**SRM Institute of Science and Technology**

**College of Engineering and Technology**

**School of Computing**

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu

**Academic Year: 2024 - 25 EVEN**

**B.Tech-School of Computing**

**Test: FT2** **Date: 21.2.2025**

**Course Code & Title****: 21CSE356T – Natural Language Processing Duration: 2 periods**

**Sem: VI Sem** **Max. Marks: 50**

**Batch II SET : D**

ANSWER KEY

1.D

2.D

3.A

4.B

5.B

6.D

7.C

8.A

9.B

10.B

11.A **1.Tokenization:** The first step is to break the sentence into individual words or tokens: "The," "thief," "robbed," "the," "apartment." (1 mark)

**2.Part-of-Speech Tagging:** Each token is assigned a part of speech: (1 mark)

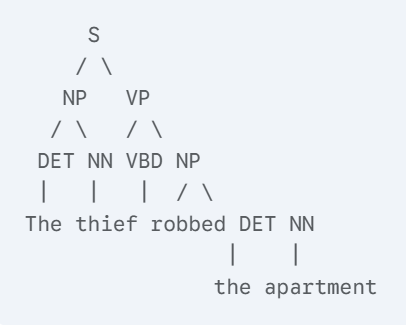
* + The (Determiner - DET)
  + thief (Noun - NN)
  + robbed (Verb - VBD - past tense)
  + the (Determiner - DET)
  + apartment (Noun - NN)

**3.Parsing (Syntactic Analysis):** This is where the tree structure is built. A simplified approach (Context-Free Grammar) might look like this: (3 marks)

* + **Sentence (S) -> Noun Phrase (NP) + Verb Phrase (VP)**
  + **NP -> DET + NN**
  + **VP -> VBD + NP**

Applying these rules, the parser groups the tokens:

* **NP:** "The thief" (DET + NN)
* **VP:** "robbed the apartment" (VBD + NP)
* **NP:** "the apartment" (DET + NN)



11.B TF-IDF stands for **Term Frequency-Inverse Document Frequency**. It's a statistical measure used to evaluate the importance of a word (or term) within a document that is part of a larger collection of documents (a corpus). It's a crucial technique in text analysis and information retrieval.

Here's a breakdown of the two components and how they combine:

**1. Term Frequency (TF): (1 mark)**

* **Definition:**  How often a term appears in a particular document.
* **Calculation:** Several ways exist to calculate TF, but a common approach is: TF(term, document) = (Number of times term appears in document) / (Total number of terms in document)
* **Intuition:** A term that appears frequently in a document is likely to be more important to that document's topic.

**2. Inverse Document Frequency (IDF): (1 mark)**

* **Definition:** How rare a term is across the entire corpus of documents.
* **Calculation:** A common formula is: IDF(term) = log((Total number of documents in corpus) / (Number of documents containing term))
* **Intuition:** A term that appears in many documents is less informative than a term that appears in only a few. Common words like "the," "a," or "is" will have low IDF values because they appear in almost every document. Rare words related to a specific topic will have high IDF values.

**3. TF-IDF: (3 mark)**

* **Calculation:** TF-IDF(term, document) = TF(term, document) \* IDF(term)
* **What it measures:** The TF-IDF score combines the term frequency and inverse document frequency. A high TF-IDF score indicates that a term is frequent in a specific document but rare across the corpus. This suggests that the term is important to that document's meaning.

**Applications:**

* **Information Retrieval**
* **Text Summarization**
* **Text Classification**
* **Keyword Extraction**
* **Document Clustering**

**Example:**

Let's say we have a corpus of two documents:

* **Document 1:** "The cat sat on the mat."
* **Document 2:** "The dog sat on the floor."

The word "cat" appears once in Document 1 and zero times in Document 2. It's also relatively rare across the corpus. Therefore, it will have a high TF-IDF score in Document 1, indicating its importance to that document. The word "the" appears in both documents and is common, so it will have a low IDF and therefore a low TF-IDF score in both documents, reflecting its low importance.

12.A. **1. Lexicon: (2 mark)**

* **Definition:** A lexicon is essentially a dictionary or a mental repository of words and their associated information in a language. It's not just a list of words; it includes:
  + **Forms:** The different forms a word can take (e.g., "run," "running," "runs," "ran").
  + **Meanings:** The various meanings a word can have (e.g., "bank" as a financial institution or a river bank).
  + **Pronunciation:** How the word is pronounced.
  + **Part of Speech:** The grammatical category of the word (e.g., noun, verb, adjective).
* **In NLP:** A computational lexicon is a data structure used by NLP systems to store and access this lexical information. It's crucial for tasks like tokenization, stemming, lemmatization, part-of-speech tagging, and word sense disambiguation.

**2. Lexeme: (1mark)**

* **Definition:** A lexeme is the abstract form of a word that underlies its different inflected forms. It's the base form of a word that carries the core meaning. Think of it as the dictionary entry for a word.
* **Example:** The lexeme "run" encompasses all its forms: "run," "running," "runs," and "ran." These different forms are called "word forms" or "surface forms," while "run" is the underlying lexeme.

**Relationships Between Lexemes: ( 2 mark)**

Lexemes in a lexicon are interconnected through various semantic relations. Understanding these relations is essential for NLP tasks like semantic analysis and knowledge representation. Here are some key types:   (Any 2)

* **Synonymy:** Lexemes with the same or very similar meanings (e.g., "happy" and "joyful"). Synonyms can be near-synonyms, with slight differences in connotation or usage.
* **Antonymy:** Lexemes with opposite meanings (e.g., "hot" and "cold"). Antonyms can be complementary (either/or), contrary (a scale between), or antiparallel (reverse direction).
* **Hyponymy (and Hypernymy):** Hyponymy is an "is-a" relationship. A hyponym is a more specific type of a hypernym (the broader category). For example, "dog" is a hyponym of "animal" (its hypernym). "Animal" is the hypernym of "dog."
* **Meronymy (and Holonymy):** Meronymy is a "part-of" relationship. A meronym is part of a holonym (the whole). For example, "finger" is a meronym of "hand" (its holonym). "Hand" is the holonym of "finger."
* **Homonymy:** Lexemes that have the same spelling or pronunciation but different meanings (e.g., "bank" as a financial institution or a river bank). Homonyms can be homographs (same spelling) or homophones (same pronunciation).
* **Polysemy:** A single lexeme with multiple related meanings (e.g., "run" can mean to move quickly on foot, to manage a business, etc.). Polysemy is different from homonymy because the meanings are related.
* **Troponymy:** A troponym describes a more specific manner of doing something. For example, "walk" is a troponym of "move," and "stroll" is a troponym of "walk."
* **Entailment:** One lexeme entails another if the meaning of the first implies the meaning of the second. For example, "kill" entails "die."

12.B. Both Bag-of-Words (BOW) and n-gram models are techniques used in Natural Language Processing (NLP) to represent text in a numerical way that machine learning models can understand. However, they differ significantly in how they capture the information within the text:

**Bag-of-Words (BOW) (2.5 marks)**

* **Definition:** Represents text as an unordered collection of words (or tokens), disregarding grammar and word order.
* **Representation:** Creates a vocabulary of all unique words in the corpus (collection of documents). Each document is then represented as a vector where each element corresponds to a word in the vocabulary, and the value represents the frequency of that word in the document.
* **Example:**
  + Sentence 1: "The cat sat on the mat."
  + Sentence 2: "The mat is black."
  + BOW representation:
    - Sentence 1: {the: 2, cat: 1, sat: 1, on: 1, mat: 1}
    - Sentence 2: {the: 1, mat: 1, is: 1, black: 1}
* **Advantages:**
  + Simple to implement.
  + Computationally efficient.
  + Can be effective for tasks like document classification and topic modeling.
* **Disadvantages:**
  + Ignores word order, losing important contextual information.
  + Doesn't capture semantic relationships between words.

**N-gram Models**

* **Definition:** Captures sequences of n adjacent words (or tokens) in the text, preserving some word order information.
* **Representation:** Creates a vocabulary of all unique n-grams in the corpus. Each document is represented as a vector where each element corresponds to an n-gram, and the value represents the frequency of that n-gram in the document.
* **Example:**
  + Sentence 1: "The cat sat on the mat."
  + Sentence 2: "The mat is black."
  + Bigram (2-gram) representation:
    - Sentence 1: {the cat: 1, cat sat: 1, sat on: 1, on the: 1, the mat: 1}
    - Sentence 2: {the mat: 1, mat is: 1, is black: 1}
* **Advantages:**
  + Captures some word order information, improving performance in tasks like language modeling and machine translation.
  + Can be more effective than BOW for capturing local context.
* **Disadvantages:**
  + Vocabulary size can become very large for higher-order n-grams (e.g., trigrams, 4-grams), leading to computational challenges and data sparsity issues.
  + Still doesn't fully capture long-range dependencies in the text.

**Key Differences Summarized:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Bag-of-Words (BOW)** | **N-gram Models** |
| Word Order | Ignored | Partially preserved |
| Context | Limited | More context captured |
| Vocabulary Size | Smaller | Larger (especially for higher-order n-grams) |
| Computation | More efficient | More computationally intensive |
| Applications | Document classification, topic modeling | Language modeling, machine translation |