Assignment - 1: Descriptive Analysis of Hindi Dependency Data - Report

Linguistic Data 3: Data Modeling in ILs

Vivek Hruday Kavuri
2022114012

January 25, 2026

Abstract

This report presents a descriptive linguistic and statistical analysis of the Hindi-Urdu Treebank (HUTB). Using a custom Python implementation, we analyzed 41,399 sentences to determine basic corpus statistics, word order typology, case marking patterns, intervening distance between markers, and Part-Of-Speech (POS) distributions. A notable finding is the statistical dominance of SVO order (57.81%) over the canonical SOV order (38.78%) in this specific dataset, derived using a robust dependency-aware extraction algorithm.

1 1. Basic Corpus Statistics

1.1 Implementation Details

The analysis script iterates through the CoNLL-formatted data line by line.

- **Sentence Count:** Incremented upon encountering a blank line that signifies the end of a sentence block.
- **Token Count:** Calculated by summing the number of non-punctuation tokens. Tokens with the POS tag `SYM` (Symbol/Punctuation) were explicitly excluded to ensure the count reflects only linguistic content.
- **Word Types:** A Python `set` data structure was used to store unique word forms (excluding punctuation), providing the vocabulary size of the corpus.

1.2 Results

Table 1: Basic Corpus Statistics

Metric	**Value**
Total Number of Sentences	41,399
Total Word Tokens (excluding punctuation)	603,578
Total Word Types (Vocabulary Size)	21,366
Average Sentence Length	14.58 tokens
Minimum Sentence Length	1 token
Maximum Sentence Length	106 tokens

2 2. Word Order Patterns

2.1 Algorithm Implementation

Hindi is linguistically classified as an SOV language, but it exhibits relatively free word order and often uses complex predicates (Main Verb + Auxiliaries). To accurately capture the effective word order, we implemented a **Dependency-Aware Surface Order Extraction** algorithm:

1. **Index Mapping:** A hash map is built for each sentence to map every Token ID to its linear index (position) in the sentence string.
2. **Predicate Identification (V):**
   - The algorithm identifies the main verb head using the dependency label `deprel='main'`.
   - **Verb Complex Handling:** Since the "verb" in Hindi is often a phrase (e.g., *khaa liyaa hai*), using the position of the main verb alone can be misleading. The algorithm searches for all children of the main verb that are auxiliaries (POS tag starting with 'V').
   - **Surface Position:** The **rightmost** index (maximum linear position) among the main verb and its auxiliaries is selected as the representative position of the Verb (V_{pos}).
3. **Argument Identification (S O):**
   - **Subject (S):** Identified by the dependency label `k1` (karta).
   - **Object (O):** Identified by the dependency label `k2` (karma).
   - **Constraint:** To handle complex sentences (e.g., "Ram said [that he ate apple]"), the algorithm strictly checks `parent_id`. Only Subjects and Objects that are **direct children** of the identified Main Verb are counted. This prevents cross-clause errors.
4. **Pattern Determination:** The indices of S, O, and V_{pos} are sorted to produce the final string pattern (e.g., if $Index(S) < Index(V) < Index(O)$, the pattern is SVO).

2.2 Results

The analysis was performed on sentences where a distinct Subject, Object, and Main Verb could be identified.

Table 2: Frequency of Word Order Patterns

Pattern	**Count**	**Percentage**
SVO	7,773	57.81%
SOV	5,214	38.78%
OSV	442	3.29%
Other (OVS, VSO, VOS)	17	0.13%

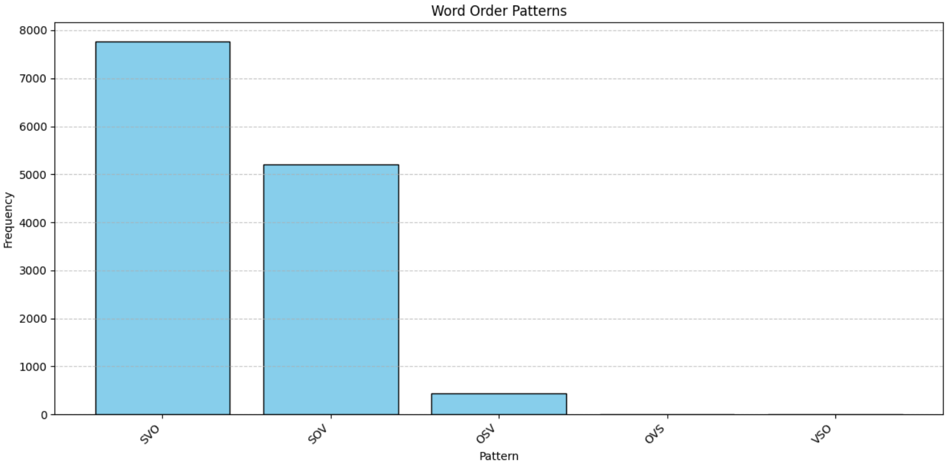

Figure 1: Distribution of Word Order Patterns showing SVO dominance

2.3 Discussion

While Hindi is canonically SOV, the data reveals a dominance of **SVO (57.81%)**. Because our algorithm explicitly handles the verb complex (by taking the rightmost auxiliary), this result is robust against simple errors. This dominance suggests two possibilities:

1. **Extraposition:** Heavy constituent shift often moves objects (especially long or complex ones like clauses labeled as `k2`) to the post-verbal position.
2. **Annotation Specifics:** The label `k2` in this treebank might be used for complements that naturally follow the verb in specific constructions.

3 3. Case Marker and Vibhakti Analysis

3.1 Algorithm Implementation

Case markers in Hindi (Vibhaktis) are often clitics or separate postpositions encoded in the morphological feature column.

- **Extraction:** We utilized Regular Expressions (`re`) to parse the morphological string. The pattern `vib-([^|]+)` was used to extract the content of the 'vib' attribute.
- **Unmarked Noun Detection:** A noun (POS starting with 'NN') was classified as "Unmarked" if:
  1. The extracted vibhakti was empty or explicitly '0'.
  2. OR the morphological case state was explicitly marked as `case-d` (Direct Case), which linguistically implies the absence of an oblique marker.

3.2 Results

The corpus shows a high frequency of Genitive (`kA`) and Locative (`meM`) markers.

Table 3: Top 8 Case Markers (Vibhaktis)

Marker	**Count**	**Percentage**
0_kA (Genitive)	42,947	15.96%
0_meM (Locative)	22,467	8.35%
yA	22,154	8.24%
hE	18,669	6.94%
0_ko (Acc/Dat)	16,730	6.22%
0_ne (Ergative)	13,642	5.07%
0_se (Instr/Abl)	11,437	4.25%
0_para (Locative)	8,504	3.16%

Unmarked Nouns:

- **Count:** 126,311
- **Percentage:** 49.00% of all nouns are unmarked (Direct Case).

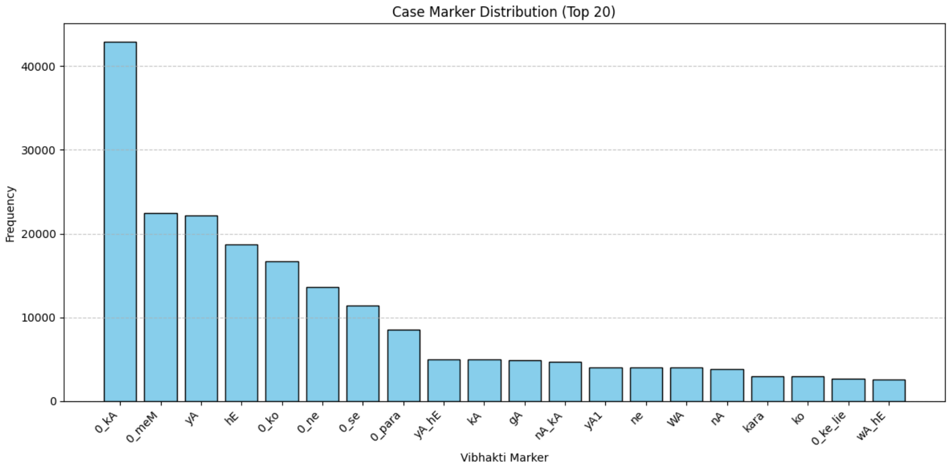

Figure 2: Frequency Distribution of Case Markers

4 4. Intervening Distance Analysis

4.1 Algorithm Implementation

This analysis measures the linear distance between case-marked entities to understand argument clustering.

1. **Collection:** For each sentence, we create a sorted list of tokens that possess a non-null vibhakti marker.
2. **Distance Calculation:** We iterate through this list in pairs (M_i, M_{i+1}). The distance is calculated as:
$$D = Index(M_{i+1}) - Index(M_i) - 1$$
This subtracts 1 to exclude the tokens themselves, counting only the *intervening* words. Punctuation tokens were skipped during indexing.

3. **Categorization:**
    - If $Marker(M_i) == Marker(M_{i+1})$, the distance is added to the **Same Marker** list.
    - Otherwise, it is added to the **Different Marker** list.

4.2 Results

- **Average Intervening Distance:** 1.29 words (Standard Deviation: 1.54)
- **Same Marker Distance:** 1.64 words
- **Different Marker Distance:** 1.27 words

4.3 Discussion

The results show that **Different Markers (1.27)** appear significantly closer to each other than **Same Markers (1.64)**. This supports the "Argument Structure" hypothesis: a sentence typically presents a sequence of distinct arguments (e.g., Agent + Recipient + Theme) in a cluster, whereas repeating the same case role (e.g., two Ergative agents) is rare and syntactically distant (e.g., in coordinate structures).

5 5. POS Tag Distribution

5.1 Algorithm Implementation

To ensure accurate distribution analysis without double-counting (e.g., preventing a Proper Noun `NNP` from being counted in the generic Noun `NN` category), we used strict filtering logic:

- **NN (Common Noun):** Matches exact `NN` or `NN:*` but explicitly excludes `NNP`.
- **NNP (Proper Noun):** Matches tags starting with `NNP`.
- **VM (Main Verb):** Matches tags starting with `VM`.
- **Ratio Calculation:** Computed as $\frac{Total\_VM}{Total\_NN+Total\_NNP}$.

5.2 Results

Table 4: Major POS Category Distribution

Category	Count	Percentage
NN (Common Noun)	170,128	28.19%
VM (Main Verb)	91,916	15.23%
PSP (Postposition)	82,327	13.64%
NNP (Proper Noun)	78,632	13.03%
PRP (Pronoun)	35,599	5.90%
JJ (Adjective)	32,981	5.46%
CC (Conjunction)	32,739	5.42%

Verb-to-Noun Ratio:

- Total Nouns: 257,794
- Total Verbs: 91,916
- **Ratio:** 0.36

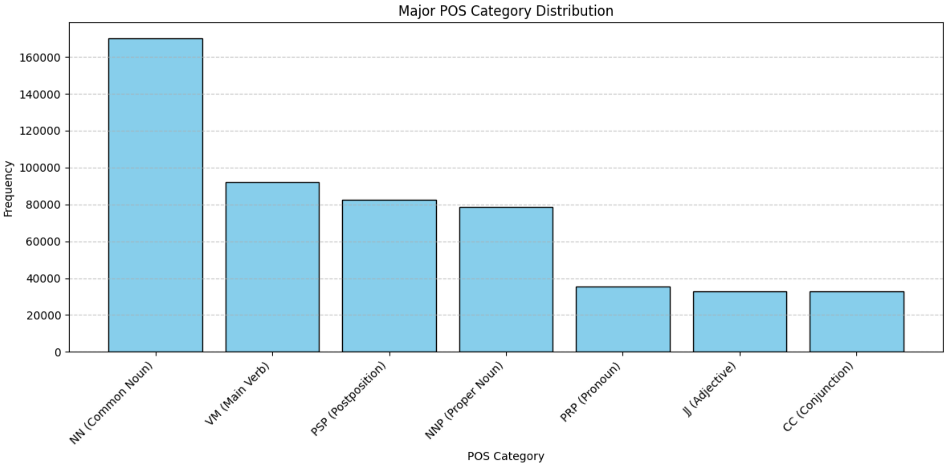

Figure 3: Distribution of Major POS Categories