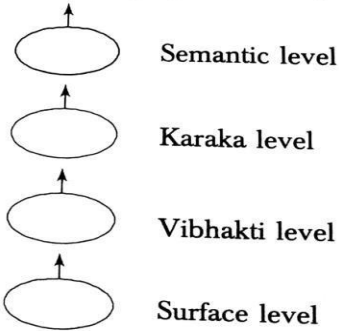


Sl No	Questions and Solutions	Marks
1.a	<p>What is Natural Language Processing. Explain different challenges in NLP with examples.</p> <p>Natural Language Processing is concerned with development of computational models of aspects of Human Language Processing.</p> <p>Challenges:</p> <ul style="list-style-type: none"> ● Representing & interpreting NL is a challenging task. ● Natural languages are highly ambiguous and vague, achieving such representation can be difficult. ● It is almost impossible to embody all sources of knowledge that humans use to process language. ● Identifying the semantics in natural language is difficult. ● Words alone do not make sentence. It is their syntactic and semantic relation that give meaning to a sentence. ● A language keeps on evolving. 	10
1.b	<p>What are Karaka Relations. Explain Karaka theory with example.</p> <p>Karaka literally means CASE, these case relations are based on the way the word group participates in the activity denoted by the verb group. Karaka relations are assigned based on the roles played by various participants in main activity.</p> <p>Levels of Paninian Grammar:</p> <div style="text-align: center;">  </div> <p>Example- maan Bachche ko aangan mein haath se rotii khilaatii hei</p> <p>Various karaka's are (case marker in hindi)</p> <ol style="list-style-type: none"> 1. Karta (subject) - maan 2. Karma (Object) - rotii 3. Karana (instrument)- haath 4. Sampradana (beneficiary)- bachche 5. Apadan (separation)- ko, se/dwara, ke (Case marker) 	10
2.a	<p>Explain n-gram model. How data sparseness problem is handled in n-gram.</p> <ul style="list-style-type: none"> ● A statistical language model is a probability distribution P(s) over all possible word sequences. ● The dominant approach in statistical language modelling is the n-gram model. <p>n-gram Model</p> <p>The goal of a statistical language model is to estimate the probability of a sentence.</p> <p>This is achieved by decomposing sentence probability into a product of conditional probabilities using the chain rule as follows:</p> $P(s)=P(w_1, w_2, w_3, \dots, w_n)$ $=P(w_1) P(w_2/w_1) P(w_3/w_1 w_2) P(w_4/w_1 w_2 w_3) \dots P(w_n/w_1 w_2 \dots w_{n-1})$	10

SCHEME and SOLUTION

	$= \prod_{i=1}^n P(w_i h_i)$ <p>where h_i is history of word w_i defined as $w_1 w_2 \dots w_{i-1}$</p> <ul style="list-style-type: none"> ✓ In order to calculate sentence probability: ✓ calculate the probability of a word, given the sequence of words preceding it. ✓ An n-gram model simplifies the task by approximating the probability of a word given all the previous words by the conditional probability given previous n-1 words only. $P(w_i h_i) \approx P(w_i w_{i-n+1} \dots w_{i-1})$ ✓ Thus, an n-gram model calculates $P(w_i h_i)$ by modelling language as Markov model of order n-1, i.e., by looking at previous n-1 words only. ➤ The n-gram model suffers from data sparseness problem. An n-gram that does not occur in the training data is assigned zero probability, so that even a large corpus has several zero entries in its bi-gram matrix. ➤ This is because of the assumption that the probability of occurrence of a word depends only on the preceding word (or preceding n-1 words), which is not true in general. 	
2.b	<p>Explain with example Binding Theory.</p> <ul style="list-style-type: none"> ➤ Binding is defined by Sells (1985) as follows α binds β iff α C-commands β and α and β are co-indexed ➤ As we noticed in following sentences: $[ei \text{ INFL kill Mukesh}]$ $[Mukesh \ i \ \text{was killed (by ei)}]$ Mukesh was killed ➤ Empty clause (ei) and Mukesh (NP_i) are bound. ➤ This theory gives a relationship between NPs. ✓ Now, binding theory can be given as follows: <ul style="list-style-type: none"> ◆ An anaphor (+a) is bound in its governing category. ◆ A pronominal (+p) is free in its governing category. ◆ An R-expression (-a, -p) is free. ✓ This theory applies to binding at A-positions (argument positions). ✓ Governing category is <ul style="list-style-type: none"> ◆ The local domain NP or ◆ S containing it (G or p or R-expression) and its governor 	04
2.c	<p>Consider the following Training set.</p> <p>The Arabian Knights These are the fairy tales of the east The stories of the Arabian knights are translated in many languages</p> <p>Find the probability of following test sentence using bi-gram model.</p> <p><i>The Arabian Knights are the fairy tales of the east</i></p>	06

	<p>Bi-gram model:</p> <p>$P(\text{the}/\langle s \rangle) = 0.67$ $P(\text{Arabian}/\text{the}) = 0.4$ $P(\text{knight}/\text{Arabian}) = 1.0$</p> <p>$P(\text{are}/\text{these}) = 1.0$ $P(\text{the}/\text{are}) = 0.5$ $P(\text{fairy}/\text{the}) = 0.2$ $P(\text{tales}/\text{fairy}) = 1.0$ $P(\text{of}/\text{tales}) = 1.0$ $P(\text{the}/\text{of}) = 1.0$ $P(\text{east}/\text{the}) = 0.2$ $P(\text{stories}/\text{the}) = 0.2$ $P(\text{of}/\text{stories}) = 1.0$ $P(\text{are}/\text{knight}) = 1.0$ $P(\text{translated}/\text{are}) = 0.5$ $P(\text{in}/\text{translated}) = 1.0$ $P(\text{many}/\text{in}) = 1.0$ $P(\text{languages}/\text{many}) = 1.0$</p> <p>Test sentence(s): The Arabian knights are the fairy tales of the east.</p> <p>$P(\text{The}/\langle s \rangle) \times P(\text{Arabian}/\text{the}) \times P(\text{Knights}/\text{Arabian}) \times P(\text{are}/\text{knight}) \times P(\text{the}/\text{are}) \times P(\text{fairy}/\text{the})$ $\times P(\text{tales}/\text{fairy}) \times P(\text{of}/\text{tales}) \times P(\text{the}/\text{of}) \times P(\text{east}/\text{the})$ $= 0.67 \times 0.4 \times 1.0 \times 1.0 \times 0.5 \times 0.2 \times 1.0 \times 1.0 \times 1.0 \times 0.2$ $= 0.0067$</p>																																																									
<p>3.a</p>	<p>Explain Minimum edit distance algorithm. Compute minimum edit distance between TUTOR and TUMOR</p> <p>The minimum edit distance algorithm is as below</p> <p>Input: Two strings X and Y Output: The minimum edit distance between X and Y</p> <pre> 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=0 to m do for j=0 to n do dist[i,j] = min { dist[i-1,j] + insert_cost, // insert dist[i-1,j-1] + subst_cost(Xi,Yj), // replace dist[i,j-1] + delet_cost } // remove </pre> <p>◆ How the algorithm computes the minimum edit distance between tutor and tumour is shown</p> <table border="1" data-bbox="603 1379 1007 1910"> <tr> <td></td> <td>#</td> <td>t</td> <td>u</td> <td>m</td> <td>o</td> <td>u</td> <td>r</td> </tr> <tr> <td>#</td> <td>0</td> <td>1</td> <td>2</td> <td>3</td> <td>4</td> <td>5</td> <td>6</td> </tr> <tr> <td>t</td> <td>1</td> <td>0</td> <td>1</td> <td>2</td> <td>3</td> <td>4</td> <td>5</td> </tr> <tr> <td>u</td> <td>2</td> <td>1</td> <td>0</td> <td>1</td> <td>2</td> <td>3</td> <td>4</td> </tr> <tr> <td>t</td> <td>3</td> <td>2</td> <td>1</td> <td>1</td> <td>2</td> <td>3</td> <td>4</td> </tr> <tr> <td>o</td> <td>4</td> <td>3</td> <td>2</td> <td>2</td> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td>r</td> <td>5</td> <td>4</td> <td>3</td> <td>3</td> <td>2</td> <td>2</td> <td>2</td> </tr> </table>		#	t	u	m	o	u	r	#	0	1	2	3	4	5	6	t	1	0	1	2	3	4	5	u	2	1	0	1	2	3	4	t	3	2	1	1	2	3	4	o	4	3	2	2	1	2	3	r	5	4	3	3	2	2	2	<p>10</p>
	#	t	u	m	o	u	r																																																			
#	0	1	2	3	4	5	6																																																			
t	1	0	1	2	3	4	5																																																			
u	2	1	0	1	2	3	4																																																			
t	3	2	1	1	2	3	4																																																			
o	4	3	2	2	1	2	3																																																			
r	5	4	3	3	2	2	2																																																			
<p>3.b</p>	<p>List POS Tagging methods. Explain Rule based Tagger with example.</p> <ul style="list-style-type: none"> ✓ Part-of-speech tagging is the process of assigning a part-of-speech to each word in a sentence. ✓ The input to a tagging algorithm is the sequence of words and specified tag sets. 	<p>10</p>																																																								

SCHEME and SOLUTION

	<p>✓ The output is a single best part-of-speech tag for each word.</p> <p>Rule-based Tagger</p> <ul style="list-style-type: none"> ➤ Most rule-based taggers have a 2-stage architecture. ➤ The first stage is simply a dictionary look-up procedure, which returns <ul style="list-style-type: none"> ◆ a set of potential tags and ◆ appropriate syntactic features for each word. ➤ The second stage uses a set of hand-coded rules <ul style="list-style-type: none"> ◆ to discard contextually illegitimate tags ◆ to get a single part-of-speech of each word. <p>Example 1: The show must go on, The potential tags for the word show is {VB, NN}.</p> <ul style="list-style-type: none"> ➤ We resolve this ambiguity by using the rule “IF preceding word is determiner THEN eliminate VB tag”. ➤ Using this rule the word ‘show’ can only be noun in given sentence. <p>Example 2: rule that uses morphological information.</p> <p>✓ IF word ends in ---ing and preceding word is a verb THEN label it a verb (VB)</p> <p>Advantages:</p> <ul style="list-style-type: none"> ➤ Speed is an advantage of the rule-based tagger. ➤ They are deterministic. <p>Limitations:</p> <ul style="list-style-type: none"> ➤ The skill and effort required in writing disambiguation rules. ➤ Time is spent in writing a rule-set. ➤ It is usable for only one language. 	
4.a	<p>Explain probabilistic CYK algorithm. List any two problems associated with PCFG.</p> <ul style="list-style-type: none"> ➤ Checks whether a input string belongs to Context Free Grammar or not. ➤ It is applicable on Chomsky Normal Form ➤ CNF is of the form as follows: <div style="text-align: center;"> <p>A--> BC</p> <p>A--> w (w--> input words w= x1x2x3.....xn)</p> </div> ➤ Consider an example: <div style="text-align: center;"> <p>The girl wrote an essay</p> </div> 	10

Natural Language Processing- 18CS743

January 2022-2023

SCHEME and SOLUTION

```

Let  $w = w_1 w_2 w_3 w_4 \dots w_j \dots w_n$ 
and  $w_j = w_i \dots w_{i+j-1}$ 
// Initialization step
for  $i := 1$  to  $n$  do
  for all rules  $A \rightarrow w_i$  do
    chart  $[i, 1] = \{A\}$ 
// Recursive step
for  $j = 2$  to  $n$  do
  for  $i = 1$  to  $n-j+1$  do
    begin
      chart  $[i, j] = \emptyset$ 
      for  $k = 1$  to  $j-1$  do
        chart  $[i, j] := \text{chart}[i, k] \cup \{A \mid A \rightarrow BC \text{ is a production and } B \in \text{chart}[i, k] \text{ and } C \in \text{chart}[k+1, j]\}$ 
      end
    end
if  $S \in \text{chart}[1, n]$  then accept else reject
  
```

Figure 4.12 The CYK algorithm

Example: The girl wrote an essay

The Girl Wrote An Essay $n=5$ $n=|w|$

Step 5: $j=5$

Step 4: $j=4$

Step 3: $j=3$

Step 2: $j=2 \Rightarrow \text{length} = 2$

Step 1:

Indicates Derivations of entire sp. of S in present here, then it helps to CFN

GRAMMAR:

- $S \rightarrow NP VP$
- $VP \rightarrow V NP$
- $NP \rightarrow Det Noun$
- $NP \rightarrow Noun VP$
- $V \rightarrow write$
- $Noun \rightarrow girl$
- $Noun \rightarrow essay$

	$i \backslash j$	1	2	3	4	5
1	1	Der → The	Noun → Girl	Verb → Wrote	Der → An	Noun → Essay
2	1,2	NP → Der Noun	Noun VP → Noun Verb	NP → Der Noun		
3	1,2,3			VP → Verb NP		
4	1,2,3,4					
5	1,2,3,4,5					
		The	Girl	Wrote	An	Essay

STEP 2: The Girl or Girl Wrote or Wrote An or An Essay

2 Combinations: $t_{12} = t_{11} + t_{21}$ or $t_{21} = t_{21} + t_{31}$ or $t_{32} = t_{32} + t_{41}$ or $t_{42} = t_{42} + t_{51}$

STEP 3: The Girl Wrote or Girl Wrote An or Wrote An Essay

3 Combinations: $t_{13} = \text{The} + \text{Girl Wrote} \Rightarrow \text{Det} + \text{NP}$ or $\text{The Girl} + \text{Wrote} \Rightarrow \text{NP} + \text{V}$

STEP 4: The Girl Wrote An or Girl Wrote An Essay

4 Combinations: $t_{14} = \text{The Girl Wrote} + \text{An}$ or The Girl Wrote An or $\text{The Girl} + \text{Wrote An}$

STEP 5: The Girl Wrote An Essay

5 Combinations: $t_{15} = (\text{The Girl Wrote} + \text{An Essay})$ or $(\text{The Girl} + \text{Wrote An Essay})$

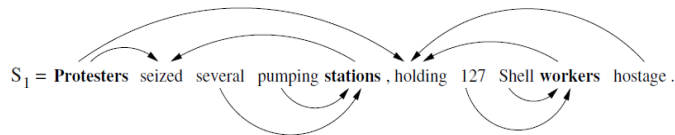
SCHEME and SOLUTION

	<ul style="list-style-type: none"> ➤ Its first problem lies in the independence assumption. We calculate the probability of a parse tree assuming that the rules are independent of each other. ➤ However, this is not true. How a node expands depends on its location in the parse tree. ➤ Example: Francis et al. (1999) showed that pronouns occur more frequently as subjects rather than objects. ➤ These dependencies are not captured by a PCFG, as the probability of, say, expanding an NP as a pronoun versus a lexical NP, is independent of whether the NP appears as a subject or an object. ➤ Another problem associated with a PCFG is its lack of sensitivity to lexical information. Lexical information plays a major role in determining correct parse in case of PP attachment ambiguities and coordination ambiguities. ➤ Two structurally different parses that use the same rules will have the same probability under a PCFG, making it difficult to identify the correct or most probable parse. ➤ The words appearing in a parse may make certain parses unnatural. This however, requires a model which captures lexical dependency statistics for different words. 	
<p>4.b</p>	<p>Write a note on:</p> <p>i. Phrase level construction</p> <ul style="list-style-type: none"> ✓ The constituents are identified by their ability to occur in similar contexts. ✓ One of the simplest ways to decide whether a group of words is a phrase is: ✓ to see if it can be substituted with some other group of words without changing the meaning. ✓ If such a substitution is possible then the set of words forms a phrase. ✓ This is called the substitution test. ✓ Example, Hena reads a book <p style="text-align: center;"> Hena reads a storybook Those girls read a book She reads a comic book </p> <ul style="list-style-type: none"> ✓ We can easily identify the constituents that can be replaced for each other in these sentences. ✓ These are hena, she, and Those girls and a book, a story book, and a comic book. ✓ These are the words that form a phrase. ✓ In linguistics, such constituents represent a paradigmatic relationship. ✓ Elements that can substitute each other in certain syntactic positions are said to be members of one paradigm. ✓ Phrase types are named after their head, which is the lexical category that determines the properties of the phrase. ✓ Thus, if head is noun, the phrase is called a noun phrase. 	

SCHEME and SOLUTION

	<p>ii. Sentence level construction</p> <ul style="list-style-type: none"> ✓ A sentence can have varying structure. ✓ The 4 commonly known structures are: <ul style="list-style-type: none"> * Declarative structure * Imperative structure * yes-no question structure * Wh-question structure 	
<p>5.a</p>	<p>Explain how relation pattern can be captured with string kernel.</p> <ul style="list-style-type: none"> ✓ The Blaschke and ELCS do relation extraction, where a rule is simply a sparse (gappy) subsequence of words or POS tags anchored on the two protein-name tokens. ✓ Therefore, the 2 methods have a common limitation, they end up using only a subset of all possible anchored sparse subsequences. ✓ Here, we exploit dual learning algorithms that process examples only via computing their dot-products, such as in Support Vector Machines (SVMs) . ✓ An SVM learner tries to find a hyperplane that separates positive examples from negative examples. ✓ The feature space is further pruned down by utilizing the following property of natural language statements: ✓ When a sentence asserts a relationship between 2 entity mentions, it generally does this using one of the following 4 patterns: <ol style="list-style-type: none"> 1. [FB] Fore–Between: words before and between the 2 entity mentions are simultaneously used to express the relationship. Examples: ‘interaction of P1 with P2,’ ‘activation of P1 by P2’ 2. [B] Between: only words between the two entities are essential for asserting the relationship. Examples: ‘P1 interacts with P2,’ ‘P1 is activated by P2’ 3. [BA] Between–After: words between and after the two entity mentions are simultaneously used to express the relationship. Examples: ‘P1 – P2 complex,’ ‘P1 and P2 interact.’ 4. [M] Modifier: the two entity mentions have no words between them. Examples: U.S. troops (a Role:Staff relation), Serbian general (Role:Citizen). 	<p>10</p>
<p>5.b</p>	<p>Explain shortest path hypothesis with example.</p> <ul style="list-style-type: none"> ✓ If entities e1 and e2 are arguments of the same predicate, then the shortest path between them will pass through the predicate: <ul style="list-style-type: none"> ◆ which may be connected directly to the two entities, or ◆ indirectly through prepositions. ✓ If e1 and e2 belong to different predicate-argument structures that share a common argument, then the shortest path will pass through this argument. ✓ Table shows the paths corresponding to 2 sentences from figure. 	<p>10</p>

SCHEME and SOLUTION



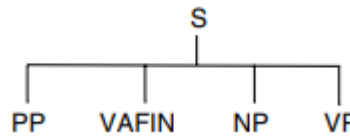
- For the first path, it is reasonable to infer that if a Person entity (e.g., ‘protesters’) is doing some action (e.g., ‘seized’) to a Facility entity (e.g., ‘station’), then the Person entity is Located at that Facility entity.
- The second path captures the fact that the same Person entity (e.g., ‘protesters’) is doing two actions (e.g., ‘holding’ and ‘seized’) , one action to a Person entity (e.g., ‘workers’), and the other action to a Facility entity (e.g., ‘station’).

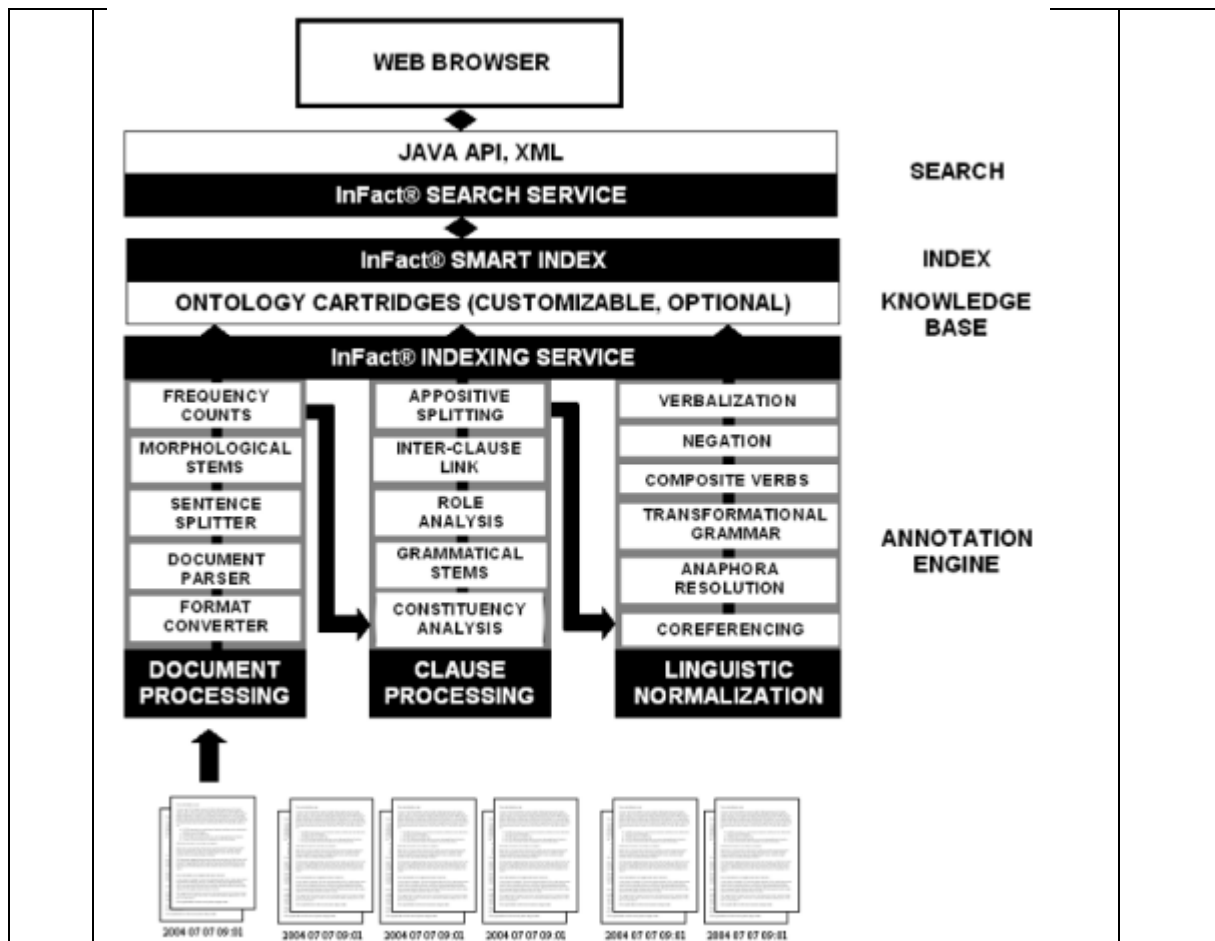
Table 3.1. Shortest Path representation of relations.

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

6a	<p>Explain the strategies used in active learning approach.</p> <ol style="list-style-type: none"> (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. In the following, the rationale behind choosing these steps is explained. <p>Steps a), b), c): In these steps, statistics about the syntactical structure of the corpus are created, with the intention of capturing its underlying distribution, so that representative instances for labeling can be selected. Step d): This step has been regarded as applicable to our corpus, due to the nature of the text. Our corpus contains repetitive descriptions of the same diagnostic measurements on electrical machines, and often, even the language used has a repetitive nature. Actually, this does not mean that the same words are repeated (although often standard formulations are used, especially in those cases when nothing of value was observed). Rather, the kind of</p>	10
----	--	----

SCHEME and SOLUTION

	<p>sentences used to describe the task has the same syntactic structure.</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>[PP Im Nutaustrittsbereich] wurden [NP stärkere Glimmentladungsspuren] festgestellt. <i>In the area of slot exit stronger signs of corona discharges were detected.</i></p> <p>[PP Bei den Endkeilen] wurde [NP ein ausreichender Verkeildruck] festgestellt. <i>At the terminals' end a sufficient wedging pressure was detected.</i></p> <p>[PP An der Schleifringbolzenisolation] wurden [NP mechanische Beschädigungen] festgestellt. <i>On the insulation of slip rings mechanical damages were detected.</i></p> <p>[PP Im Wickelkopfbereich] wurden [NP grossflächige Decklackablätterungen] festgestellt. <i>In the winding head area extensive chippings of the top coating were detected.</i></p> </div> <p>Examples of sentences with the same structure.</p> <div style="text-align: center; margin: 10px 0;">  <pre> graph TD S --> PP S --> VAFIN S --> NP S --> VP </pre> </div> <p>Step e): The committee of classifiers consists of a maximum entropy (MaxEnt) classifier from Mallet [19], a Winnow classifier from SNoW [2], and a memory-based learner (MBL) from TiMBL [6]. For the MBL, we selected k=5 as the number of the nearest neighbours. The classification is performed as follows: if at least two classifiers agree on a label, the label is accepted. If there is disagreement, the cluster of labels from the five nearest neighbours is examined. If the cluster is not homogenous (i.e., it contains different labels), the instance is included in the set of instances to be presented to the user for manual labeling.</p> <p>Step f): If one selects new sentences for manual annotation only based on the output of the committee-based classifier, the risk of selecting outlier sentences is high</p>	
<p>6b</p>	<p>Explain functional overview of InFact system with neat diagram.</p> <p>InFact consists of an indexing and a search module. Indexing pertains to the processing flow on the bottom of the diagram. InFact models text as a complex multivariate object using a unique combination of deep parsing, linguistic normalization and efficient storage. The storage schema addresses the fundamental difficulty of reducing information contained in parse trees into generalized data structures that can be queried dynamically. In addition, InFact handles the problem of linguistic variation by mapping complex linguistic structures into semantic and syntactic equivalents. This representation supports dynamic relationship and event search, information extraction and pattern matching from large document collections in real time.</p> <p>Indexing</p> <p>InFact's Indexing Service performs in order: 1) document processing, 2) clause processing, and 3) linguistic normalization.</p>	<p>10</p>



Functional overview of InFact

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.stellant.com), which can convert 370 different input file types. Our customized document parsers can process disparate styles and recognized zones within each document. 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, so the relevant section can be displayed in response to a query.

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.

Linguistic Normalization

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

	<p>distance, syntactic position, and head noun.</p>	
<p>7a</p>	<p>With a neat diagram explain explain the evolutionary mode for KDT</p> <div data-bbox="427 367 1201 741" data-label="Diagram"> </div> <p>The Evolutionary Model for Knowledge Discovery from Text</p> <p>The whole processing starts by performing the IE task which applies extraction patterns and then generates a rule-like representation for each document of the specific domain corpus. After processing a set of n documents, the extraction stage will produce n rules, each one representing the document’s content in terms of its conditions and conclusions. Once generated, these rules, along with other training data, become the “model” which will guide the GA-based discovery.</p> <p>KDT can potentially benefit from successful techniques from Data Mining or Knowledge Discovery from Databases (KDD) [14] which have been applied to relational databases. However, Data Mining techniques cannot be immediately applied to text data for the purposes of TM as they assume a structure in the source data which is not present in free text. Hence new representations for text data have to be used. Also, while the assessment of discovered knowledge in the context of KDD is a key aspect for producing an effective outcome, the evaluation/assessment of the patterns discovered from text has been a neglected topic in the majority of the KDT approaches. Consequently, it has not been proven whether the discoveries are novel, interesting, and useful for decision makers.</p> <p>Bag-of-Words-Based Approaches</p> <p>Initial information (i.e., terms, keywords) has been extracted, KDD operations can be carried out to discover unseen patterns. Representative methods in this context have included Regular Associations, Concept Hierarchies citeFeldman, Full Text Mining, Clustering, Self-Organizing Map</p> <p>High-Level Representation Approaches</p> <p>Another mainstream in KDT involves using more structured or higher-level representations to perform deeper analysis so to discover more sophisticated novel / interesting knowledge. Although in general, the different approaches have been concerned with either performing exploratory analysis for hypothesis formation or finding new connections/relations between previously analysed natural language knowledge, it has also involved using term-level knowledge for other purposes than just statistical analysis.</p>	

SCHEME and SOLUTION

7b	<p>Explain word matching feedback system.</p> <p>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. Important words were generally content words that were deemed important to the meaning of the sentence and could include words not found in the sentence. 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. In essence, the attempt was made to imitate LSA. A trainee’s explanation was analyzed by matching the words in the explanation against the words in the target sentence and words in the corresponding association lists.</p> <p>This was accomplished in two ways: (1) Literal word matching and (2) Soundex matching.</p> <p>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.</p> <p>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.’ Note that the later example was originally transformed to ‘c888368’ and two 8s were dropped (‘m’ and ‘n’ are both mapped to ‘8’). If the trainee’s self-explanation contains ‘thonderstorm’ or ‘tonderstorm,’ both will be matched with ‘thunderstom’ and this is called a soundex match. An exact soundex match is required for short words (i.e., those with fewer than six alpha-characters) due to the high number of false alarms when soundex is used. For longer words, a match on the first four soundex symbols suffices. We are considering replacing this rough and ready approach with a spell-checker. A formula based on the length of the sentence, the length of the explanation, the length criterion mentioned below, the number of matches to the important words, and the number of matches to the association lists produces a rating of 0 (inadequate), 1 (barely adequate), 2 (good), or 3 (very good) for the explanation. The rating of 0 or inadequate is based on a series of filtering criteria that assesses whether the explanation is too short, too similar to the original sentence, or irrelevant. 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. For example, if the length of the sentence is 10 words and the length priority is 1, then the required length of the self-explanation would be 10 words. If</p>	10
----	--	----

SCHEME and SOLUTION

	<p>the length of the sentence is 30 words and the length priority is 0.5, then the self-explanation would require a minimum of 15 words. 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. If the explanation is close in length to the sentence, with a high percentage of word overlap, the explanation would be deemed too similar to the target sentence. If the explanation failed any of these three criteria (Length, Relevance, and Similarity), the trainee would be given feedback corresponding to the problem and encouraged to revise the self-explanation. Once the explanation passes the above criteria, then it is evaluated in terms of its overall quality. The three levels of quality that guide feedback to the trainee are based on two factors: 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. This 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. However, because we envisioned a scaled-up system rapidly adaptable to many texts, we needed a system that required relatively little manual effort per text. Therefore, WB-ASSO was replaced. 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 [9, 20]. 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. While WB-ASSO is based on a richer corpus of words than WB-TT, the replacement was successful because the latter was intended for use together with LSA which incorporates the richness of a corpus of hundreds of documents. In contrast, WB-ASSO was used on its own. Some hand-coding remained in WB-TT because the length criterion for an explanation was calculated based on the average length of explanations of that sentence collected from a separate pool of participants and on the importance of the sentence according to a manual analysis of the text. Besides being relatively subjective, this process was time consuming because it required an expert in discourse analysis as well as the collection of self-explanation protocols. Consequently, the hand-coded length criterion was replaced with one that could be determined automatically from the number of words and content words in the target sentence (we called this wordbased with total text and automated criteria, or WB2-TT). The change from WB-TT to WB2-TT affected only the screening process of the length and similarity criteria. Its lower-bound and upper-bound lengths are entirely based on the target sentence' length. The overall quality of each self-explanation is still computed with the same formula used in WB-TT.</p>	
8a	<p>Define the following</p> <ol style="list-style-type: none"> a. Structure <p>Measures how much of the rules' structure is exhibited in the current</p>	

SCHEME and SOLUTION

	<p>hypothesis.</p> <p>b. 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.</p> <p>c. Interestingness How interesting is the hypothesis in terms of its antecedent and consequent: Unlike other approaches to measure “interestingness” which use an external resource (e.g., WordNet) and rely on its organisation, we propose a different view where the criterion can be evaluated from the semi-structured information provided by the LSA analysis.</p> <p>d. Coherence This metrics addresses the question whether the elements of the current hypothesis relate to each other in a semantically coherent way. Unlike rules produced by DM techniques in which the order of the conditions is not an issue, the hypotheses produced in our model rely on pairs of adjacent elements which should be semantically sound, a property which has long been dealt with in the linguistic domain, in the context of text coherence</p> <p>e. Coverage The coverage metric tries to address the question of how much the hypothesis is supported by the model</p>	
<p>8b</p>	<p>Write a short notes on:</p> <p>a. LSA</p> <p>b. 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’s practical advantage over shallow word overlap measures is that it goes beyond lexical similarities such as chair/chairs or run/ran, and manages to rate the relative semantic similarity between terms such as chair/table, table/wood, and wood/forest. As such, LSA does not only tell us whether two items are the same, it tells us how similar they are. Further, as Wolfe and Goldman report, there is substantial evidence to support the notion that the reliability of LSA is not significantly different from human raters when asked to perform the same judgments. As a measure of semantic relatedness, LSA has proven to be a useful tool in a variety of studies. These include computing ratings of the quality of summaries and essays, tracing essay elements to their sources, optimizing texts to-reader matches based on reader knowledge and projected difficulty of unread texts [53], and for predicting human interpretation of metaphor difficulty [28]. For this study, however, we adapted the LSA cohesion measuring approach used by Foltz, Kintsch & Landauer. Foltz and colleagues formed a representation of global cohesion by using LSA to analyze the relationship of ever distant textual paragraphs. As the distances increased, so the LSA score of similarity decreased. The results suggested that LSA was a useful and practical tool for measuring the relative degrees of similarity between textual sections.</p> <p>c. Sequence Model Formally, the procedure described above can be modeled as a Markov chain. Denoting the input sequence of ordered pages p_1, \dots, p_n by $P = (p_1, \dots, p_n)$ and the output sequence of document types by $D = (d_1, \dots, d_n)$, the probability of</p>	

SCHEME and SOLUTION

	<p>a specific sequence of document types D given the input sequence of pages can be written as $p(D P) = \prod_{j=1}^n p(d_j D_{j-1}, P)$, where d_j denotes the document type of the j-th page and 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, D_{j-1}, and P hold at some level of accuracy and allow estimations of the probability $p(D P)$ that are efficient yet accurate enough for the given purpose. We started by assuming that the document type d_j at time step j only depends on the page content pc_j 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 $p(D P) \approx \prod_{j=1}^n p(d_j pc_j)$ as well as the following approximation, which is very common and has been widely used in several fields, e.g., for information extraction, $p(D P) \approx \prod_{j=1}^n p(d_j d_{j-1}, pc_j)$ and finally $p(D P) \approx \prod_{j=1}^n p(d_j d_{j-1}, d_{j-2}, pc_j)$. (8.4) Instead of trying to approximate the probability of $p(D P)$ ever more accurately by relaxing the independence assumptions one also can describe pages in more detail by breaking up the document types based on the page position within a document. Functionally, this is achieved by altering the output language. In the extreme, this would lead to a model of the data in which the symbols of the output language are different for each page number within the document. You would have symbols like TaxForm1, TaxForm2, TaxForm3, etc., for the different page numbers within a tax form. Here, we increased the alphabet of the original output language threefold. 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 D, where D is given by $D = (d_1, \dots, d_n)$ with d_j denoting the document type as well as the page type. The definitions of the page type events {start, middle, end} are: start : $\{pc_j, t t = 1, t \leq l\}$ middle : $\{pc_j, t t > 1, t < l\}$ end : $\{pc_j, t t > 1, t = l\}$ where j is the global page number within the batch and t is the local page number within a document of length l. One of the models considered using the new output language is $p(D P) \approx \prod_{j=1}^n p(d_j pc_j)$, under the constraint that the sequence of page types is consistent with the definitions, e.g., every document has to end with the end page type with the exception of one-page documents. The last model has, owing to this constraint, many similarities with the model The main difference between the two models is that the model of determines boundaries between documents based on the previous document types, whereas the model of relies mainly on the difference of start, middle, and end pages within the document type to identify boundaries. Accordingly, the model of can separate subsequent instances of the same document type, whereas the model of cannot. Finally, we also tested models that conditioned the output symbol at a given time step not only on the content of the current page but also on the previous and the next $p(D P) \approx \prod_{j=1}^n p(d_j d_{j-1}, d_{j-2}, pc_{j-1}, pc_j, pc_{j+1})$ $p(D P) \approx \prod_{j=1}^n p(d_j pc_{j-1}, pc_j, pc_{j+1})$, where the model has the same constrained output language as the model i.e., an output language consistent with the definitions of the events {start, middle, end}.</p>	
--	---	--

SCHEME and SOLUTION

9a.	<p>Explain six criteria that can be used for evaluation of IR (Information Retrieval) system The success of an IR system may be judged by a range of criteria including relevance, speed, user satisfaction, usability, efficiency and reliability. However, the most important factor in determining a system's effectiveness for users is the overall relevance of results retrieved in response to a query. Explanation of these criterias.</p>	10					
9b	<p>Explain WORDNET and its application WordNet's IS-A hierarchy (which enumerates different kinds of properties) and morphological cues (e.g., "-iness", "-ity" suffixes). WordNet synonyms and antonyms in conjunction with a set of seed words in order to find actual opinion words. Finally, opinion words are used to extract associated infrequent features. The system only extracts explicit features. WordNet-Based and Web-Based Adjective Similarity Rules. Notation: $s_1, s_2 = \text{WordNet synsets}$</p> <table border="1" data-bbox="309 741 1289 902"> <tr> <td>adj_1 and adj_2 are similar if</td> </tr> <tr> <td>$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$</td> </tr> <tr> <td>$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), pertain(adj_2, s_2), isA(s_1, s_2)$</td> </tr> <tr> <td>$\exists s_1, s_2$ s.t. $attribute(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$</td> </tr> <tr> <td>$\exists p \in \{ "[X], even[Y]", "[X], almost[Y]", \dots \}$ s.t. $hits(p(adj_1, adj_2)) > t, t = \text{threshold}$</td> </tr> </table>	adj_1 and adj_2 are similar if	$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$	$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), pertain(adj_2, s_2), isA(s_1, s_2)$	$\exists s_1, s_2$ s.t. $attribute(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$	$\exists p \in \{ "[X], even[Y]", "[X], almost[Y]", \dots \}$ s.t. $hits(p(adj_1, adj_2)) > t, t = \text{threshold}$	
adj_1 and adj_2 are similar if							
$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$							
$\exists s_1, s_2$ s.t. $pertain(adj_1, s_1), pertain(adj_2, s_2), isA(s_1, s_2)$							
$\exists s_1, s_2$ s.t. $attribute(adj_1, s_1), attribute(adj_2, s_2), isA(s_1, s_2)$							
$\exists p \in \{ "[X], even[Y]", "[X], almost[Y]", \dots \}$ s.t. $hits(p(adj_1, adj_2)) > t, t = \text{threshold}$							
10	<p>Write a short note on:</p> <ol style="list-style-type: none"> Indexing <div data-bbox="402 1025 1321 1554" style="background-color: #e0e0e0; padding: 10px;"> <p>In a small collection of documents, an IR system can access a document to decide its relevance to a query. However, in a large collection of documents, this technique poses practical problems. Hence, a collection of raw documents is usually transformed into an easily accessible representation. This process is known as indexing. Most indexing techniques involve identifying good document descriptors, such as keywords or terms, which describe the information content of documents. A good descriptor is one that helps describe the content of the document and discriminate it from other documents in the collection. Luhn (1957, 1958) is considered the first person to advance the notion of automatic indexing of documents based on their content. He assumed that the frequency of certain word-occurrences in an article gave meaningful</p> </div> Eliminating stop words The lexical processing of index terms involves elimination of stop words. Stop words are high frequency words which have little semantic weight and are thus, unlikely to help in retrieval. Stemming Stemming normalizes morphological variants, though in a crude manner, by removing affixes from the words to reduce them to their stem, eg. The words compute, computing, computes, and computer are all be reduced to same word stem, comput. Zipf's Law Zipf made an important observation on the distribution of words in natural languages. This observation is known as Zipf's law. The frequency of words multiplied by their ranks in a large corpus is more or less constant. Frequency * Rank = constant. 						

Natural Language Processing- 18CS743
January 2022-2023

SCHEME and SOLUTION

