier, HD–head, etc.) or long-distance dependencies between terms.

Tree Representation



XML Representation

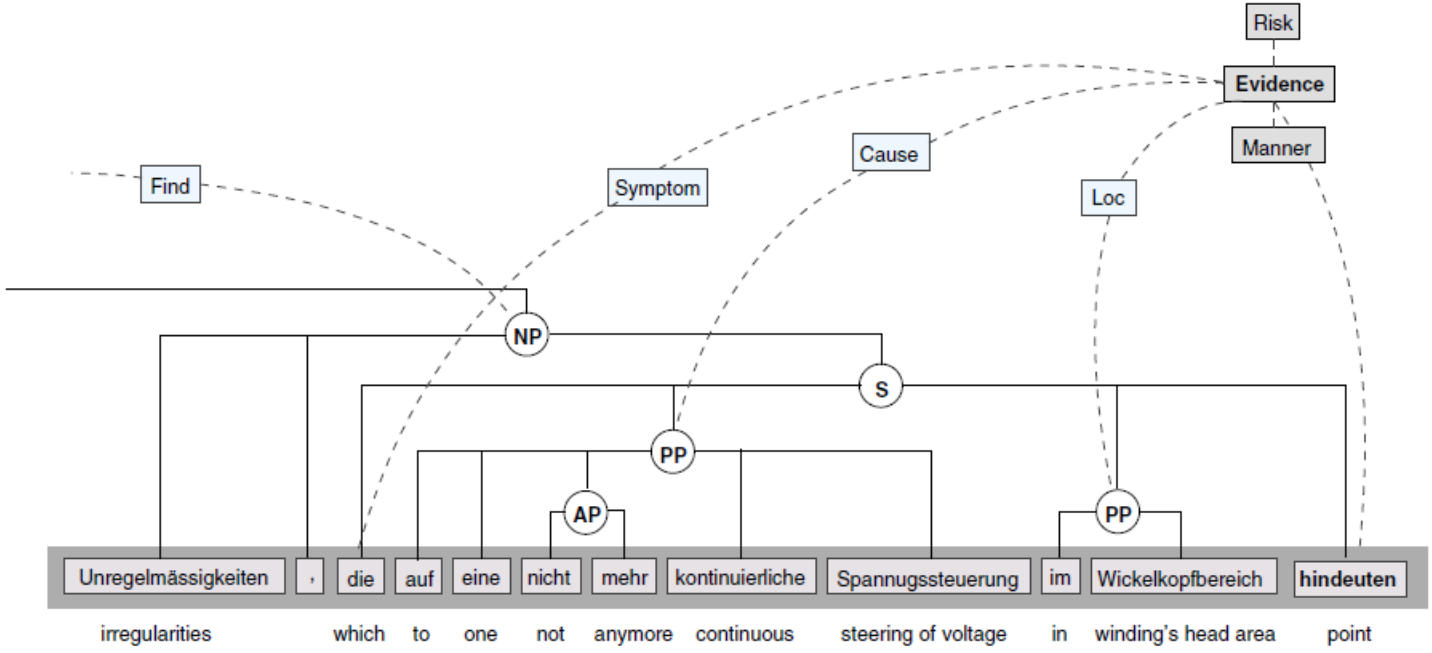
```
...
 <t lemma="Spannung-Steuerung" word="Spannungssteuerung" pos="NN"
      id="sentences._108_28" />
 <t lemma="in" word="im" pos="APPRART"
      id="sentences._108_29" />
 <t lemma="Wickel-Kopf-Bereich" word="Wickelkopfbereich" pos="NN"
      id="sentences._108_30" />
 <t lemma="hindeuten" word="hindeuten" pos="VVFIN" id="sentences._108_31" />
</terminals>
<nonterminals>
 <nt id="sentences._108_500" cat="PP">
  <edge idref="sentences._108_3" label="NK" />
  <edge idref="sentences._108_2" label="DA" />
  <edge idref="sentences._108_1" label="DA" />
 </nt>
...
```

Fig. 4.11. XML representation of a portion of the parse tree from Figure 4.10.

Feature Creation

Phrase type **NN**
Grammatical function **NK**
Terminal (is the constituent a terminal or non-terminal node?) **1**
Path (path from the target verb to the constituent, denoting u(up) and d(down) for the direction) **uSdPPd**
Grammatical path (like Path, but instead of node labels, branch labels are considered) **uHDdMOdNK**
Path length (number of branches from target to constituent) **3**
Partial path (path to the lowest common ancestor between target and constituent) **uPPuS**
Relative Position (position of the constituent relative to the target) **left**
Parent phrase type (phrase type of the parent node of the constituent) **PP**
Target (lemma of the target word) **hindeuten**
Target POS (part-of-speech of the target) **VVFIN**
Passive (is the target verb passive or active?) **0**
Preposition (the preposition if the constituent is a PP) **none**
Head Word (for rules on head words refer to [5]) **Spannung-Steuerung**
Left sibling phrase type **ADJA**
Left sibling lemma **kontinuierlich**
Right sibling phrase type **none**
Right sibling lemma **none**
Firstword, Firstword POS, Lastword, Lastword POS (in this case, the constituent has only one word, thus, these features get the same values: Spannung-Steuerung and NN. For non-terminal constituents like PP or NP, first word and last word will be different.)
Frame (the frame evoked by the target verb) **Evidence**
Role (this is the class label that the classifier will learn to predict. It will be one of the roles related to the frame or none, for an example refer to Figure 4.12.) **none**

Annotation


Risk
Evidence
Manner
Cause
Find
Symptom
Loc
NP
S
PP
AP
PP
Unregelmässigkeiten
,
die
auf
eine
nicht
mehr
kontinuierliche
Spannugssteuerung
im
Wickelkopfbereich
hindeuten
irregularities
which
to
one
not
anymore
continuous
steering of voltage
in
winding's head area
point

XML Representation of an annotated frame

```
<frames>
 <frame name="Evidence" id="sentences._108__f1">
  <target><fenode idref="sentences._108_31"/></target>
   <fe name="Symptom" id="sentences._108_f1_e1">
    <fenode idref="sentences._108_22"/>
   </fe>
   <fe name="Cause" id="sentences._108__f1_e2">
    <fenode idref="sentences._108_509"/>
   </fe>
    <fe name="Loc" id="sentences._108__f1_e5">
    <fenode idref="sentences._108_510"/>
   </fe>
...
```

Active Learning

a) Divide the corpus in clusters of sentences with the same target verb. If a cluster has fewer sentences than a given threshold, group sentences with verbs evoking the same frame into the same cluster.

b) Within each cluster, group the sentences (or clauses) with the same parse subtree together.

c) Select sentences from the largest groups of the largest clusters and present them to the user for annotation.

d) Bootstrap initialization: apply the labels assigned by the user to groups of sentences with the same parse sub-tree.

e) Train all the classifiers of the committee on the labeled instances; apply each trained classifier to the unlabeled sentences.

f) Get a pool of instances where the classifiers of the committee disagree and present to the user the instances belonging to sentences from the next largest clusters not yet manually labeled.

g) Repeat steps d)–f) a few times until a desired accuracy of classification is achieved.

Examples of sentences with the same structure

[$_{PP}$ Im Nutaustrittsbereich] wurden [$_{NP}$ stärkere Glimmentladungsspuren] festgestellt.
In the area of slot exit stronger signs of corona discharges were detected.

[$_{PP}$ Bei den Endkeilen] wurde [$_{NP}$ ein ausreichender Verkeildruck] festgestellt.
At the terminals' end a sufficient wedging pressure was detected.

[$_{PP}$ An der Schleifringbolzenisolation] wurden [$_{NP}$ mechanische Beschädigungen] festgestellt.
On the insulation of slip rings mechanical damages were detected.

[$_{PP}$ Im Wickelkopfbereich] wurden [$_{NP}$ grossflächige Decklackablätterungen] festgestellt.
In the winding head area extensive chippings of the top coating were detected.

Evaluations

Learning Performance on the Benchmark Datasets

Table 4.4. Statistics for the benchmark datasets.

Subcorpus	Cases No.	Sentences No.	Unique Sentences No.	Annotated Roles No.
Isolation Current	491	1134	585	1862
Wedging System	453	775	602	1751

Table 4.5. Learning results for the benchmark datasets.

Subcorpus	Recall	Precision	$F_{\beta=1}$ measure
Isolation Current	0.913	0.934	0.92
Wedging System	0.838	0.882	0.86

Active Learning versus Uniform Random Selection

Table 4.6. Random Learning Results.

Sentences No.	Recall	Precision	$F_{\beta=1}$ measure
10	0.508	0.678	0.581
20	0.601	0.801	0.687
30	0.708	0.832	0.765
40	0.749	0.832	0.788

Table 4.7. Active Learning Results.

Sentences No.	Recall	Precision	$F_{\beta=1}$ measure
10	0.616	0.802	0.697
20	0.717	0.896	0.797
30	0.743	0.907	0.817
40	0.803	0.906	0.851

Bootstrapping Based on Other Sets

Table 4.8. Results for bootstrapping based on other labeled sets

Training File	Testing File	Recall	Precision
Isolation Current	Wedging System	0.765	0.859
Wedging System	Isolation Current	0.642	0.737

Conclusions

- We have presented an approach for extracting knowledge from text documents containing descriptions of knowledge tasks in a technical domain.
- Knowledge extraction in our approach is based on the annotation of text with knowledge roles (a concept originating in knowledge engineering), which we map to semantic roles found in frame semantics.
- The framework implemented for this purpose is based on deep NLP and active learning.
- Experiments have demonstrated a robust learning performance, and the obtained annotations were of high quality.

References

1. A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of the Workshop on Computational Learning Theory, COLT '98, Madison, WI*, pages 92–100, 1998.
2. A. J. Carlson, C. M. Cumby, N. D. Rizzolo, J. L. Rosen, and D. Roth. SNoW: Sparse Network of Winnow. 2004.
3. X. Carreras and L. Màrquez. Introduction to the coNLL shared task: Semantic role labeling. In *Proc. of 8th Conference of Natural Language Learning*, pages 89–97, Boston, MA, 2004.
4. X. Carreras and L. Màrquez. Introduction to the coNLL-2005 shared task: Semantic role labeling. In *Proc. of 9th Conference of Natural Language Learning*, pages 152–165, Ann Arbor, MI, June 2005.
5. M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.
6. W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. TiMBL: Tilburg Memory Based Learner. 2004.
7. A. Dubey. *Statistical Parsing for German*. PhD thesis, University of Saarland, Germany, 2003.
8. A. Dubey. What to do when lexicalization fails: Parsing German with suffix analysis and smoothing. In *Proc. of 43rd Annual Meeting of ACL, Ann Arbor, MI*, pages 314–321, 2005.
9. M. Ellsworth, K. Erk, P. Kingsbury, and S. Padó. PropBank, SALSA, and FrameNet: How design determines product. In *Proc. of the LREC 2004 Workshop on Building Lexical Resources from Semantically Annotated Corpora, Lisbon, Portugal*, 2004.

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

A Case Study in Natural Language Based Web Search

Giovanni Marchisio, Navdeep Dhillon, Jisheng Liang, Carsten Tusk, Krzysztof Koperski, Thien Nguyen, Dan White, and Lubos Pochman

Introduction

- Is there a public for natural language based search?
- This study, based on our experience with a Web portal, attempts to address criticisms on the lack of scalability and usability of natural language approaches to search.
- Our solution is based on InFact®, a natural language search engine that combines the speed of keyword search with the power of natural language processing.
- InFact performs clause level indexing, and offers a full spectrum of functionality that ranges from Boolean keyword operators to linguistic pattern matching in real time, which include recognition of syntactic roles, such as subject/object and semantic categories, such as people and places.

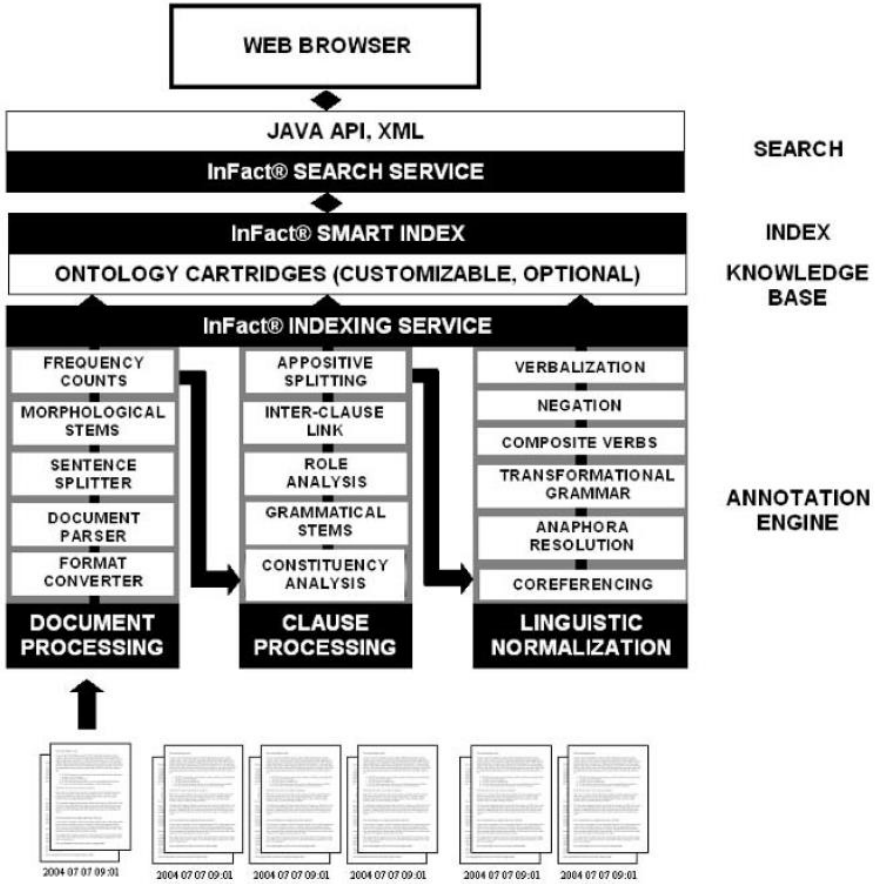
Contd..

- InFact relies on a new approach to text parameterization that captures many linguistic attributes ignored by standard inverted indices.
- Examples are syntactic categories (parts of speech), syntactical roles (such as subject, objects, verbs, prepositional constraints, modifiers, etc.) and semantic categories (such as people, places, monetary amounts, etc.).

Contd..

- The goal of our experiment was to demonstrate that, once scalability barriers are overcome, a statistically significant percentage of Web users can be converted from keyword search to natural language based search.
- InFact has been the search behind the GlobalSecurity.org site (www.globalsecurity.org) for the past six months.
- According to the Alexa site (www.alexa.com), GlobalSecurity.org has a respectable overall traffic rank (no. 6,751 as of Feb 14, 2006). Users of the site can perform keyword searches, navigate results by action themes, or enter explicit semantic queries.
- An analysis of query logs demonstrate that all these non-standard information discovery processes based on NLP have become increasingly popular over the first six months of operation.

InFact System Overview



Indexing

- InFact's Indexing Service performs in order:

1) document processing,

2) clause processing, and

3) linguistic normalization.

Document Processing

- The first step in document processing is format conversion, which we handle through our native format converters, or optionally via search export conversion software from StellantTM (www.stellent.com), which can convert 370 different input file types.
- Customized document parsers address the issue that a Web page may not be the basic unit of content, but it may consist of separate sections with an associated set of relationships and metadata.
- For instance a blog post may contain blocks of text with different dates and topics.
- The challenge is to automatically recognize variations from a common style template, and segment information in the index to match zones in the source documents.
- We extract morphological stems and compute frequency counts, which are then entered in the index.

Clause Processing

- The indexing service takes the output of the sentence splitter and feeds it to a deep linguistic parser.
- A sentence may consist of multiple clauses.
- Unlike traditional models that store only term frequency distributions, InFact performs clause level indexing and captures syntactic category and roles for each term, and grammatical constructs, relationships, and inter-clause links that enable it to understand events.

Contd..

- Our parser performs a full constituency and dependency analysis, extracting part-of-speech (POS) tags and grammatical roles for all tokens in every clause.
- In the process, tokens undergo grammatical stemming and an optional, additional level of tagging.
- For instance, when performing grammatical stemming on verb forms, we normalize to the infinitive, but we may retain temporal tags (e.g., past, present, future), aspect tags (e.g., progressive, perfect), mood/modality tags (e.g., possibility, subjunctive,irrealis, negated, conditional, causal) for later use in search.

Example

- "Appointed commander of the Continental Army in 1775, George Washington molded a fighting force that eventually won independence from Great Britain" consists of three clauses, each containing a governing verb (appoint, mold, and win).
- InFact decomposes it into a primary clause ("George Washington molded a fighting force") and two secondary clauses, which are related to the primary clause by an appositive construct ("Appointed commander of the Continental Army in 1775") and a pronoun ("that eventually won independence from Great Britain"), respectively.
- Each term in each clause is assigned a syntactic category or POS tag (e.g., noun, adjective, etc.) and a grammatical role tag (e.g., subject, object, etc.).

Linguistic Normalization

- We apply normalization rules at the syntactic, semantic, or even pragmatic level.
- Our approach to coreferencing and anaphora resolution make use of syntactic agreement and/or binding theory constraints, as well as modeling of referential distance, syntactic position, and head noun.

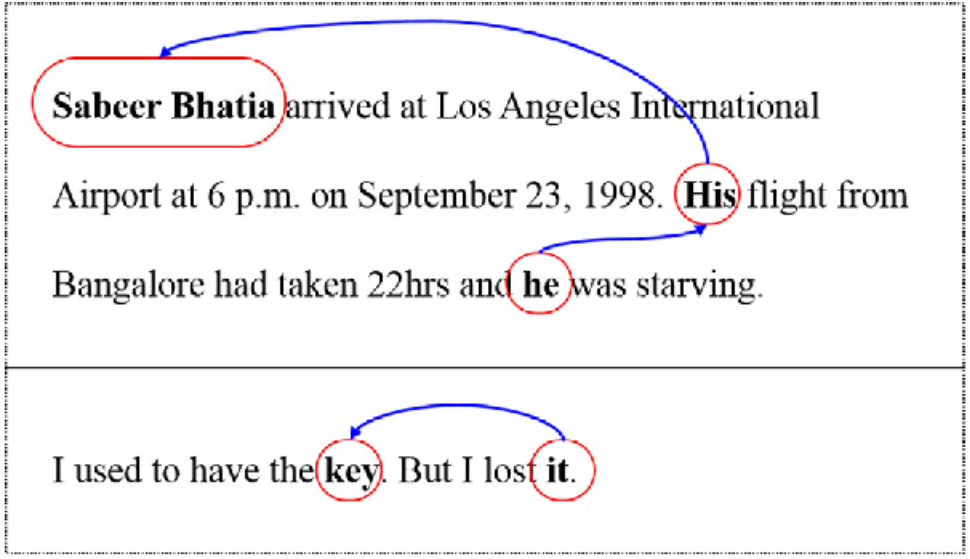


Fig. 1. Anaphora resolution as a sequential mention connection problem.

Storage

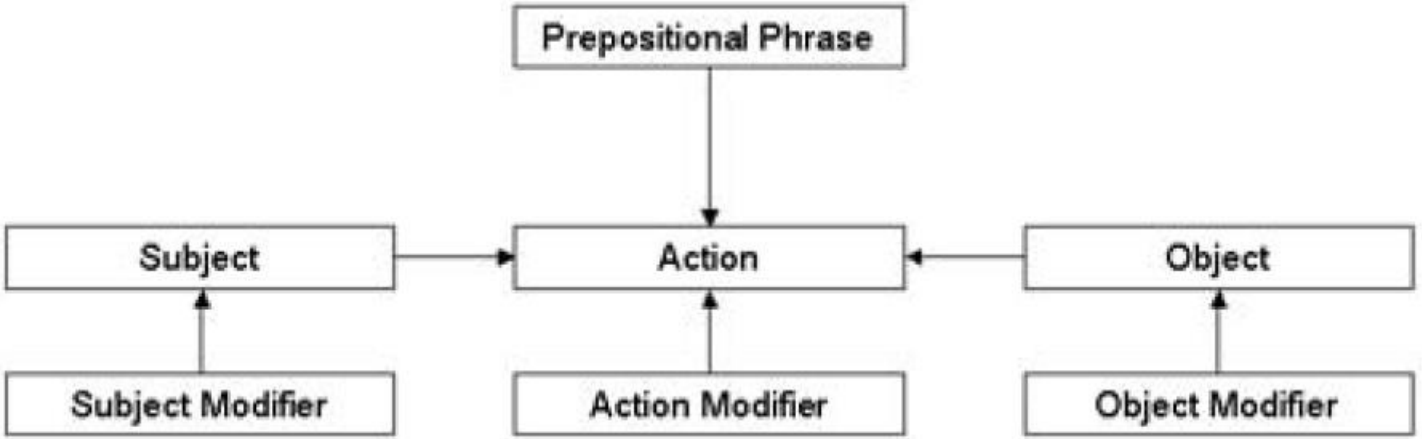
- (Label Modifier Root POS Head-label Role Antecedent [Attributes])

Label is a unique numeric ID;
Modifier is a term in the sentence;
Root is the root form (or category) of the modifier;
POS is its lexical category;
Headlabel is the ID of the term that modifier modifies;
Role specifies the type of dependency relationship between head and modifier, such as subject, complement, etc;
Antecedent is the antecedent of the modifier;
Attributes is the list of semantic attributes that may be associated with the modifier

Parse Tree Representation

Label	Modifier	Root	POS	Head Label	Role	Antecedent	Attributes
1	Appointed	Appoint	V				
2	commander		N	1	Obj		Person/title
3	of		Prep	2	Mod		
4	the		Det	5	Det		
5	Continental Army		N	3	Pcomp		Organization/name
6	in		Prep	1	Mod		
7	1775		N	6	Pcomp		Numeric/date
8	George Washington		N	9	Subj		Person/name
9	molded	mold	V				
10	a		Det	12	Det		
11	fighting		A	12	Mod		
12	force		N	9	Obj		
13	that		N	15	Subj	12	
14	eventually		A	15	Mod		
15	won	win	V				
16	independence		N	15	Obj		
17	from		Prep	16	Mod		
18	Great Britain		N	17	Pcomp		Location/country

Indexing Structure



Index-Abstraction

Subject	Subject-Modifier	Object	Object-Modifier	Verb	Verb-Modifier	Prep	Pcomp	Dist	Neg
		Washington	George	appoint				1	F
		commander		appoint				1	F
		Army	Continental	appoint				3	F
				appoint		in	1775	2	F
Washington	George	force	fighting	mold				2	F
force	fighting	independence		win				1	F
				win		from	Great Britain	3	F
				win	eventually			1	F

Index Structures

- InFact stores the normalized triplets into dedicated index structures that

• are optimized for efficient keyword search

• are optimized for efficient cross-document retrieval of arbitrary classes of relationships or events (see examples in the next section)

• store document metadata and additional ancillary linguistic variables for filtering of search results by metadata constraints (e.g., author, date range), or by linguistic attributes (e.g., retrieve negated actions, search subject modifier field in addition to primary subject in a relationship search)

• (optionally) superimposes annotations and taxonomical dependencies from a custom ontology or knowledge base.

Contd..

- InFact stores “soft events” instead of fitting textual information into a rigid relational schema that may result in information loss. “Soft events” are data structures that can be recombined to form events and relationships.
- “Soft events” are pre-indexed to facilitate thematic retrieval by action, subject, and object type.
- For instance, a sentence like “The president of France visited the capital of Tunisia” contains evidence of
- 1) a presidential visit to a country’s capital and
- 2) diplomatic relationships between two countries.

Search

- InFact employs a highly expressive query language (IQL or InFact Query Language) that combines the power of grammatical roles with the flexibility of Boolean operators, and allows users to search for actions, entities, relationships, and events.
- InFact represents the basic relationship between two entities with an expression of the kind:

Subject Entity > Action > Object Entity,

Entity 1 <> Action <> Entity 2

- For instance, the query "**** > eat > cake***" will retrieve a list of anybody or anything that eats a cake; and a query like "***John > * > Jane***" will retrieve a list of all uni-directional relationships between John and Jane.

Contd..

- InFact query language, we can search for:
- **Any relationships involving an entity of interest**

For example, the query "*George Bush* <> * <> * " will retrieve any events involving "*George Bush*" as subject or object

- **Relationships between two entities or entity types**

For example, the query "*China* <> * <> *Afghan** " will retrieve all relationships between the two countries.

- **Events involving one or more entities or types**

For example, the query "*Pope* > *visit* > *[country]*" will return all instances of the Pope visiting a country.

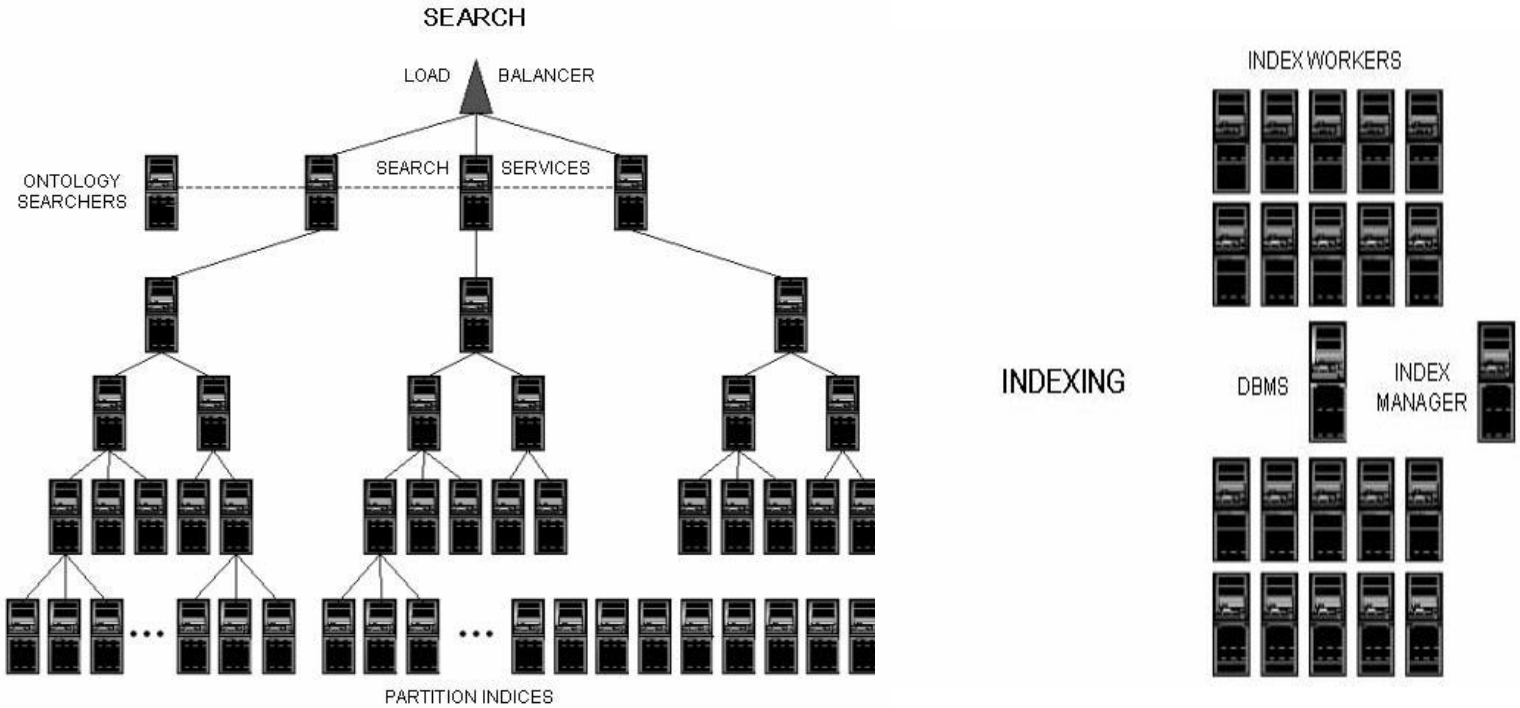
- **Events involving a certain action type.**

"Action types" are groups of semantically linked actions. For example, query "*[Person]* > *[Communication]* > *[Person]*" will retrieve all events involving communication between two people.

Specifying document-level constraints

- **Document metadata tags** – lists of returned actions, relationships or events are restricted to documents that contain the specified metadata values.
- **Boolean keyword expressions** – lists of returned actions, relationships or events are restricted to documents that contain the specified Boolean keyword expressions.
- For instance, a query like: *[Organization/Name] > buy > [Organization/Name]^[money]; energy NOT oil* will return documents that mentions a corporate acquisition with a specific monetary amount, and also contain the keyword "energy" but do not contain the keyword "oil."

Architecture and Deployment



The GlobalSecurity.org Experience

- InFact started powering the GlobalSecurity.org Web site on June 22, 2005.
- Based in Alexandria, VA, and "launched in 2000, GlobalSecurity.org is the most comprehensive and authoritative online destination for those in need of both reliable background information and breaking news . . .GlobalSecurity.org's unique positioning enables it to reach both a targeted and large diversified audience.
- The content of the website is updated hourly, as events around the world develop, providing indepth coverage of complicated issues.
- The breadth and depth of information on the site ensures a loyal repeat audience.
- The director of GlobalSecurity.org, John Pike, regularly provides commentary and analysis on space and security issues to PBS, CNN, MSNBC, Fox, ABC, CBS, NBC, BBC, NPR, and numerous print and online publications.

Operational Considerations

- When preparing the GlobalSecurity.org deployment, one of our prime concerns was the response time of the system.
- For this reason, we kept the data size of the partition indices small enough so that most operations occur in memory and disk access is minimal.
- We split the GlobalSecurity.org data across two index chunks, each containing roughly 14 GB of data in each partition index.
- Another concern was having sufficient capacity to handle the user load.
- To account for future user traffic, we specified the deployment for 2-3 times the maximum expected load of about 11,000 queries per day.
- Another area of concern was the distribution of query types. Our system has significantly varying average response time and throughput (measured in queries/ minute) depending on the type of queries being executed.

Usability Considerations

GlobalSecurity.org

Education | Jobs | Salary Center | Travel | Autos | Gifts

Reliable Security Information

blackhawk

Submit | Advanced

- Military
- WMD
- Intelligence
- Homeland Security
- Space
- Public Eye

CURRENT NEWS

- **Afghanistan, NATO, EU & Drug Trade** RFE/RL 14 Feb 2006 -- NATO & the EU agree that illegal narcotics trade is one of the biggest threat to Afghanistan's recovery
- **US Denies Plot to Destabilize Hamas Govt** VOA 14 Feb 2006 -- The US is denying it is in secret talks with Israel on ways to destabilize a Hamas-led govt.
- **Georgia & Peacekeepers** RFE/RL 14 Feb 2006 -- Georgia's parliament is likely to vote to demand the withdrawal of Russian peacekeepers in S Ossetia
- **Russia & Hazing Outrage** RFE/RL 14 Feb 2006 -- Prosecutors in the republic of Bashkortostan opened an investigation into a conscript's death
- **Hussein, Codefendant Hunger Strike** RFE/RL 14 Feb 2006 -- Saddam Hussein announced that he was

IN THE NEWS

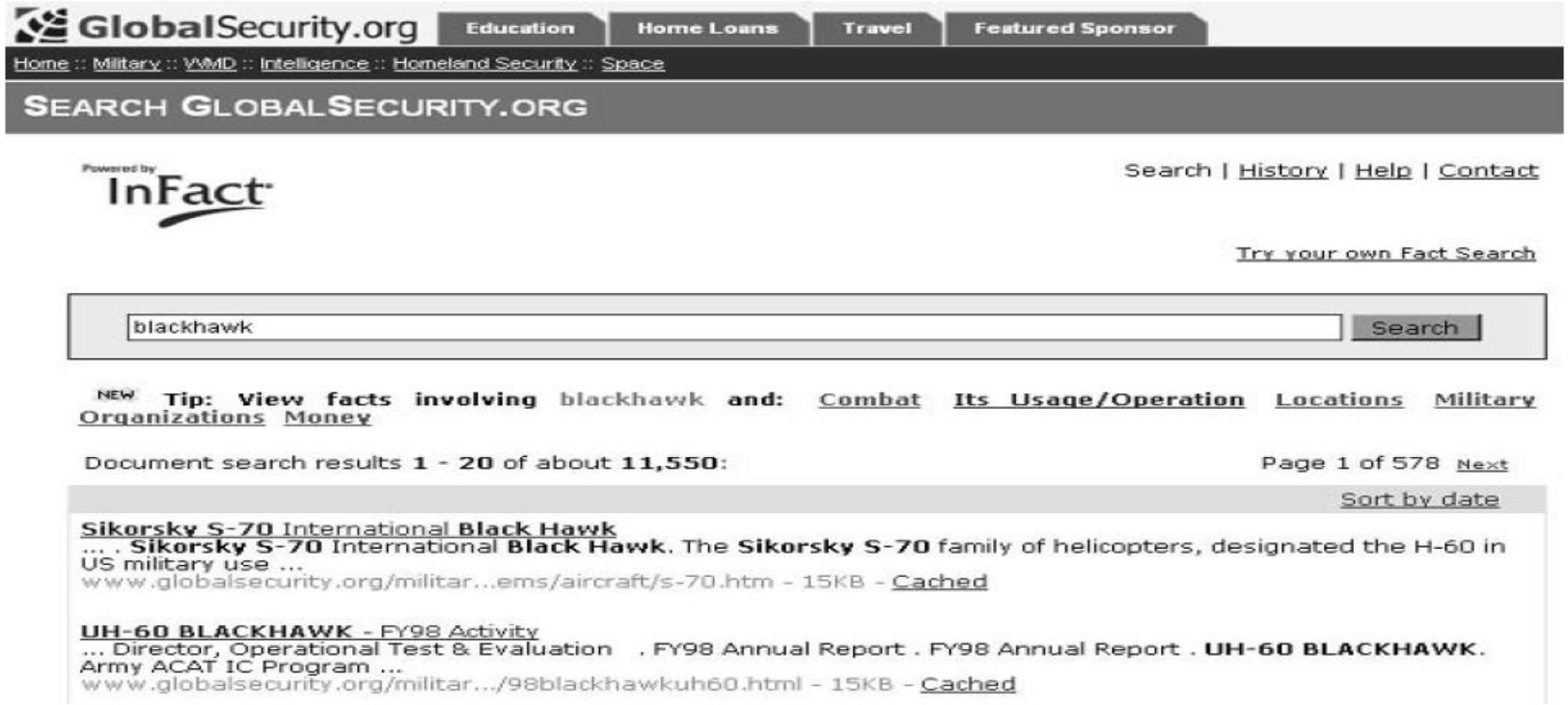
Interceptor Body Armor

Natanz, Iran

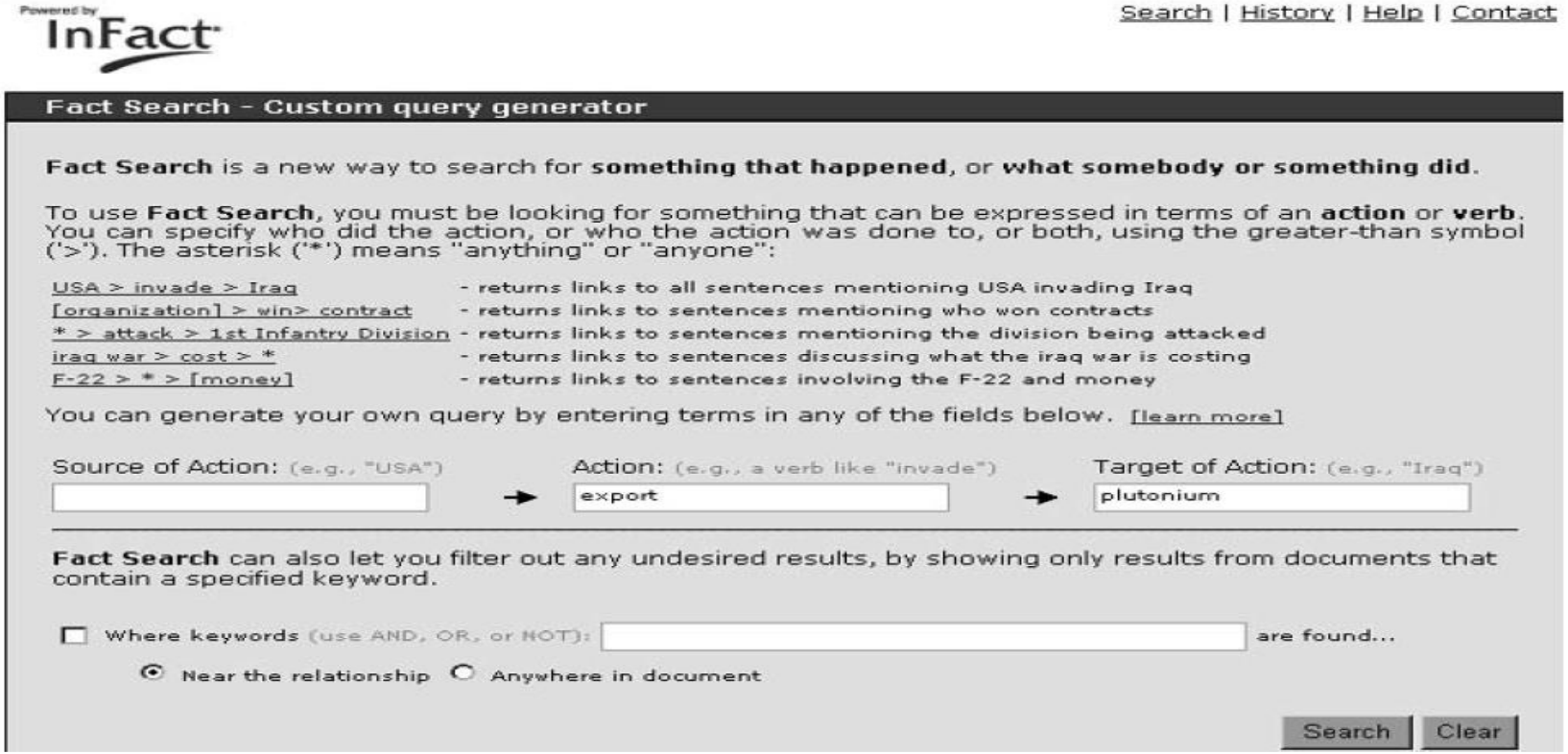
HOT SEARCH

- Iran & Nuclear Program

Keyword search result



Fact search


Powered by InFact
Search | History | Help | Contact
Fact Search - Custom query generator
Fact Search is a new way to search for something that happened, or what somebody or something did.
To use Fact Search, you must be looking for something that can be expressed in terms of an action or verb. You can specify who did the action, or who the action was done to, or both, using the greater-than symbol ('>'). The asterisk ('*') means "anything" or "anyone":
USA > invade > Iraq - returns links to all sentences mentioning USA invading Iraq
[organization] > win> contract - returns links to sentences mentioning who won contracts
* > attack > 1st Infantry Division - returns links to sentences mentioning the division being attacked
iraq war > cost > * - returns links to sentences discussing what the iraq war is costing
F-22 > * > [money] - returns links to sentences involving the F-22 and money
You can generate your own query by entering terms in any of the fields below. [learn more]
Source of Action: (e.g., "USA")
Action: (e.g., a verb like "invade")
export
Target of Action: (e.g., "Iraq")
plutonium
Fact Search can also let you filter out any undesired results, by showing only results from documents that contain a specified keyword.
Where keywords (use AND, OR, or NOT):
are found...
Near the relationship
Anywhere in document
Search
Clear

InFact Custom Query Generator

SEARCH GLOBALSECURITY.ORG

Powered by InFact

Search | History | Help | Contact

Try your own Fact Search

* > export > plutonium Search

Fact Search results **1 - 35**:

View Report Go Sort page by: Source Sort

Source	Action	Target
North Korea	smuggle[10]	plutonium : from Russia
North Korea	smuggle[6]	56 kilograms : of plutonium enough for 7-9 atomic bomb from Russia
North Korea	smuggle	Russian : plutonium
four : case : of real plutonium weapons-usable material	smuggle : out of former Soviet Union	four : case : of real plutonium weapons-usable material
three smalltime : crook	smuggle : In August 1994	some : 363 grams : of plutonium from Moscow to Munich on Lufthansa aircraft
current	trade in	weapon illicit grade plutonium : serve
individual	smuggle	plutonium : out of Eastern Europe uranium
money-hungry Russian : intelligence officer	smuggle	weapons-grade : plutonium : into Germany in August, 1994

Facts involving the “blackhawk” helicopter and military organizations.

Chinook All : BLACK HAWK	include	160th Special Operations Aviation Regiment 101st Airborne Division 10th Mountain Division 82nd Airborne Division
Marine Corps	inspect : across airfield	rigging : on Blackhawk
U.S. Army	install	UH-60 : in future ALQ-211 : on CH-47
U.S. Army	introduce : for example	fly-by-wire : capability : to Army Apache Black Hawk fleet
air force	investigate	Black Hawk fratricide : incident
Troops : from 2nd Battalion 505th Parachute Infantry Regiment 82nd Airborne Division	kick off	Operation Desert Lion : with air assault from Chinook Black Hawk helicopter
Black Hawk : helicopter	land : As part of Operation Falcon Sweep	Shakaria Soldier : of 2nd Battalion 502nd Infantry Regiment
40 : CH-47 UH-60 AH-1	lift	1st Brigade : into
211th Aviation Group	maintain	UH-60 Blackhawk AH-64 Apache
air force : contractor	maintain	Blackhawk : helicopter Army : Apache

Facts involving the “blackhawk” helicopter and money

Date	Source	Action	Target
02/23/2004	$14.6 billion	buy	796 helicopter additional : Black Hawk
10/16/1998	$690 million : appropriation : for fiscal year 1999	buy : for Colombian police extra $190 million for U.S. Customs Service $90 million for enhanced inspection surveillance along U.S.-Mexico border	six UH-60 Black Hawk : helicopter
06/05/1995	Army UH-60 helicopter : trip	almost : cost[2]	more : $1,600 : than car
04/07/2006	Black Hawk : Upgrade - Program	cost	increased : $2,922.5 million
01/02/2004	Blackhawk : helicopter	cost	$8.6 million
06/27/2003	additional : investment : of $331 million for additional spare part	increase[2]	readiness : of Apache Blackhawk helicopter
01/01/2001	FY 2002 : decrease : of $28.2 million	reflect[2]	reduced depot maintenance : requirement : for UH-60 helicopter for UH-1 helicopter retirement pending
02/02/2004	8 new Black Hawk $124.8 : aircraft	upgrade : for 1st	5 : Black Hawk : to UH-60M $78.3 model
06/11/1996	$75,000,000 : for Blackhawk	advance	Rotary Wing Aircraft : blackhawk procurement

InFact: each fact is hyperlinked to the exact location

Date	Source	Action	Target
02/23/2004	$14.6 billion	buy	796 helicopter additional : Black Hawk
10/16/1998	$690 million : appropriation : for fiscal year 1999	buy : for Colombian police extra $190 million for U.S. Customs Service $90 million for enhanced inspection surveillance along U.S.-Mexico border	six UH-60 Black Hawk : helicopter
06/05/1995	Army UH-60 helicopter : trip	almost : cost[2]	more : $1,600 : than car
04/07/2006	Black Hawk : Upgrade - Program	cost	increased : $2,922.5 million
01/02/2004	Blackhawk : helicopter	cost	$8.6 million
06/27/2003	additional : investment : of $331 million for additional spare part	increase[2]	readiness : of Apache Blackhawk helicopter
01/01/2001	FY 2002 : decrease : of $28.2 million	reflect[2]	reduced depot maintenance : requirement : for UH-60 helicopter for UH-1 helicopter retirement pending
02/02/2004	8 new Black Hawk $124.8 : aircraft	upgrade : for 1st	5 : Black Hawk : to UH-60M $78.3 model
06/11/1996	$75,000,000 : for Blackhawk	advance	Rotary Wing Aircraft : blackhawk procurement

In contrast, the unit cost of a P-3 manned aircraft used by U.S. Immigration and Customs Enforcement is $36 million. **Blackhawk helicopters which are frequently used on the borders cost $8.6 million per unit**. However, the benefit of the Blackhawk's relative low unit cost is diminished by its lack of endurance. Blackhawks have a maximum endurance of 2 hours and 18 minutes.16 Consequently, UAVs longer dwell time would allow them to patrol the border longer. The range of UAVs is a significant asset when compared to border agents on patrol or stationery surveillance equipment. If an illegal border entrant attempts to transit through dense woods or mountainous terrain, UAVs would have a greater chance of

Queries/day vs day of operation

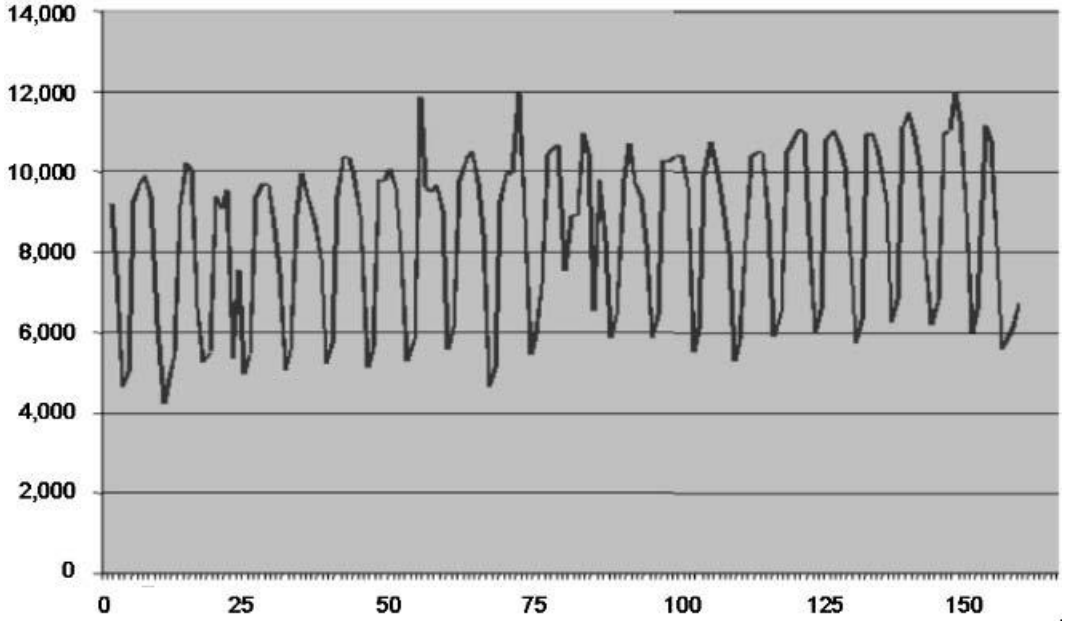


Fig. 5.11. Queries/day vs day of operation (June 22, 2005, to November 30, 2005).

Queries/week vs week of operation

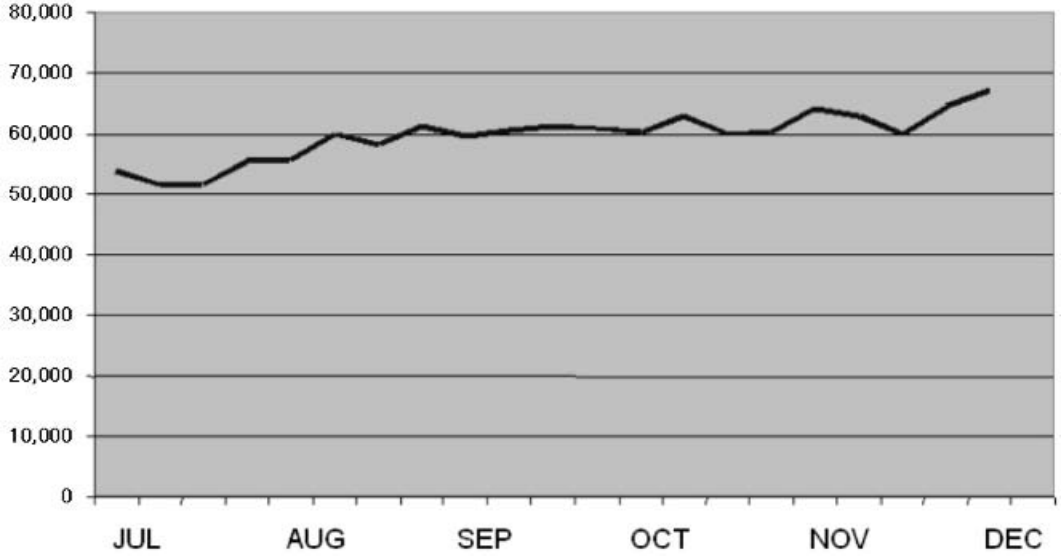


Fig. 5.12. Queries/week vs week of operation (June to November, 2005).

Percentage of tips clicked

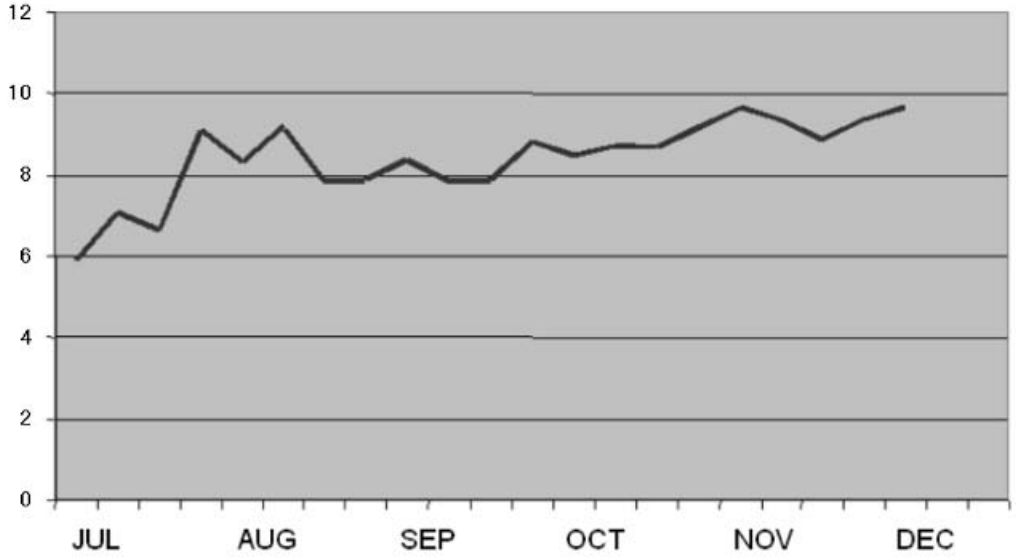


Fig. 5.13. Percentage of tips clicked (June to November, 2005).

Conclusion

- We deployed a natural language based search to a community of Web users, and measured its popularity relative to conventional keyword search.
- It addressed criticisms of NLP approaches to search to the effect that they are not scalable and are too complex to be usable by average end-users.
- Approach rests on a sophisticated index parameterization of text content, that captures syntactic and semantic roles, in addition to keyword counts, and enables interactive search and retrieval of events patterns based on a combination of keyword distributions and natural language attributes.
- Distributed indexing and search services are designed to scale to large document collections and large numbers of users. It was successfully deployed on a Web site that serves a community of 100,000 users.
- An analysis of query logs shows that, during the first six months of operation, traffic has increased by almost 40%. Even more significantly, we are encountering some success in promoting natural language searches.
- The study demonstrates that the percentage of users that avail themselves of guided fact navigation based on natural language understanding has increased from 4% to 10% during the first six months of operation.

References

1. D. Appelt and D. Israel. Introduction to information extraction technology. IJCAI-99 tutorial. http://www.ai.sri.com/~appelt/ie-tutorial/ijcai99.pdf.
2. D. Appelt and D. Israel. Semantic approaches to binding theory. In *Proceedings of the Workshop on Semantic Approaches to Binding Theory. ESSLLI*, 2003.
3. A. Arampatzis, T. van der Weide, P. van Bommel, and C. Koster. Linguistically-motivated information retrieval. In M. Dekker, editor, *Encyclopedia of Library and Information Science*, Springer Verlag, volume 69, pages 201–222. 2000.
4. C. F. Baker, C. J. Fillmore, and J. B. Lowe. The Berkeley FrameNet project. In C. Boitet and P. Whitelock, editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, pages 86–90, San Francisco, California, 1998. Morgan Kaufmann Publishers.
5. I. Dagan, O. Glickman, and B. Magnini. The pascal recognizing textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop Recognizing Textual Entailment*, 2005.
6. M. Dimitrov. A light-weight approach to coreference resolution for named entities in text. Master's thesis, University of Sofia, 2002.

Contd..

7. D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288, 2002.
8. D. Gildea and M. Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 239–246, 2002.
9. GlobalSecurity.org. http://www.globalsecurity.org/org/overview/history.htm.
10. M. Kameyama. Recognizing referential links: An information extraction perspective. In *Proceedings of the ACL'97/EACL'97 Workshop on Operation Factors in Practical, Robust Anaphora Resolution*, pages 46–53, 1997.
11. A. Kao and S. Poteet. Report on KDD conference 2004 panel discussion can natural language processing help text mining? *SIGKDD Explorations*, 6(2):132–133, 2004.
12. C. Kennedy and B. Boguraev. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)*, pages 113–118, 1996.
13. S. Lappin and H. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.
14. D. Lin and P. Pantel. DIRT - discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, pages 323–328, 2001.
15. C. Manning and H. Schutze. *Foundation of Statistical Natural Language Processing*. The MIT Press, 2000.
16. R. Miltov. Robust pronoun resolution with limited knowledge. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'98)/ACL'98*, pages 869–875.

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Evaluating Self-Explanations in iSTART: Word Matching, Latent Semantic Analysis, and Topic Models

Chutima Boonthum, Irwin B. Levinstein, and Danielle S. McNamara

Introduction

- iSTART (Interactive Strategy Trainer for Active Reading and Thinking) is a webbased, automated tutor designed to help students become better readers via multimedia technologies.
- It provides young adolescent to college-aged students with a program of self-explanation and reading strategy training [19] called Self-Explanation Reading Training, or SERT.
- The reading strategies include :

(a) comprehension monitoring, being aware of one's understanding of the text;
(b) paraphrasing, or restating the text in different words;
(c) elaboration, using prior knowledge or experiences to understand the text (i.e., domain-specific knowledge-based inferences) or common sense, using logic to understand the text (i.e., domain-general knowledge based inferences);
(d) predictions, predicting what the text will say next; and
(e) bridging, understanding the relation between separate sentences of the text.

Contd..

- The computational challenge here is to provide appropriate feedback to the students concerning their self-explanations.
- To do so requires capturing some sense of both the meaning and quality of the self-explanation.
- Interpreting text is critical for intelligent tutoring systems, such as iSTART, that are designed to interact meaningfully with, and adapt to, the users' input. iSTART was initially proposed as using
- Latent Semantic Analysis (LSA; [13]) to capture the meanings of texts and to assess the students' self-explanation; however, while the LSA algorithms were being built, iSTART used simple word matching algorithms.
- In the course of integrating the LSA algorithms, we found that a combination of word-matching and LSA provided better results than either separately

Contd..

- Our algorithms are constrained by two major requirements, speedy response times and speedy introduction of new texts.
- Since the trainer operates in real time, the server that calculates the evaluation must respond in 4 to 5 seconds.
- The algorithms must not require any significant preparation of new texts.
- In order to accommodate the needs of the teachers whose classes use iSTART, the trainer must be able to use texts that the teachers wish their students to use for practice within a day or two.

iSTART: Feedback Systems

- iSTART was intended from the outset to employ LSA to determine appropriate feedback.
- The initial goal was to develop one or more benchmarks for each of the SERT strategies relative to each of the sentences in the practice texts and to use LSA to measure the similarity of a trainee's explanation to each of the benchmarks.
- A benchmark is simply a collection of words, in this case, words chosen to represent each of the strategies.
- However, while work toward this goal was progressing, we also developed a preliminary "word-based" (WB) system to provide feedback in our first version of iSTART so that we could provide a complete curriculum for use in experimental situations.
- The second version of iSTART has integrated both LSA and WB in the evaluation process; however, the system still provides only overall quality feedback.

iSTART: Feedback Systems

- Word matching is a very simple and intuitive way to estimate the nature of a self explanation.
- In the first version of iSTART, several hand-coded components were built for each practice text. For example, for each sentence in the text, the "important words" were identified by a human expert and a length criterion for the explanation was manually estimated.
- For each important word, an association list of synonyms and related terms was created by examining dictionaries and existing protocols as well as by human judgments of what words were likely to occur in a self-explanation of the sentence.
- In the sentence "All thunderstorms have a similar life history," for example, important words are *thunderstorm, similar, life*, and *history*. An association list for *thunderstorm* would include *storms, moisture, lightning, thunder, cold, tstorm, t-storm, rain, temperature, rainstorms*, and *electric-storm*.

Types of Matching

- **Literal word matching** - Words are compared character by character and if there is a match of the first 75% of the characters in a word in the target sentence (or its association list) then we call this a literal match.
- This also includes removing suffix -s, -d, -ed, -ing, and -ion at the end of each words.
- For example, if the trainee's self-explanation contains 'thunderstom' (even with the misspelling), it still counts as a literal match with words in the target sentence since the first nine characters are exactly the same.
- On the other hand, if it contains 'thunder,' it will not get a match with the target sentence, but rather with a word on the association list.

Contd..

- **Soundex matching** - This algorithm compensates for misspellings by mapping similar characters to the same soundex symbol [1, 5].
- Words are transformed to their soundex code by retaining the first character, dropping the vowels, and then converting other characters into soundex symbols.
- If the same symbol occurs more than once consecutively, only one occurrence is retained.
- For example, ‘thunderstorm’ will be transformed to ‘t8693698’; ‘communication’ to ‘c8368.’

Three criteria

- **Length** is assessed by a ratio of the number of words in the explanation to the number in the target sentence, taking into consideration the length criterion.
- ***Relevance*** is assessed from the number of matches to important words in the sentence and words in the association lists.
- ***Similarity*** is assessed in terms of a ratio of the sentence and explanation lengths and the number of matching important words.

Factors for Feedback

1) The number of words in the explanation that match either the important words or association-list words of the target sentence compared to the number of important words in the sentence and
2) The length of the explanation in comparison with the length of the target sentence.

This algorithm will be referred as *WB-ASSO*, which stands for *word-based with association list*.

iStart Versions

- The first version of iSTART (word-based system) required a great deal of human effort per text, because of the need to identify important words and, especially, to create an association list for each important word.
- Instead of lists of important and associated words we simply used content words (nouns, verbs, adjectives, adverbs) taken literally from the sentence and the entire text.
- This algorithm is referred to as *WB-TT*, which stands for *word-based with total text*. The content words were identified using algorithms from Coh-Metrix, an automated tool that yields various measures of cohesion, readability, other characteristics of language.
- The iSTART system then compares the words in the self-explanation to the content words from the current sentence, prior sentences, and subsequent sentences in the target text, and does a word-based match (both literal and soundex) to determine the number of content words in the self-explanation from each source in the text.

Latent Semantic Analysis (LSA) Feedback Systems

- Latent Semantic Analysis uses statistical computations to extract and represent the meaning of words.
- Meanings are represented in terms of their similarity to other words in a large corpus of documents.
- LSA begins by finding the frequency of terms used and the number of co-occurrences in each document throughout the corpus and then uses a powerful mathematical transformation to find deeper meanings and relations among words.

Contd..

- To construct an LSA corpus matrix, a collection of documents are selected.
- A document may be a sentence, a paragraph, or larger unit of text. A term-document frequency (TDF) matrix X is created for those terms that appear in two or more documents.
- The row entities correspond to the words or terms (hence the *W*) and the column entities correspond to the documents (hence the *D*).
- The matrix is then analyzed using Singular Value Decomposition, that is the TDF matrix X is decomposed into the product of three other matrices:

(1) vectors of derived orthogonal factor values of the original row entities W,

(2) vectors of derived orthogonal factor values of the original column entities D, and

(3) scaling values (which is a diagonal matrix) S.

$$\{X\} = \{W\}\{S\}\{D\}$$

Computation

- In iSTART, the documents are sentences from texts and trainees' explanations of those sentences.
- These documents consist of terms, which are represented by term vectors;
- hence, the document can be represented as a document vector which is computed as the sum of the term vectors of its terms:

$$D_i = \sum_{t=1}^{n} T_{ti}$$

$$Sim(D1, D2) = \frac{\sum_{i=1}^{d}(D1_i \times D2_i)}{\sum_{i=1}^{d}(D1_i)^2 \times \sum_{i=1}^{d}(D2_i)^2}$$

Benchmarks

1) the words in the title of the passage (“title”),

2) the words in the sentence (“current sentence”),

3) words that appear in prior sentences in the text that are causally related to the sentence (“prior text”), and

4) words that did not appear in the text but were used by two or more subjects who explained the sentence during experiments

LSA-based factors

1) the words in the title of the passage,

2) the words in the sentence, and

3) the words in the two immediately prior sentences.

From the word-based values we include

4) the number of content words matched in the target sentence,

5) the number of content words matched in the prior sentences,

6) the number of content words matched in the subsequent sentences, and

7) the number of content words that were not matched in 4, 5, or 6.

Topic Models (TM) Feedback System

- The Topic Models approach (TM; [10, 27]) applies a probabilistic model in finding a relationship between terms and documents in terms of topics.
- A document is conceived of as having been generated probabilistically from a number of topics and each topic consists of number of terms, each given a probability of selection if that topic is used.
- By using a TM matrix, we can estimate the probability that a certain topic was used in the creation of a given document. If two documents are similar, the estimates of the topics they probably contain should be similar.

$$\{X_{normalized}\} = \{W\}\{D\}$$

Results from Topic Models Toolbox

TOPIC 2 0.0201963151	TOPIC 38 0.0214418635
earth 0.1373291184	light 0.1238061875
sun 0.0883152826	red 0.0339683946
solar 0.0454833721	color 0.0307797075
atmosphere 0.0418036547	white 0.0262046347
moon 0.0362104843	green 0.0230159476
surface 0.0181062747	radiation 0.0230159476
planet 0.0166343877	wavelengths 0.0230159476
center 0.0148681234	blue 0.0184408748
bodies 0.0147209347	dark 0.0178863206
tides 0.0139849912	visible 0.0170544891
planets 0.0133962364	spectrum 0.0151135492
gravitational 0.0125131042	absorbed 0.0149749106
system 0.0111884060	colors 0.0148362720
appear 0.0110412173	rays 0.0116475849
mass 0.0100108964	eyes 0.0108157535
core 0.0083918207	yellow 0.0105384764
space 0.0083918207	absorption 0.0102611992
times 0.0079502547	eye 0.0095680064
orbit 0.0073614999	pigment 0.0092907293
...	...

Kullback Liebler Distance

- To measure the similarity between documents based on TM, the Kullback Liebler distance between two documents is recommended, rather than the cosine.
- A document can be represented by a set of probabilities that this document could contain topic i using the following:

$$D_t = \sum_{i=1}^{n} T_{it}$$

$$KL(D1, D2) = \frac{1}{2}\sum_{t=1}^{T} D1_t log_2(D1_t/D2_t) + \frac{1}{2}\sum_{t=1}^{T} D2_t log_2(D2_t/D1_t)$$

Metacognitive Statements

- A metacognitive filter that searches the trainees' selfexplanations for patterns indicating a description of the trainee's mental state such as "now I see …" or "I don't understand this at all."
- While the main purpose of the filter is to enable the system to respond to such non-explanatory content more appropriately, we also used the same filter to remove "noise" such as "What this sentence is saying is …" from the explanation before further processing.

Regular Expressions

```
NOTUNDERSTAND :i(?:.?m|\W+am)(?:\W+\w+)?\W+\W+(?:(?:not
               (?:\W+\w+)?\W+(?:sure|certain|clear))|
               un(?:sure|certain|clear))
UNDERSTAND :now\W+i\W+(?:know|knew|underst(?:an|oo)d|
            remember(?:ed)?|recall(?:ed)?|recogniz(?:ed)?|get|
            got|see)
CONF :(?:so\W+)?i\W+(?:was|got\W+it)\W+(?:right|correct)
```

Experiment 1

Thunderstorm Text	WB-ASSO	WB-TT	WB2-TT	LSA1	LSA2	LSA2/WB2-TT	TM1	TM2
Correlation	0.47	0.52	0.43	0.60	0.61	0.64	0.56	0.58
% Agreement	48%	50%	27%	55%	57%	62%	59%	60%
d' of 0's	2.21	2.26	0.97	2.13	2.19	2.21	1.49	2.37
d' of 1's	0.84	0.79	0.66	1.32	1.44	1.45	1.27	1.39
d' of 2's	0.23	0.36	-0.43	0.47	0.59	0.85	0.74	0.70
d' of 3's	1.38	1.52	1.41	1.46	1.48	1.65	1.51	1.41
Avg d'	1.17	1.23	0.65	1.34	1.43	1.54	1.25	1.23

Coal Text	WB-ASSO	WB-TT	WB2-TT	LSA1	LSA2	LSA2/WB2-TT	TM1	TM2
Correlation	0.51	0.47	0.41	0.66	0.67	0.71	0.63	0.61
% Agreement	41%	41%	29%	56%	57%	64%	61%	61%
d' of 0's	4.67	4.73	1.65	2.52	2.99	2.93	2.46	2.05
d' of 1's	1.06	0.89	0.96	1.21	1.29	1.50	1.38	1.52
d' of 2's	0.09	0.13	-0.37	0.45	0.49	0.94	0.74	0.61
d' of 3's	-0.16	1.15	1.28	1.59	1.59	1.79	1.60	1.50
Avg d'	1.42	1.73	0.88	1.44	1.59	1.79	1.54	1.42

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Textual Signatures: Identifying Text-Types Using Latent Semantic Analysis to Measure the Cohesion of Text Structures

Philip M. McCarthy, Stephen W. Briner, Vasile Rus, and Danielle S. McNamara

Introduction

- Just as a sentence is far more than a mere concatenation of words, a text is far more than a mere concatenation of sentences.
- Texts contain pertinent information that co-refers across sentences and paragraphs; texts contain relations between phrases, clauses, and sentences that are often causally linked and texts that depend on relating a series of chronological events contain temporal features that help the reader to build a coherent representation of the text.
- We refer to textual features such as these as cohesive elements, and they occur within paragraphs (locally), across paragraphs (globally), and in forms such as referential, causal, temporal, and structural.
- But cohesive elements, and by consequence cohesion, does not simply feature in a text as dialogues tend to feature in narratives, or as cartoons tend to feature in newspapers.

Cohesion

- Cohesion is the degree to which ideas in the text are explicitly related to each other and facilitate a unified situation model for the reader.
- As McNamara and colleagues have shown, challenging text (such as science) is particularly difficult for low-knowledge students.
- These students are cognitively burdened when they are forced to make inferences across texts.
- Adding cohesion to text alleviates this burden by filling conceptual and structural gaps.
- Recent developments in computational linguistics and discourse processing have now made it possible to measure this textual cohesion.

Coh-Metrix

- Coh-Metrix assesses characteristics of texts by using parts of speech classifiers, and latent semantic analysis.
- The indices generated from Coh-Metrix offer an assessment of the cohesiveness and readability of any given text.
- These indices have been used to indicate textual cohesion and difficulty levels in a variety of studies.
- For example, Ozuru et al. used Coh-Metrix to rate high and low cohesion versions of biology texts, the study showing that participants benefited most from the high cohesion versions.

Survey

- Best, Floyd, and McNamara [1] used Coh-Metrix to compare 61 third-graders' reading comprehension for narrative and expository texts, the study suggesting that children with low levels of world knowledge were more inclined to have comprehension problems with expository texts.
- Louwerse et al. [37] used Coh-Metrix to investigate variations in cohesion across written and spoken texts, finding evidence for a significant difference between these modes.

Contd..

- Mc-Carthy, Lewis, et al. [42] showed that Coh-Metrix indices were versatile enough to distinguish between authors even within the same register.
- Crossley et al.[12] used a wide variety of Coh-Metrix indices to show significant differences between authentic English language texts and the simplified versions used in texts designed for English language learners.
- And McCarthy, Lightman et al. [41] used Coh-Metrix to investigate variations in cohesion and difficulty across units of science and history texts, finding evidence that while difficulty scores for textbooks reflect the grade to which they are assigned, the cohesion rates differed significantly depending upon domain.

Approaches to Analyzing Texts

- Traditional approaches to categorizing discourse have tended to treat text as if it were a homogeneous whole.
- These wholes, or bodies of text, are analyzed for various textual features, which are used to classify the texts as belonging to one category or another.
- For example, Biber [2] identified *if clauses* and singular person pronoun use as key predictors in distinguishing British- from American-English.

Contd..

- The *parts* of a text serve the textual whole either by function or by form.
- In terms of *function*, Propp [48] identified that texts can be comprised of fundamental components, fulfilled by various characters, performing set functions. In terms of *form*, numerous theories of text structure have demonstrated how textual elements are inter-related.
- Labov's narrative theory, to take one example, featured six key components: the abstract (a summary), the orientation (the cast of characters, the scene, and the setting), the action (the problem, issue, or action), the evaluation (the story's significance), the resolution (what happens, the denouement), and the coda (tying up lose ends, moving to the present time and situation).

Contd..

- Unfortunately for text researchers, the identification of the kinds of discourse markers described above has proven problematic because the absence or ambiguity of such textual markers tends to lead to limited success [39].
- Morris and Hirst [46], for example, developed an algorithm that attempted to uncover a hierarchical structure of discourse based on lexical chains. Although their algorithm was only manually tested, the evidence from their study, suggesting that text is structurally identifiable through themes marked by chains of similar words, supports the view that the elements of heterogeneous texts are identifiable.
- Hearst [23] developed this idea further by attempting to segment expository texts into topically related parts. Like Morris and Hirst [46], Hearst used term repetition as an indicator of topically related parts. The output of his method is a linear succession of topics, with topics able to extend over more than one paragraph. Hearst's algorithm is fully implementable and was also tested on magazine articles and against human judgments with reported precision and recall measures in the 60th percentile, meaning around 60% of topic boundaries identified in the text are correct (precision) and 60% of the true boundaries are identified (recall).

Contd..

- The limited success in identifying textual segments may be the result of searching for a reliable fine grained analysis before a courser grain has first been established.
- For example, a courser approach acknowledges that texts have easily identifiable *beginnings*, *middles*, and *ends*, and these *parts* of a text , or at least a sample from them, are not at all difficult to locate. Indeed, textual analysis using such parts has proved quite productive.
- For example, Burrows [6] found that the introduction section of texts rather than texts as a whole allowed certain authorship to be significantly distinguished.
- And McCarthy, Lightman et al.[41] divided high-school science and history textbook chapters into sections of beginnings, middles, and ends, finding that reading difficulty scores rose with significant regularity across these sections as a chapter progressed.

Textual Signature

- Such a signature stands to serve students and researchers alike.
- For students, their work can be analyzed to see the extent to which their paper reflects a prototypical model.
- Specifically, a parts analysis may help students to see that sections of their papers are under- or over-represented in terms of the global cohesion.
- For researchers, a text-type signature should help significantly in mining for appropriate texts.
- For example, the first ten web sites from a Google search for a text about cohesion (featuring the combined keywords of *comprehension*, *cohesion*, *coherence*, and *referential*) yielded papers from the field of composition theory, English as a foreign language, and cognitive science, not to mention a disparate array of far less academic sources.

Latent Semantic Analysis

- To assess the inter-relatedness of text sections we used latent semantic analysis.
- LSA is a technique that uses a large corpus of texts together with singular value decomposition to derive a representation of world knowledge.
- LSA is based on the idea that any word (or group of words) appears in some contexts but not in others. Thus, words can be compared by the aggregate of their co-occurrences.
- This aggregate serves to determine the degree of similarity between such words.
- LSA does not only tell us whether two items are the same, it tells us how similar they are.

Predictions

- The *abstract* section was selected as the primary source of comparison as it is the only section whose function is specifically to relate the key elements of each other section of the paper.
- But the abstract does not relate to each other section of the paper equally. Instead, the abstract outlines the theme of the study (introduction); it can refer to the basic method used in the study (methods); it will briefly state a prominent result from the study (results); and it will then discuss the relevance of the studys findings (discussions).
- This definition allowed us to make predictions as to the signature generated from such comparisons. Specifically, we predicted that *abstracts* would feature far greater reference to the *introduction* (AI comparison type) and *discussion* sections (AD comparison type), less reference to the results section (AR comparison type), and less reference still to the methods section (AM comparison type).

Methods

- Using freely available on-line psychology papers from five different journals, we formed a corpus of 100 texts.
- For the purposes of simplification and consistency, we extracted from this corpus only the texts that were comprised of five author-identified sections: *abstract*, *Introduction*, *methods*, *results*, *discussion*.
- This left 67 papers in the analysis. We then removed titles, tables, figures, and footnotes before forming the paired sections outlined above (AI, AM, AR, AD). Each of the pairs from the 67 papers was then processed through the Coh-Metrix version of LSA.

Results of Experiment 1

Table 7.1. Pairwise Comparisons of the Relatedness of Text Sections to the Abstract

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.198 (.021)*	Diff=.106 (.021)*	Diff=.001(.010)
Method (AM)		Diff=-.092 (.019)*	Diff=-.197(.019)*
Results (AR)			Diff=-.105 (.018)*

Textual signature formed from means of the abstract to other sections

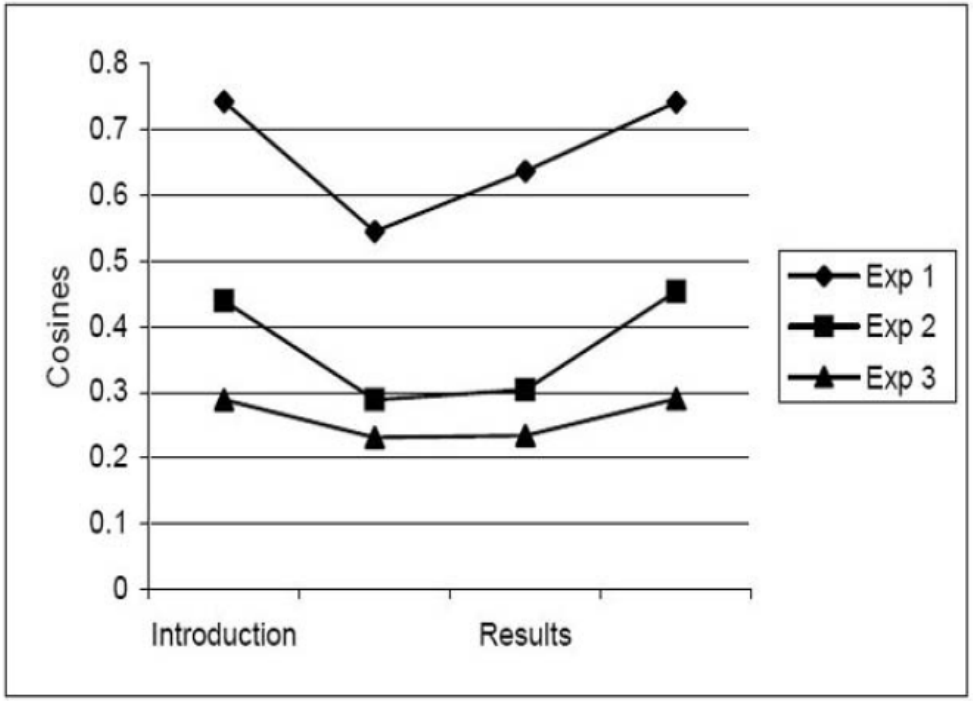


Table 7.2. Pairwise Comparisons of the Relatedness of Text Sections to the Abstract for text length

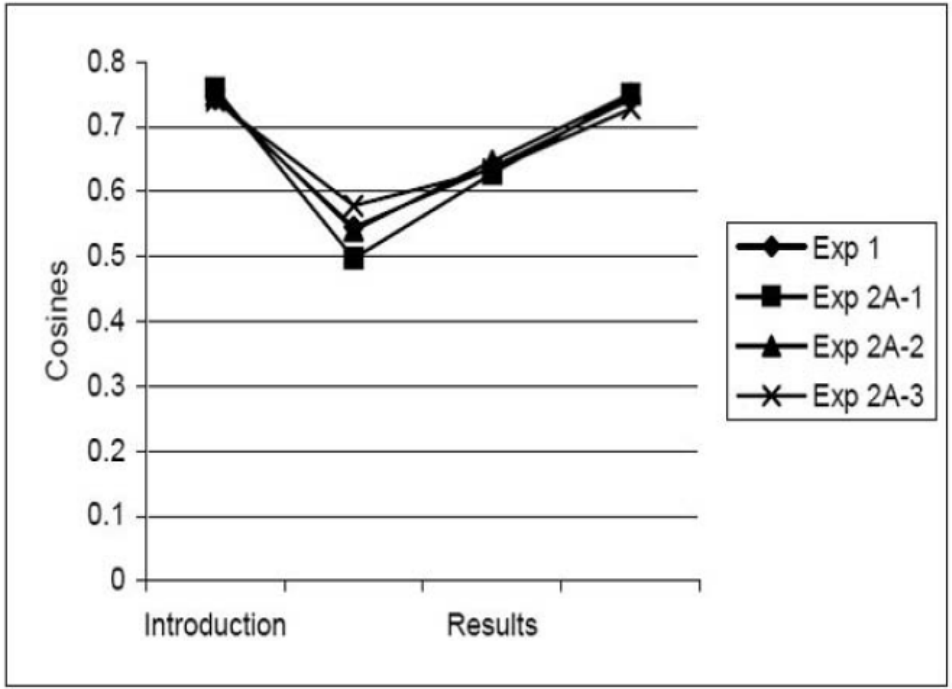
	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=302.22 (113.13)*	Diff=189.39 (107.91)	Diff=236.73 (94.37)*
Method (AM)		Diff=-112.84 (120.59)	Diff=-65.49 (105.67)
Results (AR)			Diff=47.34 (97.41)

Experiment 2

Table 7.3. Pairwise comparison of abstract to similar-themed body

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.150 (.032)*	Diff=.135 (.032)*	Diff=-.015 (.020)
Method (AM)		Diff=-.015 (.026)	Diff=-.165 (.036)*
Results (AR)			Diff=.150 (.032)*

Experiment 2a



Experiment 3

Table 7.4. Pairwise comparison of abstract to dissimilarly themed body

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.058 (.013)*	Diff=.055 (.014)*	Diff=-.002 (.013)
Method (AM)		Diff=-.003 (.015)	Diff=-.060 (.019)*
Results (AR)			Diff=-.056 (.017)*

Discussion

- The results of this study suggest that LSA comparisons of textual sections can produce an identifiable textual signature.
- These signatures serve as a prototypical model of the text-type and are distinguishable from those produced by texts which are merely similar in theme (ST), or similar in field (DT).

Contd..

- Textual signatures of the type produced in this study have the potential to be used for a number of purposes.
- For example, students could assess how closely the signature of their papers reflected a prototypical signature.
- The discrepancies between the two LSA cosines may indicate to the student where information is lacking, redundant, or irrelevant.
- For researchers looking for supplemental material, the signatures method could be useful for identifying texts from the same field, texts of the same theme, and even the part of the text in which the researcher is interested.
- Related to this issue is a key element in Question Answering systems: as textual signatures stand to identify thematically related material, the retrieval stage of QA systems may be better able to rank its candidate answers.

References

1. Best, R.M., Floyd, R.G., & McNamra, D.S. (2004). Understanding the fourth-grade slump: Comprehension difficulties as a function of reader aptitudes and text genre. Paper presented at the 85th Annual Meeting of the American Educational Research Association.
2. Biber, D. (1987). A textual comparison of British and American writing. American Speech, 62, 99-119.
3. Biber, D. (1988). Linguistic features: algorithms and functions in variation across speech and writing. Cambridge: Cambridge University Press.
4. Brill, E. (1995). Unsupervised learning of disambiguation rules for part of speech tagging. In Proceedings of the Third Workshop on Very Large Corpora, Cambridge, MA.
5. Britton, B. K., & Gulgoz, S. (1991). Using Kintschs computational model to improve instructional text: Effects of inference calls on recall and cognitive structures. Journal of Educational Psychology, 83, 329-345
6. Burrows, J. (1987). Word-patterns and story-shapes: The statistical analysis of narrative style. Literary and Linguistic Computing, 2, 6170.
7. Charniak, E. (1997) Statistical Parsing with a context-free grammar and word statistics Proceedings of the Fourteenth National Conference on Artificial Intelligence, Menlo Park: AAAI/MIT Press

Contd..

8. Charniak, E. (2000) A Maximum-Entropy-Inspired Parser. Proceedings of the North-American Chapter of Association for Computational Linguistics, Seattle, WA
9. Charniak, E. & Johnson, M. (2005) Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (pp. 173-180). Ann Arbor, MI
10. Collins, M. (1996) A New Statistical Parser Based on Bigram Lexical Dependencies. Proceedings of the 34th Annual Meeting of the ACL, Santa Cruz, CA
11. Collins, M. (1997) Three Generative, Lexicalised Models for Statistical Parsing Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid, Spain.
12. Crossley, S., Louwerse, M.M., McCarthy, P.M., & McNamara, D.S. (forthcoming 2007). A linguistic analysis of simplified and authentic texts. Modern Language Journal, 91, (2).
13. Dennis, S., Landauer, T., Kintsch, W. & Quesada, J. (2003). Introduction to Latent Semantic Analysis. Slides from the tutorial given at the 25th Annual Meeting of the Cognitive Science Society, Boston.
14. Duran, N., McCarthy, P.M., Graesser, A.C., McNamara, D.S., (2006). An empirical study of temporal indices. Proceedings of the 28th annual conference of the Cognitive Science Society, 2006.
15. Foltz, P. W., Britt, M. A., & Perfetti, C. A. (1996). Reasoning from multiple texts: An automatic analysis of readers' situation models. In G. W. Cottrell (Ed.) Proceedings of the 18th Annual Cognitive Science Conference (pp. 110-115). Lawrence Erlbaum, NJ.

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

Automatic Document Separation: A Combination of Probabilistic Classification and Finite-State Sequence Modeling

Mauritius A. R. Schmidtler, and Jan W. Amtrup

Introduction

- Large organizations are increasingly confronted with the problem of capturing, processing, and archiving large amounts of data.
- For several reasons, the problem is especially cumbersome in the case where data is stored on paper.
- First, the weight, volume, and relative fragility of paper incur problems in handling and require specific, labor-intensive processes to be applied.
- Second, for automatic processing, the information contained on the pages must be digitized, performing Optical Character Recognition (OCR). This leads to a certain number of errors in the data retrieved from paper.
- Third, the identities of individual documents become blurred. In a stack of paper, the boundaries between documents are lost, or at least obscured to a large degree.

Contd..

- As an example, consider the processing of loan documents in the mortgage industry:
- Usually, documents originate at local branch offices of an organization (e.g., bank branches, when a customer fills out and signs the necessary forms and provides additional information).
- All loan documents finalized at a local office on a given day are collated into one stack of paper (called a *batch*) and sent via surface mail to a centralized processing facility. At that facility, the arriving packets from all over the country are opened and the batches are scanned. In order to define the boundaries and identities of documents, *separator sheets* are manually inserted in the batches.
- Separator sheets are special pages that carry a barcode identifying the specific loan document that follows the sheet, e.g., Final Loan Application or Tax Form, etc. The separation and identification of the documents is necessary for archival and future retrieval of specific documents. It is also a precondition for further processing, for instance in order to facilitate the extraction of certain key information, e.g., the loan number, property address and the like.

Contd..

- The problem we are addressing in this chapter is the process of manually inserting separator sheets into loan files.
- A person must take a loan file, leaf through the stack of paper (hundreds of pages), and insert appropriate separators at the correct boundary points.
- This work is both tedious and challenging. It is tedious, since no important new information is created, but only information that previously existed is re-created.
- It is challenging, since the person needs to have a fair amount of knowledge of loan documents (hundreds of document categories) and work with a high degree of attention to detail. Nevertheless, the error rate for this process can be as high as 8%.
- The cost for the insertion is also significant, both in terms of labor and material; it is estimated that 50% of the document preparation cost is used for sorting and the insertion of separator sheets. One customer estimates the printing cost for separator sheets alone to be in excess of $1M per year.

Related Work

- Traditionally, the processing of scanned paper forms has concentrated on the handling of structured forms.
- These are paper documents that have well-defined physical areas in which to insert information, such as the social security number and income information on tax forms.
- Ideally, for these forms the separation problem does not even arise, since the documents are of a specified length.
- If, however, a sequence of documents needs to be separated, it is usually enough to concentrate a recognition process on the first page to find out which document is present.
- This information defines the number of pages in the document and thus the separation information with certainty. The recognition process is often done indirectly, coupled with a subsequent extraction system.

Contd..

- Extraction rules define areas of interest on a form and how to gather data from those *zones*.
- For instance, an extraction rule for a tax form could first specify how to identify a box on the top left corner of the document that contains the text "1040."
- Then the rule would search down the form to find a rectangular box labeled "SSN" and extract the nine digits contained in a grid directly to the right of the label.
- If this recognition rule succeeds, i.e., text can be found and recognized with sufficient confidence, the document is identified as a two-page 1040 tax form and the social security number is extracted.

Contd..

- The most direct approach to document separation would treat the task as a straightforward segmentation problem.
- Maximum Entropy (ME) methods have proven very successful in the area of segmentation of natural language sentences [2, 3].
- Each boundary (in our case the point between two pages) is characterized by features of its environment (e.g., by the words used on the preceding and following page).
- An ME classifier is then used to solve the binary problem (boundary/nonboundary) for new, unseen page transitions.
- We are unaware of any publication using this approach for automatic document separation.

Contd..

- Instead of looking for boundaries, one could also attempt to ascertain that two consecutive pages belong in the same document, thus indirectly establishing borders.
- An instance of this method is described in [4]. They define a similarity measure between two pages that takes document structure (text in headers and footers, esp. page numbers), layout information (font structure), and content (text on pages) into account.
- They use a single-linkage agglomerative clustering method to group pages together.
- The clustering process is bounded by manually set thresholds.
- They report a maximum separation accuracy of 95.68%, using a metric from [5] that measures the correctness of the number of separation points between non-adjacent pages.
- Since our data is different and we solve a combined problem of classification and separation ([4] only perform separation), their results cannot directly be compared to ours.

Data Preparation

- The input to our separation solution is the text delivered by an OCR engine of scanned page images.
- We are primarily reporting on data from the mortgage processing industry, hence the document types (Appraisal, Truth in Lending, etc.).
- Our sample here contains documents from 30 document types.
- The quality of the images varies based on their origin (original or photocopy) and treatment (fax).
- Figures 8.1 and 8.2 show two sample images (one from a Loan Application, one from a Note) and some of the OCR text generated from them.

Contd..

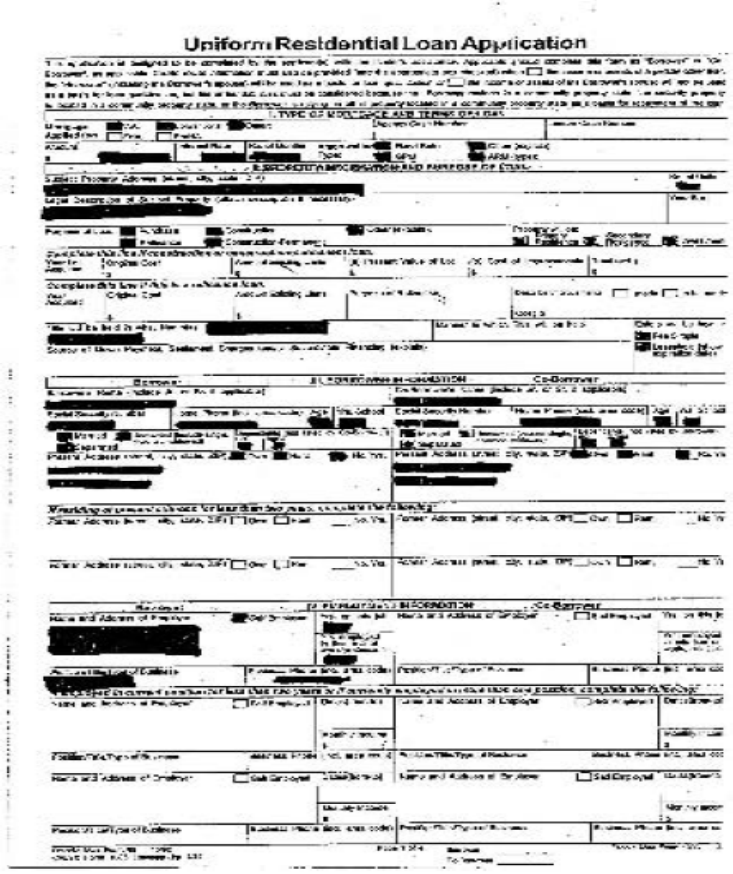
- In order to be prepared for the core classification algorithms (see below), the input text is tokenized and stemmed.
- Tokenization uses a simple regular expression model that also eliminates all special characters.
- Stemming for English is based on the Porter algorithm.
- The stream of stemmed tokens isolated from a scanned image is then converted into a feature vector.
- We are using a *bag of words* model of text representation; each token type is represented by a single feature and the value of that feature is the number of occurences of the token on the page.
- In addition, the text is filtered using a stopword list. This filtering removes words that are very common in a language

Abstractions over the input text

• By stemming, we assume that the detailed morphological description of words is irrelevant for the purpose of classification. For instance, we are unable to tell whether the feature "address" in Table 8.1 came from the input "address," "addresses," or "addressing." Inflectional and part-of-speech information is lost.

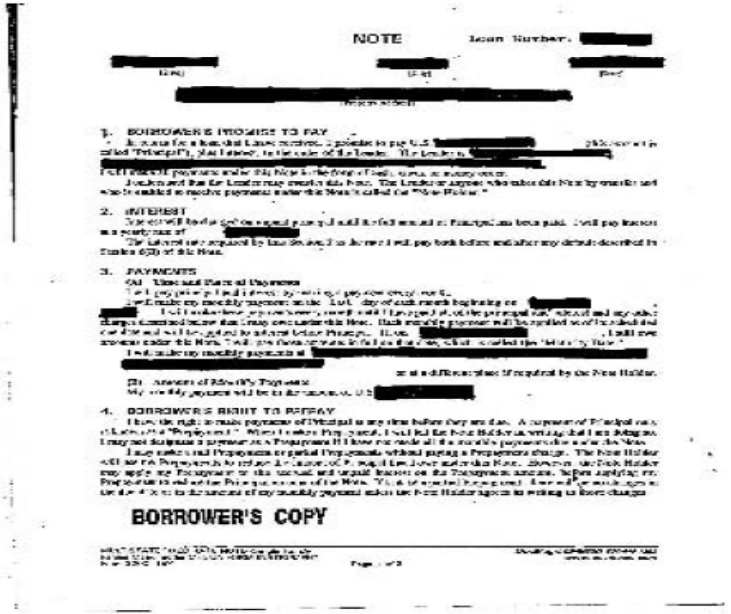
• Using bags of words, we are abstracting from the linear structure of the input text. We pose that there is little value in knowing which word appeared before or near another and the only important information is in knowing which word appears more frequently than others.

• The application of a stopword list, finally, de-emphasizes the value of syntactic information even further, since many syntactically disambiguating words are ignored.



Uniform Residential Loan Application I TYPE GF'MORTGAGE-AHD TERMS OF LOAN Mortgage flvAB—Ccnvniltiai Applied fof: E¿HA Agency Case Number Lender Case Number Amount 5 I No. of Months Amortization Fixed Rate Typo: I Other (explain): I ARM (type): / LOAN Subject Property Addresi (street, city, state, ZIP) Legal Descripllon of Subject Property (attach description If necessary) No. of Units Year Built Purpose of Loan Construction Construction-Permanent Otner (explain): Property will be: Primary ” ”an. Compl&te this line If construction or construction-permanent loan. Secondary Investment Year Lot Acquired Original Cost S Amount Exjsling Uens $ (a) Present Value of Lot $ (b) Cost o(Improvements $ Total (a+b) S Complete this line if this Is a ra/fiuncfl loan. Year Acquired Original Cost Amount Existing Uens Title will be held in what Name(s) Purpose of Refinance Describe ImprovamanU

Image and OCR text from a sample Note



NOTE

BORROWER'S COPY

4. BORROWER'S RIGHT TO PREPAY

I have the right to make payments of Principal at any time before they are due. A payment of Principal only is known as a "Prepayment." When I make a Prepayment, I will tell the Note Holder in writing that I am doing so. I may not designate a payment as a Prepayment if I have not made all the monthly payments due under the Note. I may make a full Prepayment or partial Prepayments without paying a Prepayment charge. The Note Holder will use my Prepayments to reduce the amount of Principal that I owe under this Note. However, the Note Holder may apply my Prepayment to the accrued and unpaid interest on the Prepayment amount, before applying my Prepayment to reduce the Principal amount of the Note. If I make a partial Prepayment, there willpe no changes in the due date or in the amount of my monthly payment unless the Note Holder agrees in writing to those changes.

Some of the features extracted from a Note

Token	#Occur	Token	#Occur	Token	#Occur
accru	1	chang	2	fanni	1
acm	1	charg	3	fix	1
address	1	check	1	form	3
agre	1	citi	1	freddi	1
ani	3	compani	1	ftill	1
anyon	1	date	5	holder	6
appli	4	day	1	home	1
august	1	dbs	1	howev	1
befor	4	default	1	initi	1
begin	1	describ	2	ink	1
borrow	2	design	1	instrument	1
bowi	1	differ	1	interest	9
box	1	entitl	1	juli	1
burtonsvil	1	everi	2	known	1
cash	1	famili	1	la	1

Some of the features related to the stem *borrow*

Token	#Occur	Token	#Occur	Token	#Occur
borrnu	1	borronv	4	borrovv	8
borrnwer	1	borrotr	1	borrovvcr	1
borro	92	borrou	3	borrovvef	1
borroa	1	borrov	14	borrovvei	1
borroaer	1	borrovc	1	borrovvfir	1
borroh	4	borrovcf	1	borrovvi	1
borroi	1	borrovd	1	borrovw	1
borroifril	1	borrovi	3		
borrojv	1	borrovj	3		
borrokbr	1	borrovjar	1		
borrom	2	borrovl	1		
borromad	1	borrovrti	1		
borromicrl	1	borrovt	1		
borromr	1	borrovti	1		
borron	1	borrovu	1		

Document Separation as a Sequence Mapping Problem

- Automatic Document Separation adds two pieces of information to a stream of unlabeled pages.
- It inserts boundaries, so that documents are kept together, and it assigns labels to those documents that indicate their type.
- The problem can be seen as the mapping of an input sequence to an output sequence, i.e., a sequence of scanned paper pages is mapped to a sequence of document types.
- The mapping of sequences is a well known problem in Computer Science and there exist many different applications.
- For example, compilers, speech recognition, information extraction, and machine translation are all instances that have some aspect that deals with the problem of sequence mapping: A sequence of human readable program statements to a sequence of machine code, a sequence of acoustic signals to a sequence of words, a sequence of words to a sequence of tags, and a sequence of, e.g., Spanish words to a sequence of French words.

Sequence Model

Formally, the procedure described above can be modeled as a Markov chain. Denoting the input sequence of ordered pages[6] p^c by $\mathcal{P} = (p_1^c, \ldots, p_n^c)$ and the output sequence of document types by $\mathcal{D} = (d_1, \ldots, d_n)$, the probability of a specific sequence of document types $\mathcal{D}$ given the input sequence of pages can be written as

$$p(\mathcal{D}|\mathcal{P}) = \prod_{j=1}^{n} p(d_j|\mathcal{D}_{j-1}, \mathcal{P}) \,, \tag{8.1}$$

where d_j denotes the document type of the j-th page and $\mathcal{D}_{j-1}$ the output sequence of document types up to the $(j-1)$-th page. In many practical applications, independence assumptions regarding the different events d_j, $\mathcal{D}_{j-1}$, and $\mathcal{P}$ hold at some level of accuracy and allow estimations of the probability $p(\mathcal{D}|\mathcal{P})$ that are efficient yet accurate enough for the given purpose.

Contd..

We started by assuming that the document type d_j at time step j only depends on the page content p_j^c at time step j and gradually increased the complexity of the models by taking into account the document types of previous time steps. In particular, we considered

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|p_j^c) \tag{8.2}$$

as well as the following approximation, which is very common and has been widely used in several fields, e.g., for information extraction [8],

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|d_{j-1}, p_j^c) \tag{8.3}$$

and finally

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|d_{j-1}, d_{j-2}, p_j^c)\,. \tag{8.4}$$

Contd..

Every document type symbol is split into three symbols: *Start*, *middle*, and *end* page of the document type. In our experience, forms often have distinctive first and last pages, e.g., forms ending with signature pages and starting with pages identifying the form, whereas middle pages of forms do not contain as much discriminating information. Accordingly, the sequences of the new output language are now sequences of the type $\mathcal{D}'$, where $\mathcal{D}'$ is given by $\mathcal{D}' = (d'_1, \ldots, d'_n)$ with d'_j denoting the document type as well as the page type. The definitions of the page type events $\{start, middle, end\}$ are:

$$\begin{aligned} start: &\ \{p^c_{j,t} | t = 1, t \leq l\} \\ middle: &\ \{p^c_{j,t} | t > 1, t < l\} \\ end: &\ \{p^c_{j,t} | t > 1, t = l\} \end{aligned} \tag{8.5}$$

where j is the global page number within the batch and t is the local page number within a document of length l.

Sequence Model Estimation

The problem of determining the different sequence models introduced in the previous section is given by estimating a probability of the form $p(x|p^c, y)$ with e.g., x denoting a document type and y a history of document types. As outlined in Section 8.3, a *bag of words* model is used for the page content[7] p^c, i.e., $p^c = \{(c_1, w_1), \ldots, (c_n, w_n)\}$ with c_j denoting the number of occurences of word w_j on the page, yielding

$$p(x|p^c, y) = p(p^c|x, y)\frac{p(x, y)}{p(p^c, y)} \propto \prod_{j=1}^{n} p(w_j|x, y)^{c_j} p(x, y)\,, \tag{8.9}$$

whereby in the last step the constant factor $1/p(p^c, y)$ has been omitted. As can be seen from Eq. 8.9, the sequence model estimation is reduced to the determination of the probabilities $p(w_j|x, y)$ and $p(x, y)$. These probabilities are estimated empirically by using sample documents (training examples) for the various events (x, y). For a

Contd..

- Statistical learning methods, e.g., [10, 11], utilizing methods of regularization theory, allow us to determine the tradeoff between memorization and generalization more principled than the smoothing techniques mentioned above.
- The learning method adopted here for estimating the sequence model is a Support Vector Machine[10] (SVM).
- It is commonly known that Support Vector Machines are well suited for text applications given a small number of training examples [12].
- This is an important aspect for the commercial use of the system, since the process of gathering, preparing, and cleaning up training examples is time consuming and expensive.

Classification

Table 8.3. Classification Results

	Optimized			Not optimized		
	precision	recall	F1-value	precision	recall	F1-value
Micro averages	**0.95**	**0.95**	**0.95**	**0.90**	**0.90**	**0.90**
Macro averages	**0.94**	**0.86**	**0.87**	**0.82**	**0.79**	**0.78**

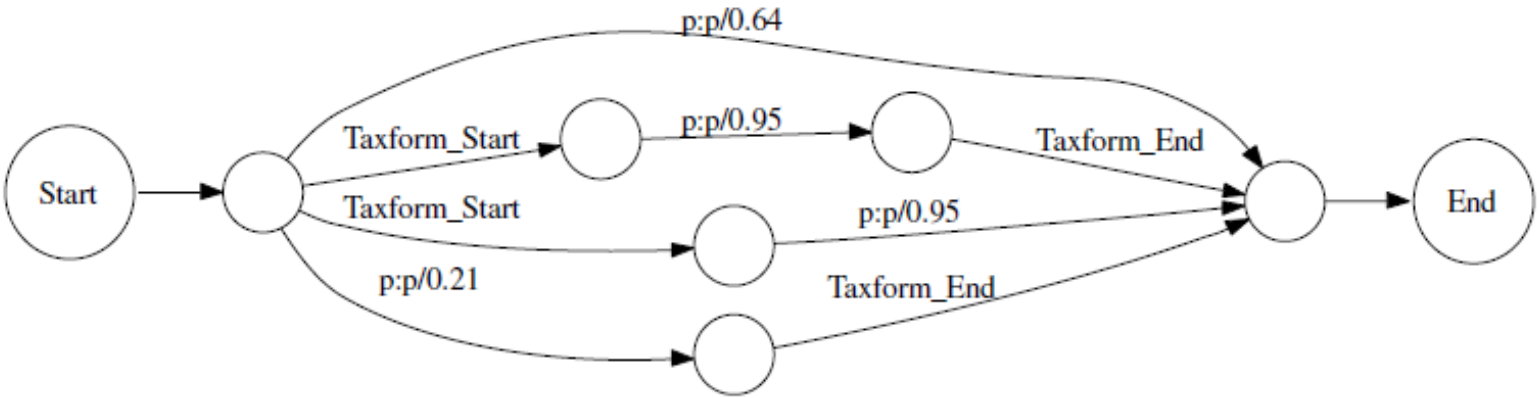
Sequence Processing

- All probability models were based on viewing the classification and separation process as a sequence mapping problem, described formally as a Markov chain as in Eq. 8.1.
- Experience from information extraction and speech recognition (e.g., [15]) shows that the results of such mappings and the search for the best sequence can be represented as a *trellis*.
- A trellis is a two-dimensional graph in which the horizontal axis represents time steps. For speech recognition, this would be incoming acoustic feature vectors.
- For information extraction, it could be words and, in our case, each time step is an incoming page within a batch.
- The vertical axis represents the states in which the mapping process may find itself and also the possible output symbols it may generate.

Partial connection structure for the trellis for model

States			
Previous	Current		
Appraisal	Appraisal	(A,A)	(A,A)
TaxForm	Appraisal	(T,A)	(T,A)
Note	Appraisal		(N,A)
Appraisal	TaxForm	(A,T)	(A,T)
TexForm	TaxForm		(T,T)
Note	TaxForm		(N,T)
Appraisal	Note		(A,N)
TexForm	Note	(T,N)	(T,N)
Note	Note		(N,N)
			time
		page 1	page 2

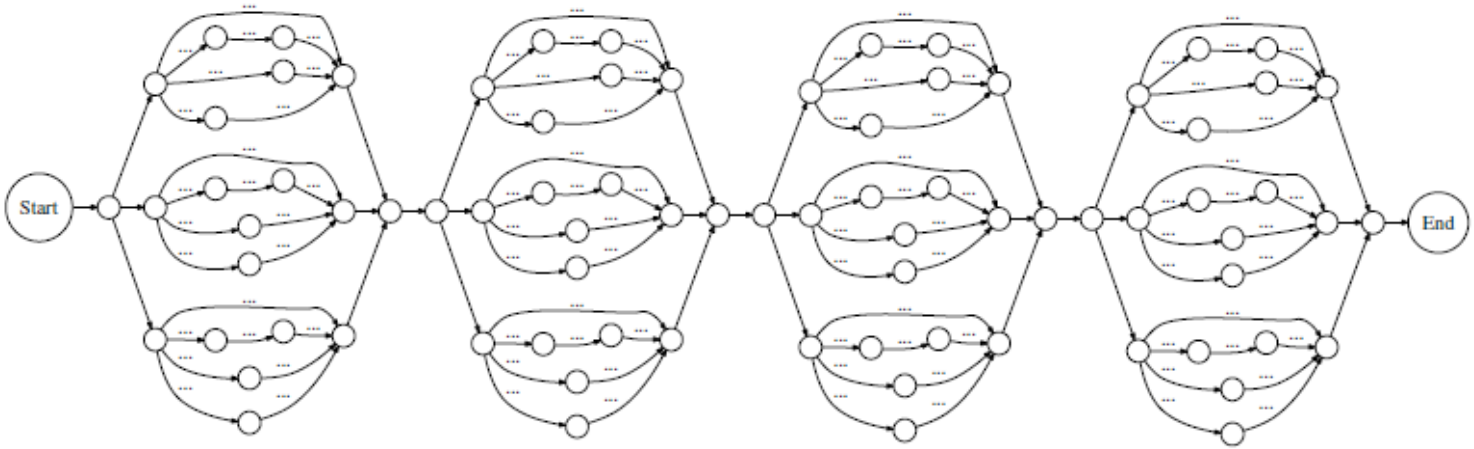
Classification results for one page



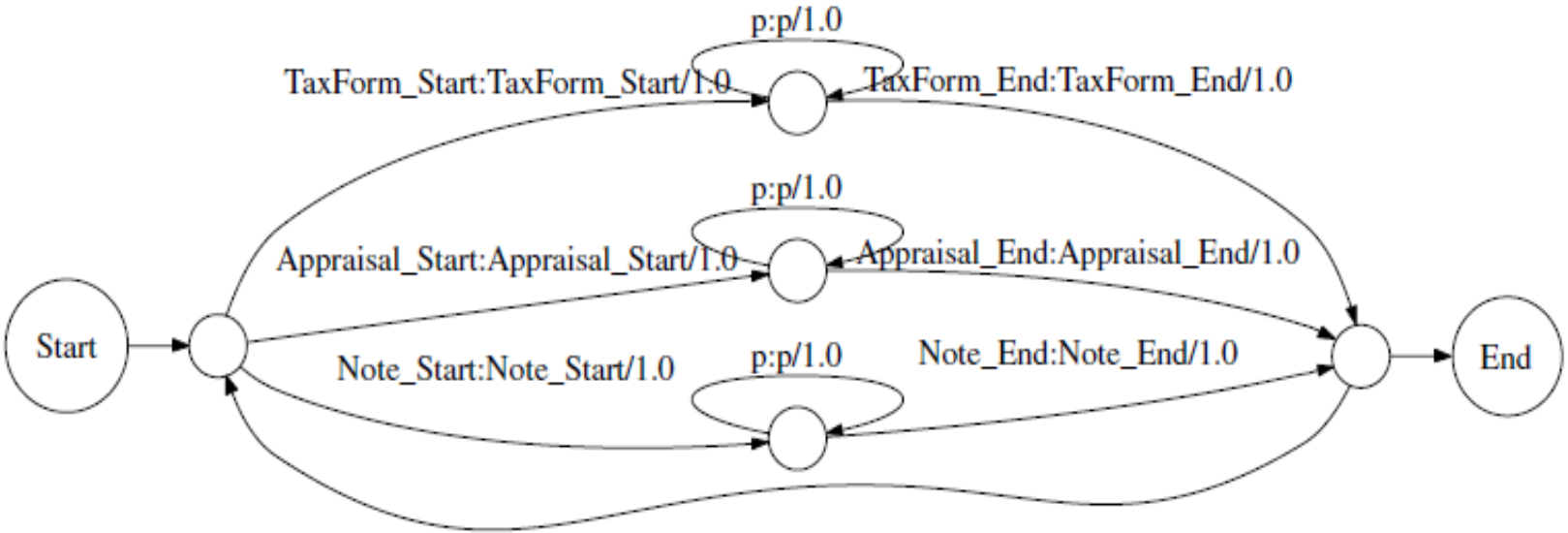
Classification Transducer

- Transitions that represent physical pages contain a symbol indicating a physical page on the lower and upper level and a classification score as weight. Which score is attached to the page depends on the topology of the transducer, which is defined by the next type of transitions.
- Transitions with an empty lower level denote boundary information about documents. There are transitions for the start and the end of a document. The occurrence of these transitions thus defines the type of page and the type of score that should be used. For instance, in Figure 8.5, the topmost transition (with score 0.64) indicates a middle page, since there are no boundaries given. The second transition chain belongs to a form that contains only a single page and consequently is bounded by both a start indicator and an end indicator. The third and fourth transitions belong to start and end pages respectively.

Classification results as an FST



FST for three document types

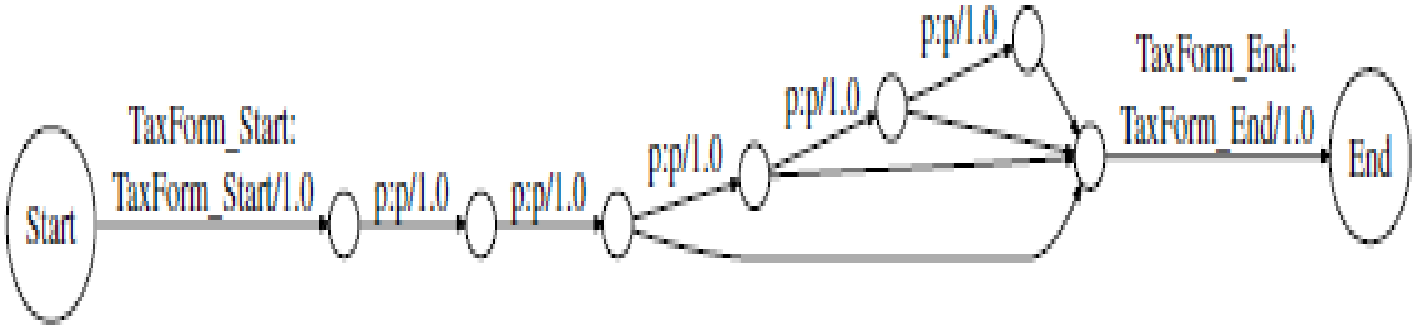


Algorithm 1 General document separation

Require: An ordered list of pages (a batch), consisting of pages p_1 to p_n.
Require: An FST *rules* that describes the possible sequences of documents, as in Figure 8.7.
Ensure: An ordered list of documents d_k, each of which consists of an ordered list of pages.

```
{Perform Classification}
for all pages p_i, 1 ≤ i ≤ n in the batch do
    for all classes c_j do {Three classes per document type (start/middle/end)}
        c_ij ← probability that p_i is of class c_j
    end for
end for
{Perform Separation}
pg ← The FST representing the classification results, as in Figure 8.6
sg ← pg composed with rules
sg ← the best path through sg
{Create documents}
D ← ∅
d ← ∅
for all Transitions in sg, in topological order do
    if The transition is labeled with "_End" then
        D+ = d
    else if The transition is labeled with a page then
        d+ = p
    end if
end for
```

A rule FST


Start
TaxForm_Start:
TaxForm_Start/1.0
p:p/1.0
p:p/1.0
p:p/1.0
p:p/1.0
p:p/1.0
TaxForm_End:
TaxForm_End/1.0
End

Memory Usage

Table 8.4. Memory usage for separation

Batch size in pages	Memory Usage 200 categories	300 categories
1000	118 MB	151 MB
2000	211 MB	363 MB

Results

Table 8.5. Comparison of separation and classifcation results of the various sequence models.

Seqence model	Probability	Micro-averaged F1-value
Eq. 8.2	$p(d_j \mid p_j^c)$	0.63
Eq. 8.3	$p(d_j \mid d_{j-1}, p_j^c)$	0.74
Eq. 8.4	$p(d_j \mid d_{j-1}, d_{j-2}, p_j^c)$	0.83
Eq. 8.6	$p(d'_j \mid p_j^c)$	0.84
Eq. 8.7	$p(d_j \mid d_{j-1}, d_{j-2}, p_{j-1}^c, p_j^c, p_{j+1}^c)$	0.86
Eq. 8.8	$p(d'_j \mid p_{j-1}^c, p_j^c, p_{j+1}^c)$	0.87

	Page level						Sequence level					
	TP	FP	FN	P	R	F1	TP	FP	FN	P	R	F1
Form A	207	4	1	0.98	1.00	0.99	108	9	8	0.92	0.93	0.93
Form B	13	0	0	1.00	1.00	1.00	4	0	0	1.00	1.00	1.00
Form C	151	9	7	0.94	0.96	0.95	79	24	13	0.77	0.86	0.81
Form D	171	0	10	1.00	0.94	0.97	108	10	12	0.92	0.90	0.91
Form E	2	0	0	1.00	1.00	1.00	2	0	0	1.00	1.00	1.00
Form F	15	0	0	1.00	1.00	1.00	12	0	0	1.00	1.00	1.00
Form G	100	0	0	1.00	1.00	1.00	22	0	0	1.00	1.00	1.00
Form H	56	0	0	1.00	1.00	1.00	53	0	0	1.00	1.00	1.00
Form I	6	0	0	1.00	1.00	1.00	6	0	0	1.00	1.00	1.00
Form J	14	0	0	1.00	1.00	1.00	14	0	0	1.00	1.00	1.00
Form K	10	3	0	0.77	1.00	0.87	10	3	0	0.77	1.00	0.87
Form L	74	11	12	0.87	0.86	0.87	21	9	7	0.70	0.75	0.72
Form M	52	11	10	0.83	0.84	0.83	18	6	4	0.75	0.82	0.78
Form N	13	0	0	1.00	1.00	1.00	13	0	0	1.00	1.00	1.00
Form O	2	1	3	0.67	0.40	0.50	1	2	3	0.33	0.25	0.29
Form P	167	8	4	0.95	0.98	0.97	106	23	11	0.82	0.91	0.86
Form Q	22	0	0	1.00	1.00	1.00	22	0	0	1.00	1.00	1.00
Form R	51	0	0	1.00	1.00	1.00	16	0	0	1.00	1.00	1.00
Form S	1	5	0	0.17	1.00	0.29	1	5	0	0.17	1.00	0.29
Form T	226	0	0	1.00	1.00	1.00	66	0	0	1.00	1.00	1.00
Form U	64	4	2	0.94	0.97	0.96	48	8	4	0.86	0.92	0.89
Form V	4	2	1	0.67	0.80	0.73	0	6	2	0.00	0.00	0.00
Form W	9	3	1	0.75	0.90	0.82	9	3	1	0.75	0.90	0.82
Form X	55	0	0	1.00	1.00	1.00	28	0	0	1.00	1.00	1.00
Form Y	376	4	13	0.99	0.97	0.98	248	18	15	0.93	0.94	0.94
Form Z	26	0	1	1.00	0.96	0.98	6	3	2	0.67	0.75	0.71
Form AA	326	8	8	0.98	0.98	0.98	26	18	7	0.59	0.79	0.68
Form AB	26	0	0	1.00	1.00	1.00	21	5	1	0.81	0.95	0.88
Form AC	332	0	2	1.00	0.99	1.00	20	2	2	0.91	0.91	0.91
Form AD	17	2	0	0.89	1.00	0.94	17	2	0	0.89	1.00	0.94
Σ	2588	75	75	–	–	—	1105	156	92	–	–	–
Micro averages	–	–	–	0.97	0.97	0.97	–	–	–	0.88	0.92	0.90
Macro averages	–	–	–	0.91	0.95	0.92	–	–	–	0.82	0.89	0.84

Production Deployments

- The deployment of an automatic document separation solution is a lengthy process, as is common for any workflow-changing installation in large organizations. Most often, a proof-of-concept phase precedes the deployment proper.
- This part of a project can be pre-sales in order to demonstrate the feasibility of the approach to the customer or it can be as the first step in a deployment to find out how much automation can be introduced with high accuracy. In a proof of concept (POC), only a small subset of document types are considered for classification and separation.
- This poses a set of unique problems to consider: The document separator is normally set up to classify all documents into a set of well-known and well-defined document types. In a POC, only a subset of document types (say, 10 out of 50) is relevant.
- However, the incoming batches still contain documents of all types. The challenge here is to "actively ignore" the remaining document types without adverse effects on the classification and separation results for the document types on which we are concentrating.

Conclusion

- We presented an automatic solution for the classification and separation of paper documents.
- The problem is to ingest a long sequence of images of paper pages and to convert those into a sequence of documents with definite boundaries and document types.
- In a manual setting, this process is costly and error prone.
- The automatic solution we describe prepares the incoming pages by running them through an OCR process to discover the text on the page.
- Basic NLP techniques for segmentation and morphological processing are used to arrive at a description of a page that associates stems with occurrence counts for a page (bag-of-words model).
- An SVM classifier is applied to generate probabilities that pages are of a given document and page type.

References

1. Schmidtler, M., Texeira, S., Harris, C., Samat, S., Borrey, R., Macciola, A.: Automatic document separation. United States Patent Application 20050134935, US Patent & Trademark Office (2005)
2. Ratnaparkhi, A.: A Simple Introduction to Maximum Entropy Models for Natural Language Processing. IRCS Report 97-08, University of Pennsylvania, Philadelphia, PA (1997)
3. Reynar, J., Ratnaparkhi, A.: A Maximum Entropy Approach to Identifying Sentence Boundaries. In: Proceedings of the ANLP97, Washington, D.C. (1997)
4. Collins-Thompson, K., Nickolov, R.: A Clustering-Based Algorithm for Automatic Document Separation. In: SIGIR 2002 Workshop on Information Retrieval and OCR. (2002)
5. Pevzner, L., Hearst, M.: A Critique and Improvement of an Evaluation Metric for Text Segmentation. Computational Linguistics **28** (2002) 19–36
6. Porter, M.: An Algorithm for Suffix Stripping. Program **14** (1980) 130–130
7. Joachims, T.: Learning to Classify Text using Support Vector Machines: Methods, Theory, and Algorithms. Kluwer (2002)
8. McCallum, A., Freytag, D., Pereira, F.: Maximum entropy markov models for information extraction and segmentation. Technical report, Just Research, AT&T Labs — Research (2000)
9. Goodman, J.: A bit of progress in language modeling. Technical Report MSR-TR-2001-72, Machine Learning and Applied Statistics Group Microsoft Research (2001)
10. Vapnik, V.: Statistical Learning Theory. JOHN WILEY & SONS, INC (1998)

References

11. Jaakola, T., Meila, M., Jebara, T.: Maximum entropy discrimination. Technical report, MIT AI Lab, MIT Media Lab (1999)
12. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Technical Report LS-8 Report 23, Universitat Dortmund Fachbereich Informatik Lehrstuhl VIII Kunstliche Intelligenz (1997)
13. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularised likelihood methods. Technical report, Microsoft Research (1999)
14. Harris, C., Schmidtler, M.: Effective multi-class support vector machine classification. United States Patent Application 20040111453, US Patent & Trademark Office (2004)
15. Jelinek, F.: Statistical Methods for Speech Recognition. Language, Speech and Communication. MIT Press, Cambridge, Massachusetts (1998)
16. Pereira, F., Riley, M.: Speech recognition by composition of weighted finite automata. Technical report, AT&T Labs — Research (1996)
17. Mohri, M., Pereira, F.C.N., Riley, M.: A Rational Design for a Weighted Finite-State Transducer Library. In: Workshop on Implementing Automata. (1997) 144–158
18. Lowerre, B.T.: The HARPY Speech Recognition System. PhD thesis, Carnegie Mellon University (1976)

|| Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi& Approved by AICTE, New Delhi.)

No. 67, BGS Health & Education City, Dr. Vishnuvardhan Road Kengeri, Bengaluru – 560 060

Subject: Natural Language Processing(18CS743)

By

CHETAN R, Assistant Professor

Semester / Section: 7A and B

Department of Information Science & Engineering

Aca. Year: ODD SEM /2021-22

INFORMATION RETRIEVAL

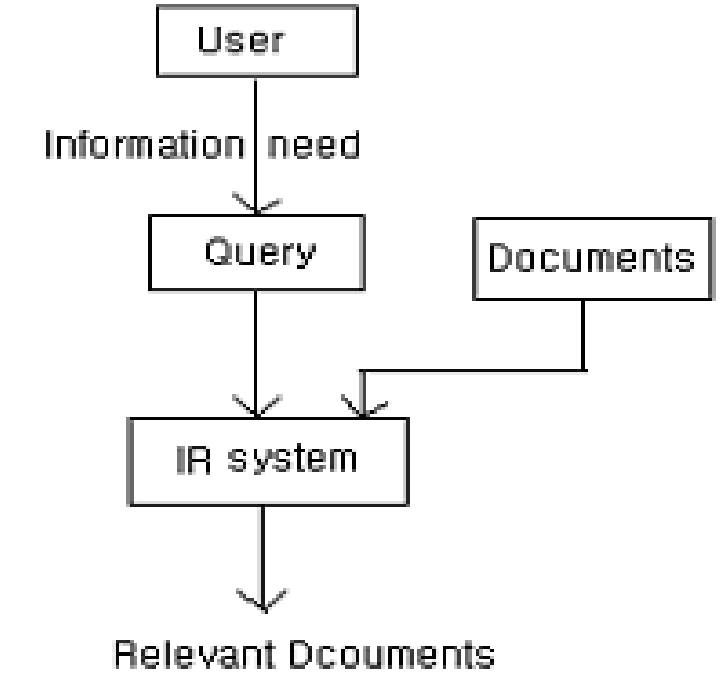
Introduction

- Information retrieval deals with the organization, storage, retrieval and evaluation of information relevant to a users query.
- A user in need of information formulates a request in the form of a query written in a natural language. The retrieval system responds by retrieving the document that seems relevant to the query.
- It dates back to the 1960s when text retrieval systems were introduced.
- Many NLP techniques, including the probabilistic model, have found application in IR systems and techniques such as latent semantic indexing, vector space retrieval, etc.
- Traditionally, IR systems are not expected to return the actual information, only documents containing that information, only documents containing that information.

Contd..

- “An information retrieval system does not inform the user on the subject of her inquiry. It merely informs on the existence and whereabouts of documents relating to her request.”
- The word document is a general term that includes non-textual information such as images and speech.

Design Features of Information Retrieval Systems



Basic Information Retrieval Process

Indexing

- The process of transforming document text to some representation of it is known as *indexing*.
- Different index structures might be used. One commonly used data structure by IR system is ***inverted index***.
- The word term can be a single word or multi-word phrases.
- The method used for phrase extraction is as follows:

1. Any pair of adjacent non-stop words is regarded a potential phrase.
2. The final list of phrases is composed of those pairs of words that occur in, say 25 or more documents in document collection.

Eliminating Stop words

- Stop words are high frequency words which have little semantic weight and are thus unlikely to help in retrieval.

```
1 print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you',
'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'sh
t's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves
t', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'b
ng', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'i
f', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into',
e', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'o
e', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'bo
me', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than',
'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 'r
n', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't
"isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "ne
n't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',
```

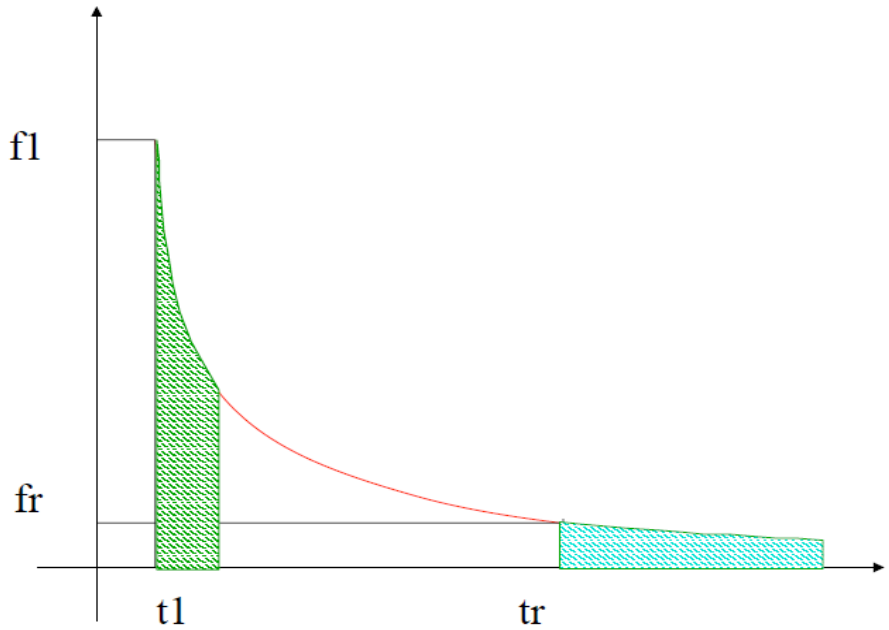
Stemming

- Stemming is the process of producing morphological variants of a root/base word.
- Stemming programs are commonly referred to as stemming algorithms or stemmers.
- A stemming algorithm reduces the words “chocolates”, “chocolatey”, “choco” to the root word, “chocolate” and “retrieval”, “retrieved”, “retrieves” reduce to the stem “retrieve”.
- Stemming is an important part of the pipelining process in Natural language processing. The input to the stemmer is tokenized words.

Zipf Law

Rank r and frequency f:

$$r \times f = const$$



Information Retrieval Model

An IR model is a pattern that defines several aspects of retrieval procedure, for example,

- how the documents and user's queries are represented
- how system retrieves relevant documents according to users' queries &
- how retrieved documents are ranked.

IR Model

- An IR model consists of
  - a model for documents
  - a model for queries and
  - a matching function which compares queries to documents.
  - a ranking function

Classical IR Model

IR models can be classified as:

- Classical models of IR
- Non-Classical models of IR
- Alternative models of IR

Classical IR Model

- based on mathematical knowledge that was easily recognized and well understood
- simple, efficient and easy to implement
- The three classical information retrieval models are:

  -Boolean

  -Vector and

  -Probabilistic models

Non-Classical models of IR

Non-classical information retrieval models are based on principles other than similarity, probability, Boolean operations etc. on which classical retrieval models are based on.

information logic model, *situation theory model* and *interaction model*.

Alternative IR models

- Alternative models are enhancements of classical models making use of specific techniques from other fields.

Example:

Cluster model, fuzzy model and latent semantic indexing (LSI) models.

Information Retrieval Model

- The actual text of the document and query is not used in the retrieval process. Instead, some representation of it.
- Document representation is matched with query representation to perform retrieval
- One frequently used method is to represent document as a set of index terms or keywords

Boolean model

- the oldest of the three classical models.
- is based on Boolean logic and classical set theory.
- represents documents as a set of keywords, usually stored in an inverted file.

Boolean model

- Users are required to express their queries as a boolean expression consisting of keywords connected with boolean logical operators (AND, OR, NOT).
- Retrieval is performed based on whether or not document contains the query terms.

Boolean model

Given a finite set

$$T = \{t1, t2, \ldots, ti, \ldots, tm\}$$

of index terms, a finite set

$$D = \{d1, d2, \ldots, dj, \ldots, dn\}$$

of documents and a boolean expression in a normal form - representing a query Q as follows:

$$Q = \wedge(\vee \theta_i), \theta i \in \{t_i, \neg t_i\}$$

Boolean model

1. The set R_i of documents are obtained that contain or not term t_i:

$$R_i = \{ d_j \mid \theta i \in d_j \}, \ \theta i \in \{t_i, \neg t_i\},$$

where $\neg t_i \in d_j$ means $t_i \notin d_j$

2. Set operations are used to retrieve documents in response to Q:

$$\cap R_i$$

Example:

- Let the set of original documents be D = { D_1, D_2, D_3}, where
- D_1 = Information retrieval is concerned with the organization, storage, retrieval, and evaluation of information relevant to users query.
- D_2 = A user having an information needs to formulate a request in the form of query written in natural language.
- D_3 = The retrieval system responds by retrieving the document that seems relevant to the query.
- Let the set of terms used to represent these documents be
- T = {information, retrieval, query}
- Then the set D of document will be represented as follows: D = {d_1, d_2, d_3} where

d_1 = {information, retrieval, query}

d_2 = {information, query}

d_3 = {retrieval, query}

Contd..

- Let the query Q be
- Q = information ^ retrieval
- First the sets R_1 and R_2 of documents are retrieved in response to Q, where
- $R_1 = \{d_j \mid \text{information} \in d_j\} = \{d_1, d_2\}$
- $R_2 = \{d_j \mid \text{retrieval} \in d_j\} = \{d_1, d_3\}$
- Then, the following documents are retrieved in response to query Q $\{d_j \mid d_j \in R_1 \cap R_2\} = \{d_1\}$

Probabilistic Model

Capture the IR problem in a probabilistic framework.

- first probabilistic model (Binary Independent Retrieval Model) by Robertson and Spark-Jones in 1976...
- late 90s, emergence of language models, still hot topic in IR
- Overall question: "what is the probability for a document to be relevant to a query ?" several interpretation of this sentence

Contd..

- Given a set of documents D, a query q, and a cut-off value α, this model first calculates the probability of relevance and irrelevance of a document to the query.
- It then ranks documents having probabilities of relevance at least that of irrelevance in decreasing order of their relevance.
- Documents are retrieved if the probability of relevance in the ranked list exceeds the cut off value.

Contd..

- More formally, if P(R/d) is the probability of relevance of a document d, for query q, and P(I/d) is the probability of irrelevance, then the set of documents retrieved in response to the query q is as follows.
- $S = \{d_j \mid P(R/d_j) \geq P(I/d_j)\}\ P(R/d_j) \geq \alpha$

Vector Space Model

- The vector space model is one of the most well-studied retrieval models.
- It represents documents and queries as vectors of features representing terms that occur within them. Each document is characterized by a Boolean or numerical vector.
- These vectors are represented in a multidimensional space, in which each dimension corresponds to a distinct term in the corpus of documents.
- In its simplest form, each feature takes a value of either zero or one, indicating the absence or presence of that term in a document or query.
- Ranking algorithms compute the similarity between document and query vectors to yield a retrieval score to each document.
- This score is used to produce a ranked list of retrieved documents.

Contd..

- Given a finite set of n documents D = $\{d_1, d_2, .. d_j, .., d_n\}$ and a finite set of m terms T = $\{t_1, t_2, .. t_i, .., t_m\}$ and each document is represented by a column vector of weights as follows:

$$(w_{1j}, w_{2j}, w_{3j}, \ldots, w_{ij}, \ldots, w_{mj})^t$$

- The document collection as a whole is represented by an mxn term document matrix as

$$\begin{pmatrix} w_{11} & w_{12} & \ldots & w_{1j} & \ldots & w_{1n} \\ w_{21} & w_{22} & \ldots & w_{2j} & \ldots & w_{2n} \\ w_{i1} & w_{i2} & \ldots & w_{ij} & \ldots & w_{in} \end{pmatrix}$$

Term Weighting

Each term that is selected as an indexing feature for a document, acts as a discriminator between that document and all other documents in the corpus. Luhn (1958) attempted to quantify the discriminating power of the terms by associating the frequency of their occurrence (term frequency) within the document. He postulated that the most discriminating (content bearing) terms are mid frequency terms. This postulate can be refined by noting the following facts:

1. The more a document contains a given word, the more that document is about a concept represented by that word.
2. The less a term occurs in particular document in a collection, the more discriminating that term is.

Inverse Document Frequency

$$\text{idfi} = \log\left(\frac{n}{n_i}\right)$$

Inverse document frequency (idf) attaches more importance to more specific terms. If a term occurs in all documents in a collection, its idf is 0. Researchers have attempted to include term distribution in the weighting function (Sparck-Jones 1972, Salton 1971) to give a more accurate quantification of term importance. Sparck-Jones showed experimentally that a weight of $\log(n/n_i)+1$, termed as inverse document frequency, leads to more effective

term frequency (tf) and idf weights, resulting in a family of tf × idf weight schemes having the following general form:

$$w_{ij} = \mathrm{tf}_{ij} \times \log\left(\frac{n}{n_i}\right)$$

Example 9.3 Consider a document represented by the three terms {tornado, swirl, wind} with the raw tf 4, 1, and 1 respectively. In a collection of 100 documents, 15 documents contain the term *tornado*, 20 contain *swirl*, and 40 contain *wind*. The idf of the term *tornado* can be computed as

$$\log\left(\frac{n}{n_i}\right) = \log\left(\frac{100}{15}\right) = 0.82$$

Table 9.2 Computing tf-idf weight

Term	Frequency (tf)	Document frequency (n_i)	idf [$\log(n/n_i)$]	Weight (tf × idf)
Tornado	4	15	0.824	0.296
Swirl	1	20	0.699	0.699
Wind	1	40	0.398	0.389

A third factor that may affect weighting function is the document length. A term appearing the same number of times in a short document and in a long document, will be more valuable to the former. Most weighting schemes can thus be characterized by the following three factors:

- Within-document frequency or term frequency (tf)
- Collection frequency or inverse document frequency (idf)
- Document length

Table 9.3 Calculating weight with different options for the three weighting factors

	Term frequency within document		
A	*n*	$tf = tf_{ij}$	Raw term frequency
	b	tf = 0 or 1 (binary weight)	
	a	$tf = 0.5 + 0.5\left(\frac{tf_{ij}}{\max tf \text{ in } D_j}\right)$	Augmented term frequency
	l	$tf = \ln(tf_{ij}) + 1.0$	Logarithmic term frequency
	L	$tf = \frac{\ln(tf_{ij} + 1.0)}{1.0 + \ln[\text{mean}(tf \text{ in } D_j)]}$	Average term frequency-based normalization
	Inverse document frequency		
B	*n*	wt = tf	No conversion
	t	$wt = tf \cdot \ln\left(\frac{n}{n_i}\right)$	Multiply tf with idf
	Document length		
C	*n*	$w_{ij} = wt$	(no conversion)
	c	w_{ij} is obtained by dividing each wt by sqrt [sum of (wts squared)]	

Automatic Method for Obtaining Indexed representation

Step 1 Tokenization This extracts individual terms form a document, converts all the letters to lower case, and removes punctuation marks. The output of the first stage is a representation of the document as a stream of terms.

Step 2 Stop word elimination This removes words that appear more frequently in the document collection.

Step 3 Stemming This reduces the remaining terms to their linguistic root, to obtain the index terms.

Step 4 Term weighting This assigns weights to terms according to their importance in the document, in the collection, or some combination of both.

Table 9.4 Vector representation of sample documents after stemming

Stemmed terms	Document 1	Document 2	Document 3
inform	0	0	1
intellig	0	0	1
model	1	1	0
probabilist	0	1	0
retriev	0	1	1
space	1	0	0
technique	0	0	1
vector	1	0	0

Similarity Measures

The inner product is given by

$$\text{sim}\ (d_j,\ q_k) = (d_j,\ q_k) = \sum_{i=1}^{m} w_{ij} \times w_{ik}$$

where m is the number of terms used to represent documents in the collection.

Other measures, e.g., dice coefficient, Jaccard's coefficient, and cosine coefficient, attempt to normalize the similarity by the length of document and query. The dice coefficient defines the similarity between query k and document j as

$$\text{sim}\ (d_j,\ q_k) = \frac{2 \times \left(\sum_{i=1}^{m} w_{ij} \times w_{ik} \right)}{\sum_{i=1}^{m} w_{ij}^2 + \sum_{i=1}^{m} w_{ik}^2}$$

Jaccard's coefficient is defined as

$$\text{sim}\ (d_j,\ q_k) = \frac{\sum_{i=1}^{m} w_{ij} \times w_{ik}}{\sum_{i=1}^{m} w_{ij}^2 + \sum_{i=1}^{m} w_{ik}^2 - \sum_{i=1}^{m} w_{ij} \times w_{ik}}$$

NON-CLASSICAL MODELS OF IR

INFORMATION LOGIC MODEL

The *information logic model* is based on a special logic technique called logical imaging. Retrieval is performed by making inferences from document to query. This is unlike classical models, where a search process is used. Unlike usual implication, which is true in all cases except that when antecedent is true and consequent is false, this inference is uncertain. Hence, a measure of uncertainty is associated with this inference. The principle put forward by van Rijsbergen is used to measure this uncertainty. This principle says:

Given any two sentences x and y, a measure of the uncertainty of $y \rightarrow x$ *relative to a given data set is determined by the minimal extent to which one has to add information to the data set in order to establish the truth of* $y \rightarrow x$.

SITUATION THEORY

The *situation theory* model is also based on van Rijsbergen's principle. Retrieval is considered as a flow of information from document to query. A structure called *infon*, denoted by ι, is used to describe the situation and to model information flow. An infon represents an n-ary relation and its polarity. The polarity of an infon can be either 1 or 0, indicating that the infon carries either positive or negative information.

For example, the information in the sentence, *Adil is serving a dish*, is conveyed by the infon

$$\iota = \langle\langle \textit{serving Adil, dish}; 1 \rangle\rangle$$

Interaction IR Model

The *interaction IR* model was first introduced in Dominich (1992, 1993) and Rijsbergen (1996). In this model, the documents are not isolated; instead, they are interconnected. The query interacts with the interconnected documents. Retrieval is conceived as a result of this interaction. This view of interaction is taken from the concept of interaction as realized in the Copenhagen interpretation of quantum mechanics. Artificial neural

ALTERNATIVE MODELS OF IR

CLUSTER MODEL

The cluster model is an attempt to reduce the number of matches during retrieval. The need for clustering was first pointed out by Salton. Before we discuss the cluster-based IR model, we would like to state the cluster hypothesis that explains why clustering could prove efficient in IR.

Closely associated documents tend to be relevant to the same clusters.

Clustering can be applied on terms instead of documents. Thus, terms can be grouped to form classes of co-occurrence terms. Co-occurrence terms can be used in dimensionality reduction or thesaurus construction. A number of methods are used to group documents. We discuss here, a cluster generation method based on similarity matrix. This method works as follows:

Let $D = \{d_1, d_2, \ldots, d_j, \ldots, d_m\}$ be a finite set of documents, and let $E = (e_{ij})_{n,n}$ be the similarity matrix. The element E_{ij} in this matrix, denotes a similarity between document d_i and d_j. Let T be the threshold value. Any pair of documents d_i and d_j $(i \neq j)$ whose similarity measure exceeds the threshold $(e_{ij} \geq T)$ is grouped to form a cluster. The remaining documents form a single cluster. The set of clusters thus obtained is

$$C = \{C_1, C_2, \ldots, C_k, \ldots, C_p\}$$

Representation vector for a cluster C_k is

$$r_k = \{a_{1k}, a_{2k}, \ldots, a_{ik}, \ldots, a_{mk}\}$$

An element a_{ik} in this vector is computed as

$$a_{ik} = \frac{\sum_{d_j \in C_k} a_{ij}}{|C_k|}$$

During retrieval, the query is compared with the cluster vectors

$$(r_1, r_2, \ldots, r_k, \ldots, r_p)$$

This comparison is carried out by computing the similarity between the query vector q and the representative vector r_k as

$$s_{ik} = \sum_{i=1}^{m} a_{ik} q_i, \quad k = 1, 2, \ldots, p$$

A cluster C_k whose similarity s_k exceeds a threshold is returned and the search proceeds in that cluster.

EXAMPLE

Let

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

be the term-by-document matrix. The similarity matrix corresponding to these documents is

$$\begin{matrix} 1.0 & & \\ 0.9 & 1.0 & \\ 0.4 & 0.4 & 1.0 \end{matrix}$$

Using a threshold of 0.7, we get the following two clusters:

$C_1 = \{d_1, d_2\}$

$C_2 = \{d_3\}$

The cluster vectors (representatives) for C_1 and C_2 are

$r_1 = (1\ 0.5\ 1\ 0\ 1)$

$r_2 = (0\ 0\ \ 1\ 1\ 0)$

Retrieval is performed by matching the query vector with r_1 and r_2.

FUZZY MODEL

In the fuzzy model, the document is represented as a fuzzy set of terms, i.e., a set of pairs $[t_i, \mu(t_i)]$, where μ is the membership function. The membership function assigns to each term of the document a numeric membership degree. The membership degree expresses the significance of term to the information contained in the document. Usually, the significance values (weights) are assigned based on the number of occurrences of the term in the document and in the entire document collection, as discussed earlier. Each document in the collection

$$D = \{d_1, d_2, \ldots, d_j, \ldots, d_n\}$$

can thus be represented as a vector of term weights, as in the following vector space model

$$(w_{1j}, w_{2j}, w_{3j}, \ldots, w_{ij}, \ldots, w_{mj})^t$$

Each term in the document is considered a representative of a subject area and w_{ij} is the membership function of document d_j to the subject area represented by term t_i. Each term t_i is itself represented by a fuzzy set f_i in the domain of documents given by

$$f_i = \{(d_j, w_{ij})\} \mid i = 1,. \ldots, m; j = 1, \ldots, n$$

This weighted representation makes it possible to rank the retrieved documents in decreasing order of their relevance to the user's query.

Example

Example 9.5 Consider the following three documents:

d_1 = {information, retrieval, query}
d_2 = {retrieval, query, model}
d_3 = {information, retrieval}

where the set of terms used to represent documents is

T = {information, model, query, retrieval}

The fuzzy sets induced by these terms are

$f_1 = \{(d_1, 1/3), (d_2, 0) (d_3, 1/2)\}$
$f_2 = \{(d_1, 0), (d_2, 1/3) (d_3, 0)\}$
$f_3 = \{(d_1, 1/3), (d_2, 1/3) (d_3, 0)\}$
$f_4 = \{(d_1, 1/3), (d_2, 1/3) (d_3, 1/2)\}$

If the query is $q = t_2 \wedge t_4$, then document d_2 will be returned.

Latent Semantic Indexing Model

Latent semantic indexing model is the application of single value decomposition to IR. The use of latent semantic indexing (LSI) is based on the assumption that there is some underlying 'hidden' semantic structure in the pattern of word-usage across documents, rather than just surface level word choice. LSI attempts to identify this hidden semantic structure through statistical techniques and use it to represent and retrieve

Now we discuss how the LSI technique is actually employed in IR. The document collection is first processed to get a $m \times n$ term-by-document matrix, W, where m is the number of index terms and n is the total number of documents in the collection. Columns in this matrix represent document vectors, whereas the rows denote term vectors. The matrix element W_{ij} represents the weight of the term i in document j. The weight may be assigned based on term frequency or some combination of local and global weighting, as in the case of vector space model. Singular value decomposition (SVD) of the term-by-document matrix is then computed. Using SVD, the matrix is represented as a product of three matrices

$$W = TSD^T$$

Example

Example 9.6 Consider the matrix shown in Figure 9.5. This matrix defines five-dimensional space in which six documents, d_1, d_2, d_3, ..., d_6, have been represented. The five dimensions correspond to five index terms *tornado, storm, tree, forest,* and *farming*. For simplicity, tf has been used to weight index terms. Figure 9.6 shows the documents in a two-dimensional space. The vectors in the figure correspond to document vectors in the matrix R, which is the representation of X in reduced two-dimensional space. The two dimensions correspond to derived concepts obtained through the application of truncated SVD.

$$X = \begin{array}{l|cccccc} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \hline \text{tornado} & 1 & 1 & 0 & 0 & 0 & 0 \\ \text{storm} & 1 & 0 & 1 & 0 & 1 & 0 \\ \text{tree} & 1 & 0 & 1 & 0 & 0 & 0 \\ \text{forest} & 0 & 0 & 1 & 1 & 0 & 0 \\ \text{farming} & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Figure 9.5 A term-document matrix representing six documents in five-dimensional space

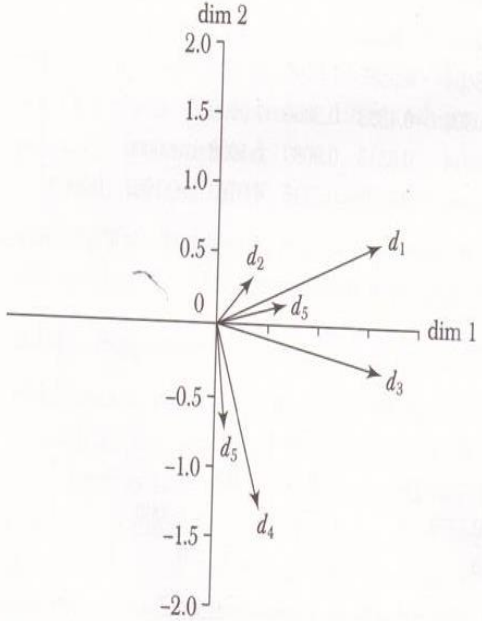


Figure 9.6 Documents in reduced two-dimensional space

$$T = \begin{pmatrix} 0.3318 & 0.3338 & 0.8064 & -0.2426 & -0.2634 \\ 0.6693 & 0.1616 & -0.2737 & 0.5853 & -0.3293 \\ 0.5514 & 0.1038 & -0.0961 & -0.2667 & 0.7777 \\ 0.3583 & -0.5745 & -0.2148 & -0.5778 & -0.4021 \\ 0.0974 & -0.7223 & 0.4684 & 0.4400 & 0.2362 \end{pmatrix}$$

$$S = \begin{pmatrix} 2.3830 & 0 & 0 & 0 & 0 \\ 0 & 1.6719 & 0 & 0 & 0 \\ 0 & 0 & 1.2415 & 0 & 0 \\ 0 & 0 & 0 & 0.8288 & 0 \\ 0 & 0 & 0 & 0 & 0.5454 \end{pmatrix}$$

Figure 9.8 The matrix S for singular values of the SVD of the term-document matrix X

$$D^T = \begin{pmatrix} 0.6515 & 0.1392 & 0.6626 & 0.1912 & 0.2809 & 0.0409 \\ 0.3584 & 0.1996 & -0.1848 & -0.7756 & 0.0967 & -0.4320 \\ 0.3516 & 0.6495 & -0.4710 & 0.2042 & -0.2205 & 0.3773 \\ 0.0916 & -0.2927 & -0.3127 & -0.1662 & 0.7062 & 0.5309 \\ 0.3392 & -0.4829 & 0.0849 & -0.3042 & -0.6037 & 0.4330 \end{pmatrix}$$

Figure 9.9 The matrix D^T for singular values of the SVD of the term-document matrix

$$R = \begin{pmatrix} 1.5526 & 0.3318 & 1.5790 & 0.4557 & 0.6693 & 0.0974 \\ 0.5992 & 0.3338 & -0.3090 & -1.2967 & 0.1616 & -0.7223 \end{pmatrix}$$

Figure 9.10 The matrix $R_{2\times6} = S_{2\times2}D^T_{2\times6}$ representing documents in two-dimensional space

$$M = \begin{pmatrix} 1.0000 & & & & & \\ 0.9131 & 1.0000 & & & & \\ 0.8464 & 0.5557 & 1.0000 & & & \\ -0.0304 & -0.4353 & 0.5066 & 1.0000 & & \\ 0.9914 & 0.8518 & 0.9089 & 0.1008 & 1.0000 & \\ -0.2322 & -0.6086 & 0.3215 & 0.9793 & -0.1027 & 1.0000 \end{pmatrix}$$

Figure 9.11 The matrix of document correlation $M = N^TN$ in the new space (N is matrix R with length-normalized columns.)

$$Z = \begin{pmatrix} 1.0000 & & & & & \\ 0.5774 & 1.0000 & & & & \\ 0.6667 & 0 & 1.0000 & & & \\ 0 & 0 & 0.4082 & 1.0000 & & \\ 0.5774 & 0 & 0.5774 & 0 & 1.0000 & \\ 0 & 0 & 0 & 0.7071 & 0 & 1.0000 \end{pmatrix}$$

Figure 9.12 The matrix of document correlation $Z = Y^TY$ in the new space (Y is matrix X with length-normalized columns.)

EVALUATION OF IR SYSTEM

CRITERIA FOR EVALUATION

Cleverdon listed the following six criteria that can be used for evaluation:

1. *Coverage of the collection*: The extent to which the system
2. *Time lag*: The time that elapses between submission of a query and getting back the response
3. *Presentation format*
4. *User effort*: The effort made by the user to obtain relevant information
5. *Precision*: The proportion of retrieved documents that are relevant
6. *Recall*: The proportion of relevant documents that are retrieved

EFFECTIVENESS MEASURES

Effectiveness is purely a measure of the ability of a system to satisfy the user in terms of the relevance of documents retrieved (Rijsbergen 1979).

Precision and Recall

Precision is defined as the proportion of relevant documents in a retrieved set. This can be seen as the probability that a relevant document is retrieved. *Recall* is the proportion of relevant documents in a collection that have actually been retrieved. Precision measures the accuracy of a system while recall measures its exhaustiveness. Precision and recall can be computed as follows:

$$\text{Precision} = \frac{\text{Number of relevant document retrieved } (NR_{\text{ret}})}{\text{Total number of documents retrieved } (N_{\text{ret}})}$$

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved } (NR_{r})}{\text{Total number of relevant documents in the collection } (NR_{\text{rel}})}$$

	Relevant	Non-relevant	
Retrieved	$A \cap B$	$\bar{A} \cap B$	B
Not-retrieved	$A \cap \bar{B}$	$\bar{A} \cap \bar{B}$	$\bar{B}$
	A	$\bar{A}$	

Figure 9.13 Relevant matrix

Referring to Figure 9.13, precision and recall will be given as follows:

$$\text{Precision} = \frac{|A \cap B|}{|B|} = \frac{NR_{\text{ret}}}{N_{\text{ret}}}$$

$$\text{Recall} = \frac{|A \cap B|}{|A|} = \frac{NR_{\text{ret}}}{NR_{\text{rel}}}$$

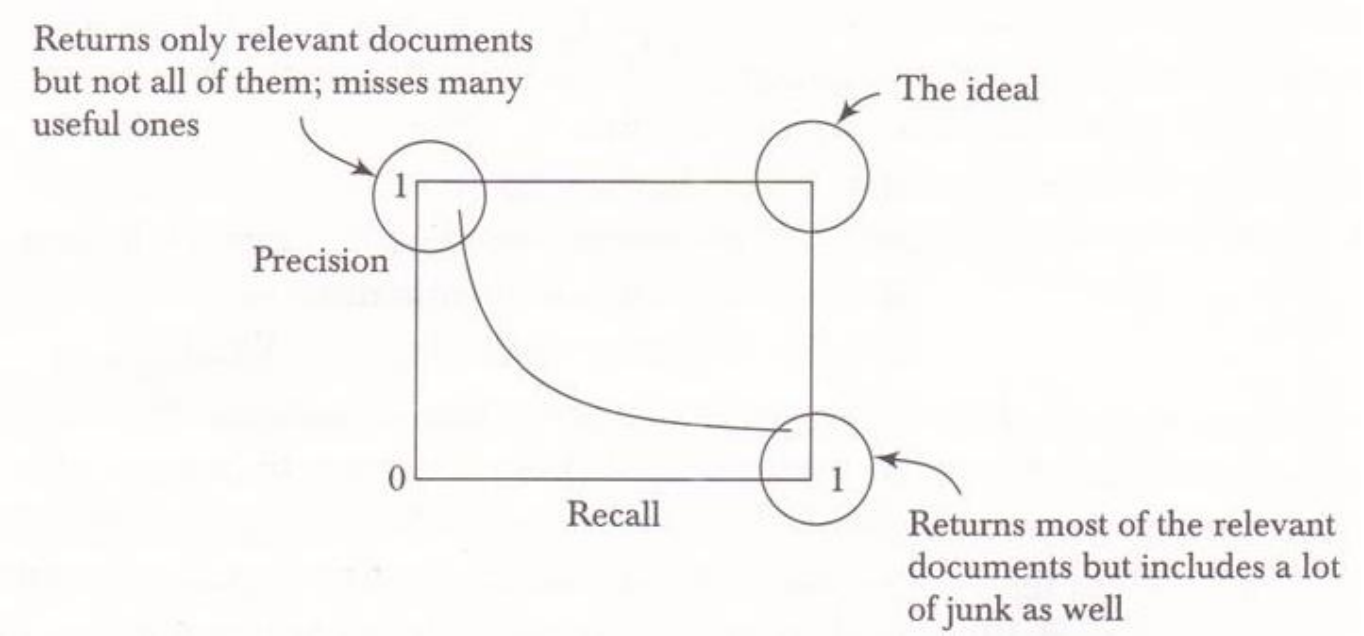
where A = Set of relevant documents

$|A|$ = No. of relevant documents in the collection (NR_{rel})

B = Set of retrieved documents

$|B|$ = No. of retrieved documents (NR_{ret})

Trade off Between Precision and Recall



F-Measure

The F-measure takes into account both precision and recall. It is defined as the harmonic mean of recall and precision.

$$F = \frac{2PR}{P+R}$$

$$E = \frac{(1+\beta^2)PR}{\beta^2 P + R} = \frac{(1+\beta^2)}{\frac{\beta^2}{R} + \frac{1}{P}}$$

Normalized recall measures how close the set of retrieved documents is to an ideal retrieval, in which the most relevant NR_{rel} document appears in the first NR_{rel} position. Relevant documents are ranked 1, 2, 3, ..., NR_{rel}, where NR_{rel} is the number of relevant documents. The ideal rank is given by

$$IdR = \frac{\sum_{r=1}^{NR_{rel}} r}{NR_{rel}}$$

Let the average rank (AvR) over the set of relevant documents retrieved by a system be

$$AvR = \frac{\sum_{r=1}^{NR_{rel}} Rank_r}{NR_{rel}}$$

User Centered Evaluation

The system-driven model is still the dominant approach followed in IR research for evaluation of IR systems. The evaluation here, is made on a test collection having known relevance judgments. These relevance judgements were usually provided by problem domain experts, and are binary, objective, topical, and static in nature, and lack a user's viewpoint. There is also major disagreement among experts in providing relevance judgements (Haynes et al. 1990, Hersh and Hickam 1994). Further, these judgements of relevance are affected not only by the expertise of the judge, but also by the order of the documents (Eisenberg and Barry 1988;

Another drawback of the system-driven approach is that it removes the end users from the retrieval process, substituting them with the queries and judgements provided with the test collection. This allows fast experimentation but makes it difficult to evaluate the effect of interactive IR techniques and is suitable only for non-interactive environment (Draper and Dunlop). In an interactive setting, a user normally starts with a query, which goes through many refinements to eventually get the desired documents. The test-collection approach poses a problem in such an environment. As performance of the IR system will eventually be measured in terms of it ability to retrieve documents relevant to a user's query, it seems realistic to follow a user-centered approach to evaluation. Such an approach will result in a much more direct measure of the overall goal. A