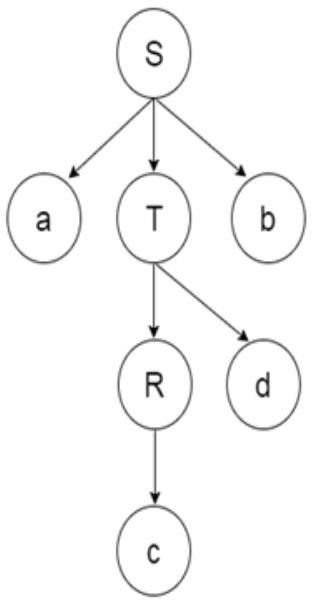
1. Parser

* Parser is a compiler that is used to break the data into smaller elements coming from lexical analysis phase.
* A parser takes input in the form of sequence of tokens and produces output in the form of parse tree.
* Parsing is of two types: top down parsing and bottom up parsing. Top down paring
* The top down parsing is known as recursive parsing or predictive parsing.
* Bottom up parsing is used to construct a parse tree for an input string.
* In the top down parsing, the parsing starts from the start symbol and transform it into the input symbol.

Parse Tree representation of input string "acdb" is as follows:

WORKING OF TOP DOWN PARSER:

* In top down technique parse tree constructs from top and input will read from left to right. In top down, In top down parser, It will start symbol from proceed to string.
* It follows left most derivation.
* In top down parser, difficulty with top down parser is if variable contain more than one possibility selecting 1 is difficult.

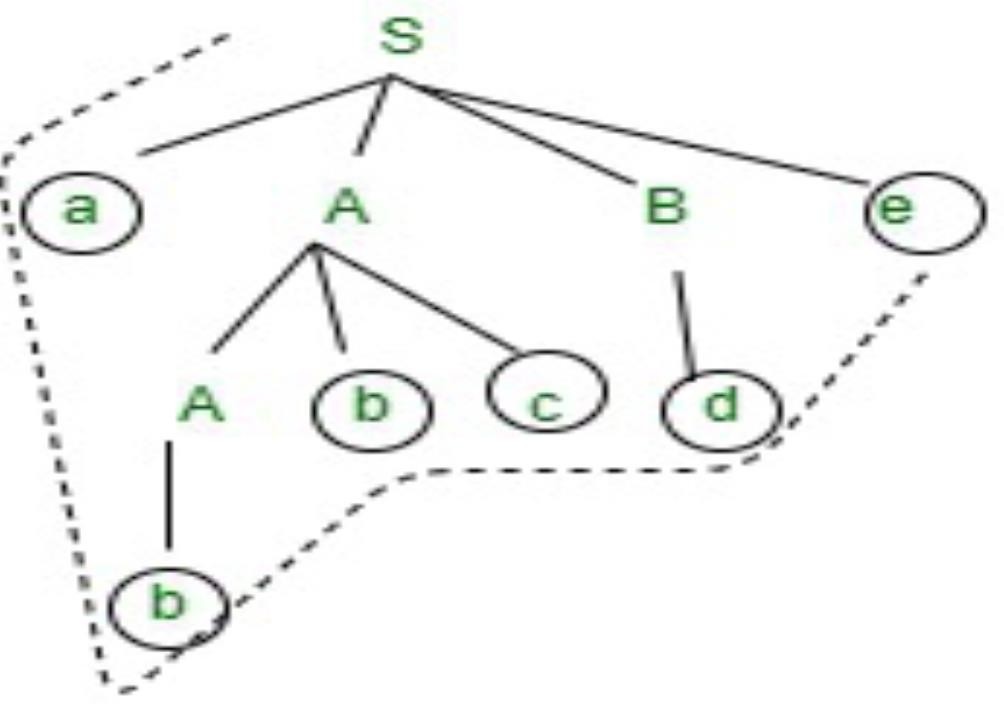
Working of Top Down Parser :

Let’s consider an example where grammar is given and you need to construct a parse tree by using top down parser

technique. Example – S -> aABe

A -> Abc | b B -> d

* Now, let’s consider the input to read and to construct a parse tree with top down approach.
* Input – abbcde
* generate a input string from the grammar for top down approach.
* First, you can start with S -> a A B e and then you will see input string a in the beginning and e in the end.
* Now, you need to generate abbcde .
* Expand A-> Abc and Expand B-> d.
* Now, You have string like aAbcde and your input string is abbcde.
* Expand A->b. Final string, you will get abbcde.



2. Language Modelling:

* Language modelling is a fundamental concept in natural language processing (NLP) that involves developing computational models to predict the likelihood of a sequence of words occurring in each language.
* It aims to capture the structure, grammar, and context of language to generate coherent and contextually appropriate text.
* Language models are primarily of two kinds: N-Gram language models and Grammar-based language models such as probabilistic context-free grammar.

3. Statistical Language Modelling:

* Statistical language modelling is a specific approach to language modelling that employs statistical techniques to estimate the probabilities of word sequences based onobserved data.
* It relies on the assumption that the probability of a word occurring depends on the preceding words in the sequence.
* Statistical language models use traditional statistical techniques like N-grams, Hidden Markov Models (HMM).

4. Neural Language Modelling:

Neural Language Models use different kinds of approaches like neural networks such as feedforward neural networks, recurrent neural nets, attention-based networks, and transformers-based neural nets late to model the language, and they have also surpassed the statistical language models in their effectiveness.

5. N-gram Models

• N-gram models are a particular set of language models based on the statistical frequency of groups of tokens.

• An n-gram is an ordered group of n tokens.

• The bigrams of the sentence. The cat eats fish. are (The, cat), (cat, eats), (eats, fish) and (fish, .).

• The trigrams are (The, cat, eats), (cat, eats, fish) and (eats, fish, .).

• The smallest n-grams with n =1 are called unigrams. Unigrams are simply the tokens appearing in the sentence.

• The conditional probability that a certain token appears after previous tokens are estimated by Maximum Likelihood Estimation on a set of training sequences.

6. Intuitive Formulation:

• The intuitive idea behind n-grams and n-gram models are that instead of computing the probability of a word given its entire history, we can approximate the history by just the last few words like humans do while understanding speech and text.

Illustration for N-gram probabilities:

P(wn ∣w1 …wn−1 )≈P(wn ) unigram

P(wn ∣w1 …wn−1 )≈P(wn ∣wn−1 )bigram

P(wn ∣w1 …wn−1 )≈P(wn ∣wn−1 wn−2 ) trigram

P(wn ∣w1 …wn−1 )≈P(wn ∣wn−1 wn−2 wn−3 ) 4gram

P(wn ∣w1 …wn−1 )≈P(wn ∣wn−1 wn−2 wn−3 wn−4 ) 5-gram

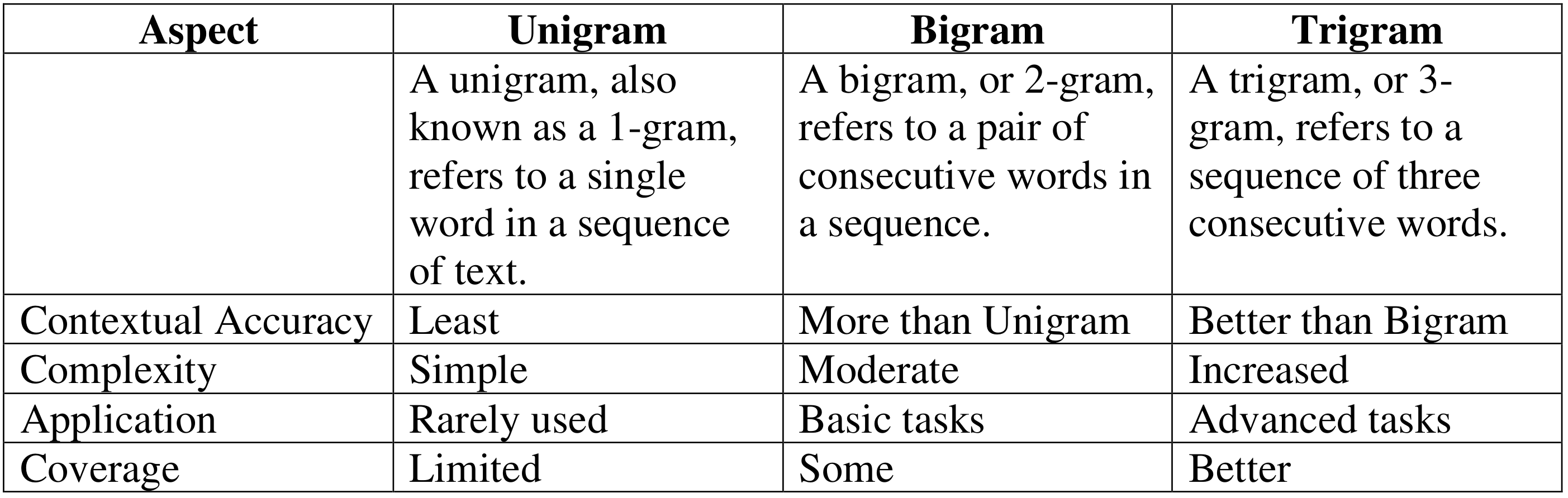
Choice of N in N-Gram models:

• The accuracy of the model increases with an increase in N.

• But with bigger N values, we run into the risk that we may not get good estimates for N-Gram probabilities, and the N-Gram tables will be more sparse.

• Smaller the N, the model will be less accurate. But we may get better estimates for N-

Gram probabilities, and the N-Gram tables will be less sparse.

  
7. Probability Estimation:

There are two main steps generally in building a machine learning model: Defining the model and Estimating the model’s parameters, which is called the training or the learning step.

• There are also two quantities we need to estimate for developing the language models for all words in the vocabulary.

• For example, Pr(Colorless green ideas sleep furiously)

8. Maximum Likelihood estimation:

• The most basic parameter estimation technique is the relative frequency estimation (frequencies are counts) which is also called the method of Maximum Likelihood Estimation (MLE).

• The estimation simply works by counting the number of times the word appears conditioned on the sentence and then normalizing the probabilities. We also need some source text corpora.

• Chain rule of probability in estimation: To estimate the probabilities, we usually rely on the Chain Rule of Probability, where we decompose the joint probability into a product of conditional probabilities using the independence assumption.

• It is also to be kept in mind that estimating conditional probabilities with long contexts is usually difficult, and for example, conditioning on 4 or more words itself is very hard.

9. Markov assumption in probability estimation:

• The use of Markov assumption in probability estimation solves a lot of problems we encounter with data sparsity and conditional probability calculations.

• The assumption is that the probability of a word depends only on the previous word(s). It is like saying the next event in a sequence depends only on its immediate past

context.

• Markov models are the class of probabilistic models that assume that we can predict the probability of some future unit without looking too far into the past.

10. Role of Conditional Probability:

Conditional probability is a fundamental concept in probability theory that quantifies the likelihood of an event occurring given that another event has already occurred.   
In other words, it measures the probability of event B happening, considering the prior occurrence of event A. Conditional probability is denoted by P(B|A), where P(B) represents the probability of event B and P(A) represents the probability of event A.Conditional probability has numerous applications across various domains:

* **Bayesian Filtering**
* **Speech Recognition**
* **Medical Diagnosis**
* **Recommendation Systems**

**11. Two main parts of semantic analysis**

#### Lexical semantics

Lexical semantics is the study of the meaning of individual words. It is concerned with the relationships between words, such as synonymy, antonymy, and hyponymy. Lexical semantics also includes the study of the internal structure of words, such as their morphemes and their grammatical categories.

**Examples of lexical semantics:**

* **Synonymy:** "Big" and "large" are synonyms.
* **Antonymy:** "Hot" and "cold" are antonyms.
* **Hyponymy:** "Dog" is a hyponym of "animal".
* **Morphemes:** The word "unhappy" is composed of the morphemes "un-", "happy", and "-y".
* **Grammatical categories:** The word "walk" is a verb.

#### Compositional semantics

Compositional semantics is the study of how the meanings of individual words combine to form the meaning of a sentence. It is concerned with the rules that govern how words are interpreted in context.

**Examples of compositional semantics:**

* **The sentence "The dog chased the cat" means that the dog was running after the cat.**
* **The sentence "The cat sat on the mat" means that the cat was positioned on the mat.**

****12. Morphological models** are computational approaches that describe and analyze the internal structure of words and how they are formed. They are used in a variety of natural language processing (NLP) tasks, such as machine translation, speech recognition, and text generation.**

**13. Morphological typology** is the study of the different types of morphological processes that exist in languages. It is concerned with the classification of languages based on their morphological features, such as the use of affixes, compounding, and reduplication.

**14. Parse Tree Representation:**

**A parse tree, also known as a syntax tree or derivation tree, is a tree structure that represents the syntactic structure of a sentence or other linguistic expression. It shows how the words or other units of meaning in the expression are related to each other according to the grammar of the language.**

**Structure of a Parse Tree**

A parse tree consists of nodes and edges. The nodes represent the words or other units of meaning in the expression, and the edges represent the grammatical relationships between them. The root node of the tree represents the entire expression, and the children of a node represent its constituent parts.

**Example of a Parse Tree**

The following is an example of a parse tree for the sentence "The boy ate the apple":

S

|-- NP --> DET N

| |-- DET --> The

| |-- N --> boy

|-- VP --> V NP

| |-- V --> ate

| |-- NP --> DET N

| |-- DET --> the

| |-- N --> apple

15. Smoothing

**In natural language processing (NLP), smoothing is a technique used to assign probabilities to unseen events or outcomes. It is an important technique for improving the accuracy of language models, which are used to predict the next word in a sequence or to translate text from one language to another.**

**The Problem of Data Sparsity**

The problem of data sparsity arises when there is not enough data to estimate the true probability of an event. For example, in a language model, there may not be enough training data to estimate the probability of a rare word appearing in a sentence. This can lead to inaccurate predictions, as the model may assign a very low probability to the rare word, even if it is the correct word in the context of the sentence.

**How Smoothing Works**

Smoothing works by redistributing some of the probability mass from more frequent events to less frequent events. This is done by adding a small amount of probability to each unseen event. The amount of probability that is added is typically proportional to the frequency of the event.

**Types of Smoothing**

There are many different types of smoothing, but some of the most common include:

* **Laplace smoothing:** Laplace smoothing adds one count to each event, regardless of whether it has been seen in the training data or not. This has the effect of making all events have a non-zero probability.
* **Good-Turing smoothing:** Good-Turing smoothing is a more sophisticated technique that takes into account the frequency of different events. It assigns more probability to events that have been seen a few times than to events that have never been seen.

### **16. CYK Algorithm in NLP**

The CYK (Cocke–Younger–Kasami) algorithm is a parsing algorithm used in natural language processing (NLP) for context-free grammars. It is particularly efficient for parsing sentences and determining their syntactic structure based on a given context-free grammar.

#### Algorithm:

The CYK algorithm operates by filling up a table in a bottom-up manner, where each cell represents a potential constituent (substring) of the input sentence. The algorithm leverages dynamic programming to efficiently compute the possible parse structures for the given sentence.

**Initialization:**

* Initialize a table with dimensions (n+1) x (n+1), where n is the length of the input sentence.
* Each cell (i, j) in the table represents the non-terminals that can generate the substring from position i to j in the sentence.

**Filling the Table:**

* Iterate through the input sentence, considering all possible combinations of non-terminals that could generate each substring.
* For each cell (i, j), consider all possible split points k such that i ≤ k < j.
* Check if there are non-terminals A and B such that A → BC is a production rule and B is in cell (i, k) and C is in cell (k, j).
* If such non-terminals are found, add A to cell (i, j).

**Backtracking:**

* After completing the table, perform backtracking to reconstruct the parse tree.
* Starting from the top-right cell (0, n), find non-terminals that can generate the entire sentence.
* Recursively follow the rules backward to construct the parse tree.

#### Working:

The CYK algorithm efficiently explores the space of possible parse structures by avoiding redundant computations. It breaks down the problem of parsing a sentence into smaller subproblems, allowing for a dynamic programming approach. The algorithm's time complexity is O(n^3), where n is the length of the input sentence.

#### Example:

Consider the following context-free grammar:

S → NP VP

NP → Det N

VP → V NP

Det → the

N → cat | dog

V → chased | caught

And the sentence: "the cat chased the dog."

**Initialization:**

Initialize a table with dimensions 6x6.

**Filling the Table:**

Iterate through the sentence, considering all possible combinations of non-terminals for each substring.

**Backtracking:**

* Starting from cell (0, 6), find non-terminals that can generate the entire sentence.
* Recursively follow the rules backward to construct the parse tree.

The resulting parse tree might look like:

S

/ \

NP VP

/ \ |

Det N V

| | |

the cat chased

This parse tree represents a valid syntactic structure for the given sentence according to the context-free grammar. The CYK algorithm efficiently determined this structure through dynamic programming.