



# Enhancing Statistical Language Modelling and Lexical Analysis Using Sanskrit's Linguistic Framework

Namrata Tapaswi<sup>1</sup>

Department of CSE (AIML),

Acropolis Institute of Technology and Research, Indore, M.P. India

namrastapaswi@gmail.com

**Abstract.** Many Indian languages can trace their ancestry back to Sanskrit, the oldest language known to man. It is essential to represent information in a native language since the linguistically divided population is adjusting to new technologies. This has prompted several forms of study in the field of natural languages aimed at improving the process of translating spoken or written languages into English. It is easier to map translation procedures for other dialects or language understanding algorithms when Sanskrit is inherited with the linguistic hierarchy of grammar formulation. The extensive vocabulary of Sanskrit makes its grammatical validations and lexical analyses reliable research. Sanskrit grammar made it easy to evaluate vernacular languages from a semantic perspective by analyzing their morphology and lexicon. The current approaches to statistical language modeling using Sanskrit are summarized in this article. This paper lays out the entire procedure for extracting expressions and rationally deducing phrase grammar rules. Statistical modelling theories are the main subject of the article, which also offers suggestions for improving the grammar's precision.

**Keywords:** Lexical, Syntactic, Semantic, Government and Binding (GB), Lexical Functional Grammar (LFG), Morphology, Context Free Grammar (CFG).

## 1 Introduction

Models are used to describe anything from simple objects to elaborate systems. Thus, a language model is an explanation of language. Natural language is a complicated entity and must create a representation (model) [1] of it to process it through a computer-based program. Language modelling is the term used for this. Language modelling can be seen as either a problem of probability estimates or a challenge of inferring grammar [3]. A probabilistic language model seeks to identify a sentence based on a probability measure, typically a maximum likelihood estimate, as opposed to a grammar-based language model, which seeks to discriminate between grammatical and non-grammatical (ill-formed) sentences. The following classification of language modelling approaches is the result of these two points of view.

### Language Model Based on Grammar

A grammar-based method builds its model on the grammar of a language. It tries to capture the syntactic organization of language [2]. The structure and arrangement of the many constituents appearing in a linguistic unit are defined by hand-coded rules known as grammar (phrase, sentence, etc.) [4]. A sentence, for instance, typically

© The Author(s) 2025

S. Bhalerao et al. (eds.), *Proceedings of the International Conference on Recent Advancement and Modernization in Sustainable Intelligent Technologies & Applications (RAMSITA-2025)*, Advances in Intelligent Systems Research 192,

[https://doi.org/10.2991/978-94-6463-716-8\\_78](https://doi.org/10.2991/978-94-6463-716-8_78)

comprises a verb phrase and a noun phrase. These structures, as well as the connections between these components, are both used in the grammar-based method.

### **Modelling of statistical language**

By using training data from a corpus, the statistical technique develops a language model. The training corpus must be sufficiently large to capture linguistic regularities. Rosenfeld emphasized:

“To enhance the functionality of diverse natural language applications, statistical language modelling (SLM) tries to accurately reproduce the patterns of natural language.”

One of the core objectives in many NLP applications, which are speech recognition, spelling correction, handwriting recognition, and machine translation, is statistical language modelling [5,6]. As of late, it has also found use in text summarization, question answering, and information retrieval. Literature has put forth a variety of statistical language models. The n-gram mode is the most widely used of them.

## **2 Related Work**

The localized visuals on internet sites that include regionally relevant language depend greatly on language processing. Due to the prevalent "English-only" environment, there are more technological advancements in the English language. The Indian context has opened places to promote historical and distinctive features of languages by posting Sanskrit-based texts on the internet to highlight their quality and diversity. It demonstrates how Sanskrit interpreters and processors will provide insight.

Future developments include the use of language modelling for XML retrieval, cluster-based language models, the use of language modelling for information retrieval, and the function of semantics within the scope of language modelling [1,2]. This research does a thorough investigation of neural network language modelling (NNLM). Different basic neural network language model designs are explained and looked at [3]. The usage of distributions that are specifically tailored for backing off is suggested [4,5]. For the first time, investigate how elements like the amount of training data, the corpus, and the order of the n-grams (bigram versus trigram) affect how well these algorithms perform in comparison [6]. assess a collection of Document-Context Language Models (DCLM), multi-level recurrent neural network language models that take context into account both within and outside of the sentence [7].

A linguistic theory called Lexical Functional Grammar (LFG) has been established with equal emphasis on theoretical linguistic and computer processing issues [8]. The f-structure is a recursive attribute-value matrix that captures complete enumeration relations, whereas the C-structure is a phrase-structure tree that acts as a foundation for phonological analysis [9]. A common parse forest serves as a representation for the collection of analyses the parser generates. The backbone's generated products from this parse forest can be regarded as a CFG in and of itself [10,11]. LFG is ideally

suitable for usage in the system's analyzing and mistake appreciation modules as well as to give students thoughtful feedback [13,14]. The use of a (base) verb and a particle together determines the interpretation and argument structure of particle verbs [[16,17]. Hindi, an Indo-European language, and English's linguistic differences can be seen as symbolizing the differences between the SOV and SVO classes of languages [18].

### **3 Many Language Models Based On Grammar**

Numerous computational grammars have been proposed and investigated. This paper introduces numerous methods for comprehending a language in a grammatical and rule-based style [7] with a focus on lexical functional grammar (LFG), generalized phrase structure grammar (GPSG), government and binding (GB), and Paninian grammar (PG). Additionally, it teaches the popular techniques for developing grammar and language statistical models.

#### **3.1 Generative Grammars**

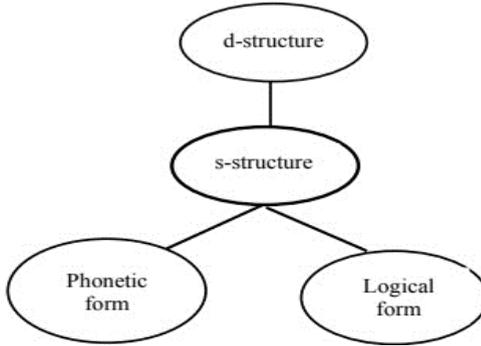
Noam Chomsky stated in 1957 in his book on syntactic structures that if familiar with a set of vocabulary and grammar rules for a language, then may produce sentences in that language [8]. Only sentences that can be generated according to the criteria are grammatical. This viewpoint, which is correctly known as generative grammar, has dominated computational linguistics. A language model can be created using the same concept [9]. A comprehensive set of rules that can produce every phrase conceivable in a language serves as a model for that language.

Language is the connection between words and sounds. Therefore, it is essential that grammatical meaning be considered in any model of a language [10]. As it is known, a sentence can be entirely grammatically correct but have no significance. This paper shall treat grammar as though it were a special case of language models.

#### **3.2 Government and Binding (GB)**

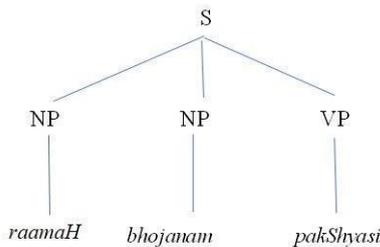
Some linguists (albeit not computational linguists) believe that a language's structure (or how effectively individual phrases are constructed) may be comprehended at the level of its interpretation, which is especially helpful when trying to resolve morphological ambiguity. The phrases are presented syntactically, though, thus it is difficult to see how meaning changes as syntax changes or vice versa.

The assumption behind transformational grammar is that sentences exist on two levels: the surface level and the deep root level. They were redesignated as s-level and d-level by government and binding (GB) theories, which also categorize further parallel representational levels, phonetic form, and logical form. Figure 1 shows the different levels at which GB theories can be used to look at language.



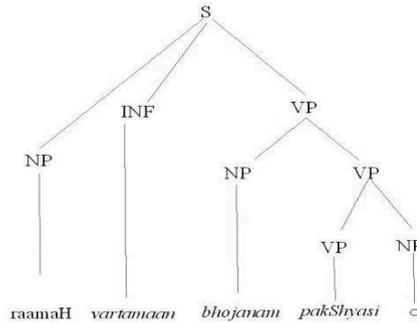
**Fig. 1.** Levels in Government and Binding

There are numerous rewriting rules in transformational grammar, which are mostly constructed and language-specific (in Sanskrit, many rules for assertive and interrogative sentences, or active and passive voice sentences) [11,18]. The ability to generate an entire set of cohesive rules might not be feasible. According to the GB, every language can be generated with fewer rules if structural unit rules are defined at a deep level. All languages have these deep-level structures, which are generalizations of noun-phrase, and verb-phrase. Because the human mind is "hard-wired" with certain universal grammar rules, it is conceivable to achieve if, as the GB theory suggests, a kid learns its mother tongue. Thus, as the information enters the mind, actual phonetic structures are created from its abstract structure. The primary focus of GB theories is the presence of deep-level, language-independent, abstract structures and their manifestation in surface-level, language-specific structures with the aid of straightforward principles. Let us use an illustration to clarify d- and s-structures. Consider the sentence: *raamaH bhojanam pakShyasi* in figure 2.

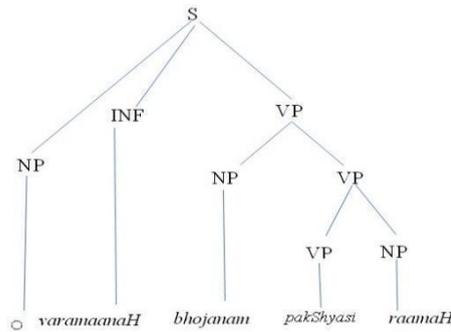


**Fig. 2.** Transformational grammar representation of *raamaH bhojanam pakShyasi*

In Government and Binding, the s-structure and d-structure are as figure 3 and figure 4 respectively.



**Fig. 3.** s-structure representation of *raamaH bhojanam pakShyasi*



**Fig. 4.** d-structure representation of *raamaH bhojanam pakShyasi*

### X-bar Theory

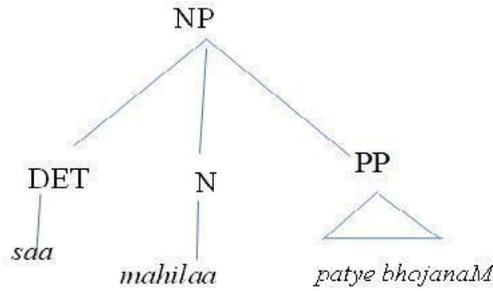
One of the fundamental ideas of GB is the X theory, which is also known as the X-bar theory. The sentence structure and various phrase structures are both described by X theory as an optimum projection of some head rather than being defined by independent rules and standards [12]. The described entities become linguistically autonomous in this way. This means that the heads of noun phrases (NP), verb phrases (VP), adjective phrases (AP), and prepositional phrases (PP) are all optimum projections of their associated nouns (N), verbs (V), adjectives (A), and prepositions (P) accordingly. Additionally, even the projection of a sentence, or S', can be thought of as the maximum projection of inflection (INFL) [13]. The GB anticipates presentations at two levels: first, the head is projected at a semi-phrasal level, marked by an X-bar, and after the second maximal projection is projected at a morphemes level, represented by an X-double bar.

The first level optimum projection for phrases is represented by S, and the second level maximal projection by S' for the sentences: *saa mahilaa patye bhojanaM*

*pachati* consider the representation of NP, VP, PP and last representation of the sentence.

Noun Prese structure (NP): *saa mahilaa patye bhojanaM*

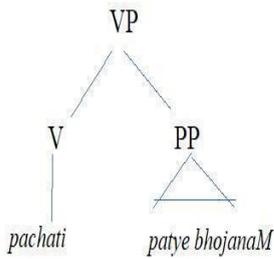
[ NP [DET *saa*] [ N *mahilaa*] [PP *patye bhojanaM*] ] in figure 5.



**Fig. 5.** Noun Prese structure *saa mahilaa patye bhojanaM*

Verb Prese structure (VP): *pachati patye bhojanaM*

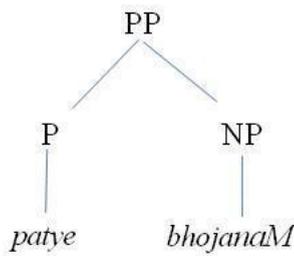
[VP [V *pachati*] [PP *patye bhojanaM*]] in figure 6.



**Fig. 6.** Verb Prese structure *pachati patye bhojanaM*

Prepositional structure (PP): *patye bhojanaM*

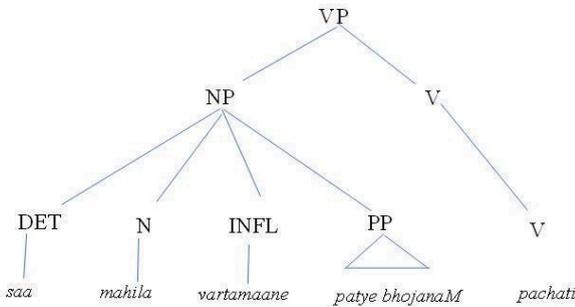
[PP [P *patye*] [N *bhojanaM*]] in figure 7



**Fig. 7.** Prepositional structure *patye bhojanaM*

Sentence structure: *saa mahilaa patye bhojanaM pachati*

[S [ NP [DET *saa*] [ N *mahilaa* ] [INFL *varutamaane*] [PP *patye bhojanaM* ] ] [V *pachati*]] in figure 8.



**Fig.8.** Optimum projection of sentence structure *saa mahilaa patye bhojanaM pachati*

### 3.3 Lexical Functional Grammar (LFG) Model

The LFG characteristics that highlight language modelling are presented in this section. In contrast to GB, LFG [13] depicts sentences at two levels of syntactic organization: constituent structure (c-structure) and functional structure (f-structure). It represented the predicate-argument structure beneath the surface of sentences using phrase structure trees [14]. The use of lexical redundancy rules could be used to resolve these problems, however, if the problem is one of explaining certain linguistic concerns, also including active/passive and dative alternations, in a transformational way. The LFG theory [15] was created due to the fusion of these two disparate ideas.

#### C-structure and f-structure in LFG

LFG provides well-defined objects termed component structure (c-structure) and functional structure to give precise computational algorithms (f-structure) [16]. The

typical phrase and sentence structure syntax is where the c-structure originates. However, because the grammatical-functional role cannot be inferred from phrase and sentence structure alone, functional descriptions nodes of c-structure, although when applied to sentences, result in f-structure. The result is the f-structure, that encodes the data derived from functional specifications and rules governing phrase and sentence construction. For the sentence: *raamaH vedaM paThati*

Design CFG rules of the sentence: *raamaH vedaM paThati*

$S \rightarrow NP VP$

$NP \rightarrow NP NVP \rightarrow V$

When annotated with functional specifications, the rules become:

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

(Rule 2):  $NP \rightarrow NP \quad N$   
 $(\uparrow_{\text{subj}} = \downarrow) \quad (\uparrow_{\text{obj}} = \downarrow)$

(Rule 3):  $VP \rightarrow V$   
 $(\uparrow = \downarrow)$

The f-structure of the parent node, which is on the left side of the rule, is referenced here by the symbol  $\uparrow$  (up arrow). The f-structure of the node in which it is indicated by the  $\downarrow$  (down arrow) symbol.

Therefore, in Rule 1,  $(\uparrow_{\text{subj}} = \downarrow)$  denotes that the first NP's f-structure connects to the sentence's subject f-structure, whereas  $(\uparrow = \downarrow)$  denotes that the VP node's f-structure connects straight to the sentence's VP f-structure. Rule 2 defines the f-structure of the noun phrase, first NP represents a noun whereas the other one represents an object. The lexicon itself can be used to determine the f-structure of V [17]. All terminals in LFG can be regarded as having the annotation  $\uparrow = \downarrow$ . Consider the sentence: *vayam relayaanena gamiShyaamaH*

Again, design rules in LFG for the sentence:

$S \rightarrow NP \quad VP$   
 $(\uparrow_{\text{SUBJ}} = \downarrow) \quad \uparrow = \downarrow$

$NP \rightarrow N$   
 $\uparrow = \downarrow$

$VP \rightarrow NP \quad VP$   
 $(\uparrow_{\text{OBJ}} = \downarrow) \quad \uparrow = \downarrow$

$VP \rightarrow V$

↑ = ↓

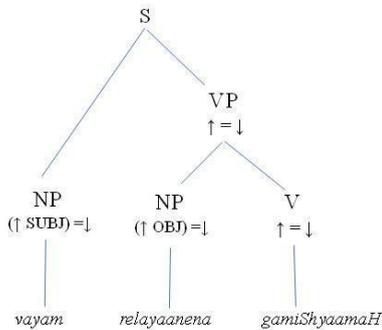
In addition to the lexical entries

N → *vayam*  
 (↑ PRED) = PRO  
 (↑ PERS) = 3  
 (↑ NUM) = PL  
 (↑ CASE) = NULL

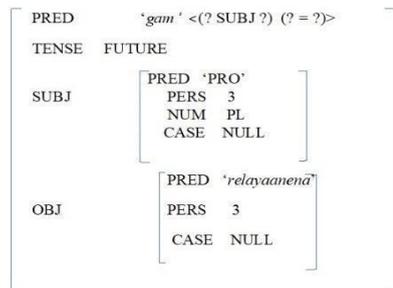
(27) N → *relayaanena*  
 (↑ PRED) = *relayaanena*  
 (↑ PERS) = 3  
 (↑ CASE) = NULL

(28) V → *gamiShyaamaH*  
 (↑ PRED) = '*gam*' <(↑ SUBJ), (↑ OBJ)>  
 (↑ TENSE) = FUTURE  
 (↑ PERS) = 3  
 (↑ CASE) = NULL

Eventually, this will result in the c-structure in figure 9 and f-structure in figure 10:



**Fig. 9.** c-structure of sentence *vayam relayaanena gamiShyaamaH*



**Fig. 10.** f-structure of sentence *vayam relayaanena gamiShyaamaH*

It is noteworthy to observe that several f-structures for the subject, object, verb, complement, etc. are combined to create the final f-structure. The functional characteristics of the verb, that foretell the entire sentence structure, serve as the foundation for this unification. All potential structures for passive constructions, dative constructs, etc., should be stated according to LFG. It is deemed to be ill-formed if the provided sentence does not adhere to the requirements. Three requirements are placed on the f-structure by LFG.

### 4 Statistical Language Model

Statistical language models are distributions over all conceivable word sequences, denoted by the symbol  $P(s)$  [15]. There have been several statistical language models put forth in the literature. The n-gram model is the most popular method for statistical language modelling.

#### 4.1 N-gram Model

Determining the probability of a statement is the aim of a statistical language model [18]. This is accomplished by applying the chain rule to decompose sentence likelihood into a product of conditional probabilities as shown below:

$$\begin{aligned}
 P(s) &= P(w_1, w_2, \dots, w_n) \\
 &= P(w_1) P(w_2/w_1) P(w_3/w_1w_2) \dots P(w_n/w_1 w_2 \dots w_{n-1})
 \end{aligned}
 \tag{1}$$

$$= \prod_{i=1}^n P\left(\frac{w_i}{h_i}\right)$$

where  $h_i$  is the history of the word  $w_i$  defined as

$$w_1 w_2 w_3 \dots w_{i-1}$$

The likelihood of showing the sequence of words before it must therefore be calculated to compute sentence probability. It is not an easy task. By roughly calculating the probability of a word based on all the previous words by the conditional probability given the previous n-1 words only, an n-gram model makes the work easier.

$$P(w_i/h_i) \approx P(w_i/w_{i-n+1} \dots w_{i-1})
 \tag{2}$$

As a result, an n-gram model computes  $P(w_i/h_i)$  by simulating language as a Markov model of rank  $n - 1$ , that is, by focusing solely on the n-1 words that came before it. A bi-gram ( $n = 1$ ) model restricts the history to just the past one word. A tri-gram ( $n = 2$ ) model bases a word's likelihood on the two words that came before it. The

probability of a sentence can be determined using bi-gram and tri-gram estimates as follows:

$$(3) \quad P(s) = \prod_{i=1}^n P\left(\frac{w_i}{w_{i-1}}\right)$$

The bi-gram approximation of  $P(\textit{kaushalyaa} / \textit{dasharathasya bhaaryaa. raamasya evam lakSmaNasya maataa cha asti})$  is  $P(\textit{kaushalyaa} / \textit{dasharathasya})$ , where a tri-gram approximation is  $P(\textit{kaushalyaa} / \textit{maataa cha})$ .

To indicate the start of the sentence in bi-gram estimation, a special word (pseudo word)  $\langle s \rangle$  is added. The likelihood of the first word in a sentence is dependent on the letter  $\langle s \rangle$ . Like this, propose two pseudo-words,  $\langle s1 \rangle$  and  $\langle s2 \rangle$ , for tri-gram estimation.

now go over how to calculate these probabilities. The n-gram model is trained on the training corpus to achieve this. Utilizing the maximum likelihood estimation (MLE) method, or using relative frequencies, estimate n-gram parameters. Take the number of individual n-gram in the training corpus and divide it by the total number of n-grams with that prefix.

$$(4) \quad P(w_i/w_{i-n+1}, \dots, w_{i-1}) = \frac{C(w_{i-n+1}, \dots, w_{i-1}, w_i)}{\sum_w C(w_{i-n+1}, \dots, w_{i-1}, w)}$$

The count of the common prefix  $w_{i-n+1}, \dots, w_{i-1}$  is the total of all n-grams that share the initial n-1 words. Therefore, the prior expression is revised as follows:

$$(5) \quad P(w_i/w_{i-n+1}, \dots, w_{i-1}) = \frac{C(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C(w_{i-n+1}, \dots, w_{i-1})}$$

The model parameter obtained from these estimations,  $P(T/M)$ , maximizes the likelihood that the training set T will match the model M. This model merely offers the most likely answer; the frequency of a term in a text could not match that of the training set. The traditional n-gram model has a lot of room for development. It is more elaborate by the training set:

Bi-gram model:

*siitaa raamasya bhaaryaa asti. siitaa dasharathaH putravadhuH asti. lakShmaNasya bhaaryaa dasharathasya putravadhu cha asti. kaushalyaa dasharathasya bhaaryaa , raamasya evam lakSmaNasya maataa cha asti*

$P(\text{siitaa} / \langle s \rangle) = 0.46$	$P(\text{raamasya} / \text{siitaa}) = 1.0$	$P(\text{bhaaryaa} / \text{raamasya}) = 0.34$
$P(\text{asti} / \text{bhaaryaa}) = 0.22$	$P(\text{dasharathaH} / \text{siitaa}) = 0.58$	$P(\text{putraVadhuH} / \text{dasharathaH}) = 1.0$
$P(\text{asti} / \text{putraVadhuH}) = 0.62$	$P(\text{bhaaryaa} / \text{lakSmaNasya}) = 0.11$	
$P(\text{dasharathasya} / \text{bhaaryaa}) = 0.23$	$P(\text{putraVadhuH} / \text{dasharathasya}) = 0.12$	
$P(\text{cha} / \text{putravadhu}) = 0.7$	$P(\text{asti} / \text{cha}) = 1.0$	$P(\text{dasharathasya} / \text{kaushalyaa}) = 1.0$
$P(\text{bhaaryaa} / \text{dasharathasya}) = 0.42$	$P(\text{raamasya} / \text{bhaaryaa}) = 0.87$	
$P(\text{evam} / \text{raamasya}) = 0.39$	$P(\text{lakSmaNasya} / \text{evam}) = 0.37$	$P(\text{maataa} / \text{lakSmaNasya}) = 0.18$
$P(\text{cha} / \text{maataa}) = 0.72$		

Test sentence: *siitaa raamasya bhaaryaa dasharathasya putravadhu cha asti*

$P(\text{siitaa} / \langle s \rangle) * P(\text{raamasya} / \text{siitaa}) * P(\text{bhaaryaa} / \text{raamasya}) * P(\text{dasharathasya} / \text{bhaaryaa}) *$

$$P(\text{putravadhu} / \text{dasharathasya}) * P(\text{cha} / \text{putravadhu}) * P(\text{asti} / \text{cha}) \\ = 0.46 * 1.0 * 0.34 * 0.23 * 0.12 * 0.78 * 1.0 = 0.00336$$

Since all probabilities must be smaller than 1, multiplying them could result in a numerical underflow, especially in lengthy sentences. Parameters are calculated in log space, which entails adding the log of each probability and subtracting the aggregate to eliminate this.

## 5 Conclusion

Language modeling is essential in natural language processing as it offers systematic approaches to represent and analyze natural languages. The two principal methodologies employed for this aim are grammar-based models and statistical models. Grammar-centric approaches, like Lexical Functional Grammar, Paninian Grammar, and Government and Binding (GB) theory, provide systematic linguistic representations. The Paninian framework is exceptionally proficient for modeling Indian languages owing to its syntactic-semantic basis, particularly the Karaka system. Statistical models, such as n-grams, assist in predicting the probability of word sequences but encounter limits due to inadequate data. Methods such as smoothing and caching have been implemented to address these difficulties.

## 6 Future Scope

Future improvements in language modeling may concentrate on hybrid models that integrate the advantages of both grammar-based and statistical methodologies, facilitating enhanced accuracy and flexibility in natural language comprehension. Refining Paninian grammar for computer applications could improve the representation of Indian languages in digital systems. The amalgamation of machine learning with conventional grammar-based models presents the potential for the creation of more flexible language models. Furthermore, mitigating data sparsity by advanced smoothing techniques or more complex models such as deep learning could enhance the performance of statistical models, particularly in low-resource languages.

## References

1. Kaur, B., & Mishra, S.: A Review on Paninian Grammar Framework in Context of Indian Languages. *International Journal of Advanced Science and Technology* 29(5), 8896–8903 (2021).
2. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I.: Language Models are Unsupervised Multitask Learners. *OpenAI* (2019).
3. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* (2019).
4. Upadhyay, A., & Shukla, A.: Sanskrit Parsing Using Dependency Grammar and Paninian Framework: Recent Progress and Future Directions. *Journal of Language Technology and Computational Linguistics* 47(3), 225–242 (2019).
5. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4171–4186 (2019).
6. Nair, S., & Reddy, B.: Natural Language Processing of Indian Languages: A Paninian Grammar Approach. *Journal of Computer Science and Information Technology* 8(1), 45–54 (2021).
7. Jain, S., & Dey, P. P.: Hybrid Language Models for Indian Languages Using Paninian Grammar and Neural Networks. *Proceedings of the 2021 International Conference on Artificial Intelligence and Machine Learning*, 235–242 (2021).
8. Ji, Y., Cohn, T., Kong, L., Dyer, C., & Eisenstein, J.: Document Context Language Models. *Proceedings of the International Conference on Learning Representations*, 148–154 (2016).
9. Gupta, R., & Jagwani, A.: Computational Sanskrit Grammar: Exploring the Paninian Path to NLP. *Proceedings of the 6th International Sanskrit Computational Linguistics Symposium*, 23–134 (2019).
10. Agarwal, S., & Malviya, R.: Improving N-gram Statistical Models for Low-Resource Languages: A Case Study on Hindi and Sanskrit. *ACM Transactions on Asian and Low-Resource Language Information Processing* 19(5), 1–14 (2020).
11. Singh, A., Mittal, H., & Yadav, S.: Morphological Analysis and Generation for Sanskrit: A Computational Approach. *Journal of Computational Linguistics* 47(2), 357–375 (2021).
12. Jurafsky, D., & Martin, J. H.: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (3rd ed.). *Pearson* (2023).
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I.: Attention is all you need. *Advances in Neural Information Processing Systems*, 5998–6008 (2017).
14. Jha, G. N., Sharma, D., & Goyal, P.: Digital Humanities and the Future of Sanskrit Computational Linguistics. *Journal of Indic Studies* 15(1), 75–92 (2021).
15. Shi, D.: A Study on Neural Network Language Modeling. *arXiv preprint arXiv:1708.07252* (2017).
16. Alam, T., & Wani, M. A.: A Systematic Review of Recent Advances in Natural Language Processing for Indian Languages. *IEEE Access* 8, 135987–136011 (2020).
17. Tapaswi, N., Jain, S., & Chourey, V.: Parsing Sanskrit Sentences Using Lexical Functional Grammar. *Proceedings of the 2012 International Conference on Systems and Informatics (ICSAI2012)* (2012).
18. Tapaswi, N.: Spellchecker for Sanskrit Sentences Based on Morphological Analysis. *UGC Index Journal Kiranavali* 14(3–4), 67–77 (2023)

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

