Automatic Sentence Simplification for Hindi: Methods and Application

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Kshitij Mishra
200902044
kshitij.mishra@research.iiit.ac.in

Advisor: Dr. Dipti Misra Sharma

International Institute of Information Technology Hyderabad
500 032, India

May 2024

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis proposal titled ***Automatic Sentence Simplification for Hindi: Methods and Application*** by *Kshitij Mishra* has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Advisor: Dr. Dipti Misra Sharma

Abstract

Cognitive and psychological studies on human reading indicate that the effort required to read and understand text increases with sentence complexity (Klein and Kurkowski, 1974). Modern natural language processing (NLP) applications face similar challenges. Processing complex sentences with high accuracy remains difficult in computational linguistics, necessitating automatic sentence simplification techniques (Chandrasekar et al., 1996). Sentence complexity can be classified into 'lexical complexity' and 'syntactic complexity'. Lexical complexity can be managed by utilizing resources like lexicons, dictionaries, and thesauruses to replace infrequent words with common ones. To address syntactic complexity, analyzing sentence structure and applying simplification operations is essential.

There are many applications of sentence simplification in NLP. Machine translation systems struggle with long, complex sentences, especially when dealing with divergent language pairs. For parsing, (McDonald and Nivre, 2007) showed that syntactic parsing of long sentences and identifying long-distance dependencies remain challenging. Simplifying sentences can aid parsing and machine translation by breaking down sentences into smaller parts. In automatic summarization, simplification can improve accuracy by extracting smaller units of information.

In this work, we studied and analyzed existing sentence simplification methods for Hindi. We developed new approaches, both rule-based and statistical, to design a better system that overcomes the limitations of existing systems while improving quality and readability. As an application, we examined the effects of sentence simplification on Hindi to English machine translation systems.

Contents

List of Figures

List of Tables

List of Related Publications

[P1] Mishra, Kshitij, et al., **“Exploring the effects of sentence simplification on Hindi to English machine translation system”**, in proceedings of *the workshop on Automatic Text Simplification-Methods and Applications in the Multilingual Society (ATS-MA 2014).*, 2014.

[P2] Mishra, Kshitij., **“Exploring ways to improve Quality of Automatic Sentence Simplification for Hindi.”**, in proceedings of *19th International Conference on Computational Linguistics and Intelligent Text Processing*, 2018.

Chapter 1

Understanding Sentence Simplification: Definitions, Complexity Measures, and Error Analysis

1.1 Introduction

Research in cognitive and psychological fields has demonstrated that the complexity of a sentence directly influences the cognitive effort required for reading and comprehension (Klein and Kurkowski, 1974) . Similar challenges are observed in the realm of natural language processing (NLP), where processing complex sentences accurately remains a formidable task. This necessitates the development of techniques dedicated to the automatic simplification of sentences (Chandrasekar et al., 1996).

Sentence complexity is primarily categorized into two types: lexical and syntactic. Lexical complexity can be effectively managed by utilizing linguistic resources such as lexicons, dictionaries, and thesauruses to replace infrequent words with more common alternatives. Addressing syntactic complexity involves analyzing the structure of sentences and applying targeted transformations to simplify them.

Sentence simplification plays a crucial role in various NLP applications. For instance, machine translation systems often struggle with translating long and complex sentences, particularly between linguistically diverse language pairs. Additionally, syntactic parsing of lengthy sentences and identification of long-distance dependencies continue to pose challenges, as highlighted by McDonald and (McDonald and Nivre, 2007). Simplifying sentences into shorter, more manageable segments can improve the efficiency of parsing and machine translation processes. In the domain of automatic text summarization, reducing sentence complexity has been shown to enhance the accuracy of information extraction, as simpler sentences facilitate clearer and more concise summaries.

This thesis presents a comprehensive study and analysis of existing methodologies for sentence simplification, with a focus on Hindi. We introduce novel approaches that combine rule-based and statistical methods aimed at enhancing the simplification process. These approaches are designed not only to address the limitations of current systems but also to yield outputs with superior quality and readability. Moreover, we explore the implications of these simplification strategies in the context of Hindi to English machine translation systems, assessing their potential to improve translational accuracy and fluency.

1.2 Related Work

(Siddharthan, 2002) presents a three stage pipelined approach for text simplification. He has also looked into the discourse level problems arising from syntactic text simplification and proposed solutions to overcome them. In his later works (Siddharthan, 2006), he discussed syntactic simplification of sentences. He has formulated the interactions between discourse and syntax during the process of sentence simplification. (Chandrasekar et al., 1996) proposed Finite state grammar and Dependency based approach for sentence simplification. They first build a structural representation of the sentence and then apply a sequence of rules for extracting the elements that could be simplified. (Chandrasekar and Srinivas, 1997) have put forward an approach to automatically induce rules for sentence simplification. In their approach all the dependency information of a words is localized to a single structure which provides a local domain of influence to the induced rules. (Sudoh et al., 2010) proposed divide and translate technique to address the issue of long distance reordering for machine translation. They have used clauses as segments for splitting. In their approach, clauses are translated separately with non-terminals using SMT method and then sentences are reconstructed based on the non-terminals. (Doi and Sumita, 2003) used splitting techniques for simplifying sentences and then utilizing the output for machine translation. (Leffa, 1998) has shown that simplifying a sentence into clauses can help machine translation. They have built a rule based clause identifier to enhance the performance of MT system. Though the field of sentence simplification has been explored for enhancing machine translation for English as source language, we don't find significant work for Hindi. (Poornima et al., 2011) has reported a rule based technique to simplify complex sentences based on connectives like subordinating conjunction, relative pronouns etc. The MT system used by them performs better for simplified sentences as compared to original complex sentences.

1.3 Complex Sentences

In this section of the thesis, we examine the multifaceted nature of sentence complexity within the scope of Natural Language Processing (NLP) applications, aiming to propose solutions that could generally enhance NLP system performance. The complexity of a sentence typically manifests through its structural components, which often include multiple clauses. According to (Bhatia, 2015), complex sentences are usually characterized by the presence of multiple clauses linked by connectives.

Furthermore, in their study on dependency parsing, McDonald and (McDonald and Nivre, 2007) have demonstrated that the length of a sentence proportionally increases its complexity, primarily because longer sentences are more challenging to process effectively. During our experiments with the Hindi language, we observed a correlation between sentence length and the increase in the number of verb chunks. This relationship suggests that the number of verb chunks could serve as a quantifiable metric for defining sentence complexity.

Criterion 1	Criterion 2	Criterion 3	Category
No	No	No	Simple
No	No	Yes	Simple
No	Yes	No	Simple
No	Yes	Yes	Simple
Yes	No	No	Simple
Yes	No	Yes	Complex
Yes	Yes	No	Complex
Yes	Yes	Yes	Complex

Table 1.1 Classification of Sentence Complexity

Additionally, the presence of conjuncts in lengthy sentences further complicates sentence structure, leading us to identify it as a second critical factor in defining complex sentences. To systematize our approach to sentence complexity in Hindi, we have established the following criteria:

- **Criterion 1**: The length of the sentence is greater than five words.
- **Criterion 2**: There are more than one verb chunk in the sentence.
- **Criterion 3**: There is at least one conjunct present in the sentence.

  Based on these criteria, we propose a classification schema to categorize sentences into complexity levels, as shown in Table 1.1. This classification helps in identifying and subsequently simplifying complex sentences to improve the accuracy and efficiency of NLP applications.

This table efficiently encapsulates our method for evaluating sentence complexity, facilitating targeted interventions in NLP tasks that involve parsing and translating complex sentences. Our research suggests that by addressing these fundamental elements of sentence complexity, we can significantly enhance the performance of NLP systems in handling the intricacies of human language.

1.4 Analysis of Existing System

In this portion of the thesis, we delve into the analysis of an existing system dedicated to sentence simplification via the utilization of Verb Frames, as detailed by (Soni et al., 2013). Their methodology is based on a rule-driven process that simplifies complex sentences and is executed in two distinct stages. The comprehensive workflow of their approach is depicted in Figure 1.1.

- **Stage 1: Structural Representation**

  Initially, the method entails acquiring a structural representation of the input sentence using a shallow parser. This stage serves as the foundational step, preparing the sentence for subsequent rule-based transformations.

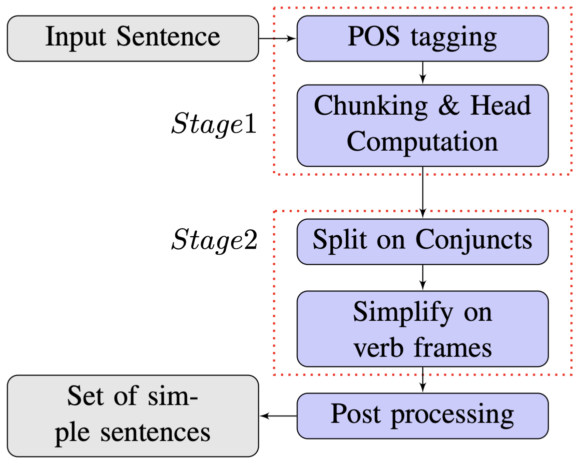


Figure 1.1 Flow-chart showing the work flow of the existing sentence simplification system

- **Stage 2: Rule Application and Simplification**

  In the subsequent stage, the output from the first stage undergoes a process where predefined rules are applied. These rules are designed to detect and address complexities in the sentence, facilitating simplification based on a pre-compiled list of conjuncts and verb frames.

1.4.1 Algorithm Overview

1. **Splitting on Conjuncts**: The initial module of the simplification process involves dissecting the sentence based on conjuncts. Specifically, conjuncts that link two independent clauses are pinpointed. Following this identification, the sentence is segmented, and each segment is then forwarded to the next module for further simplification.

2. **Simplification Using Verb Frames**:

   - Post-splitting, sentences are assessed for complexity. If a sentence is deemed complex, it is then processed to generate multiple simpler sentences. This involves transforming non-finite verbs (VGNF) and gerunds (VGNN) into finite verbs (VGF). Due to the shared arguments between the main verb and the non-finite verbs or gerunds, it poses a challenge for machines to identify implicit arguments. Explicitly breaking the sentence and reassigning these arguments significantly aids in clarification.

   - For the transformation from VGNF to VGF, the procedure commences with the identification of the head of the chunk using the output from the shallow parser. Subsequently, the verb frame corresponding to the root form of the non-finite verb is employed, and transformations are executed following the Tense-Aspect-Modality (TAM) of the main clause's finite verb. The conversion from VGNN to VGF follows a similar methodology, with the additional step of generating pronouns to replace VGNNs.

Example
ram khana khakar mohana ko bulata hai
ram food eat-do mohana call is
After eating food, Ram calls for Mohana'
Output
ram khana khata hai
ram food eat is
'Ram eats food'
ram mohana ko bulata hai
ram mohana calls is
'Ram calls mohana'

1.5 Systemic Limitations and Challenges

In this section of the thesis, we critically examine the operational efficacy and limitations of the current sentence simplification system, which utilizes verb frames as a central component. While the system demonstrates competence in handling standard cases, a deeper analysis reveals several inherent constraints that could potentially undermine its overall effectiveness. We tested the system on a test data of 100 sentences and found the following limitations:

1. **Cascading Errors**:

    - The simplification system is reliant on various auxiliary resources such as shallow parsers and verb frames. This dependency increases the susceptibility to errors, particularly cascading errors, where an issue in one component can adversely affect subsequent processing stages. The risk of cascading failures becomes a significant concern, impacting the reliability of the system.

2. **Limitations of Verb Frames**:

    - Verb frames, by design, are static resources (Bharati et al., 1995). However, the lexicon of verbs is dynamically evolving, with new verbs continually being introduced. The system's inability to recognize verbs not pre-listed in the verb frame resource poses a notable challenge, potentially leading to failures in processing new or uncommon verbs.
    - Additionally, the construction of these frames is typically based on the simple present tense, assuming habitual actions as the default. This is evident in the karaka relations and postpositions used in the frames, which reflect verb behavior in their most common form (e.g., 'khātā hai' for 'eats' in Hindi). This can limit the system's flexibility in handling verbs in other tenses and contexts.

3. **Selection of Arguments for the Verb Frames**:

- The syntactic structure of Hindi, characterized by its free word order, further complicates the selection of arguments for the verb frames. In Hindi, the position of nouns does not strictly define their relationship with the verb. For example, the sentences:
    - *ram ne fal khaya*
    - *fal ram ne khaya*
    - *khaya ram ne fal*
    - *fal khaya ram ne*

      All convey the same meaning despite the varied word order. This variability presents a unique challenge in accurately determining noun-verb relationships within the system.

4. **Verb Senses**:

    - Each verb can manifest multiple senses, and accordingly, there could be numerous applicable frames for each sense. This multiplicity requires the system to discern the correct sense of a verb in context, a task that can be complex and error-prone without sophisticated contextual analysis.

These limitations underscore the need for ongoing refinement of the sentence simplification system. Addressing these issues involves enhancing the robustness of the parsing algorithms, expanding the verb frame resource to accommodate a broader range of verbs and their various contexts, and developing more sophisticated mechanisms for handling the flexibility of Hindi syntax. The thesis aims to propose solutions that mitigate these challenges, thereby improving the system's accuracy and applicability in real-world NLP applications.

Chapter 2

Advancing Sentence Simplification: Approaches and Impact on Machine Translation

We propose a rule based system for sentence simplification, which first identifies the clause boundaries in the input sentence, and then splits the sentence using those clause boundaries. Once different clauses are identified, they are further processed to find shared argument for non-finite verbs. Then, the Tense-Aspect-Modality(TAM) information of the non-finite verbs is changed.

2.1 Algorithm

Our system comprises of a pipeline incorporating five modules. The first module preprocesses the raw input sentence adding lexical information to each word and chunks. The next module determines the boundaries of clauses (clause identification) and splits the sentence on the basis of those boundaries. Then, the clauses are processed by a gerund handler - which finds the arguments of gerunds, shared argument adder which fetches the shared arguments between verbs, TAM(Tense Aspect Modality) generator which changes the TAM of other verbs on the basis of main verb. Figure 2.1 shows the data flow of our system, components of which have been discussed in further detail in this section.

2.1.1 Preprocessing

Each lexical item within the input sentence is meticulously processed to assign part-of-speech (POS) tags, chunk information, and dependency relations. This detailed annotation follows the Shakti Standard Format (SSF) format as defined in the studies by (Bharati et al., 2009). These annotations are critical as they provide a structured representation of the sentence.

To enrich the lexical data with dependency structure, we utilize the dependency parser developed by (Jain et al., 2013). This parser is specifically chosen for its proficiency in identifying complex grammatical relationships within sentences, which is essential for the precise extraction and analysis of linguistic components.

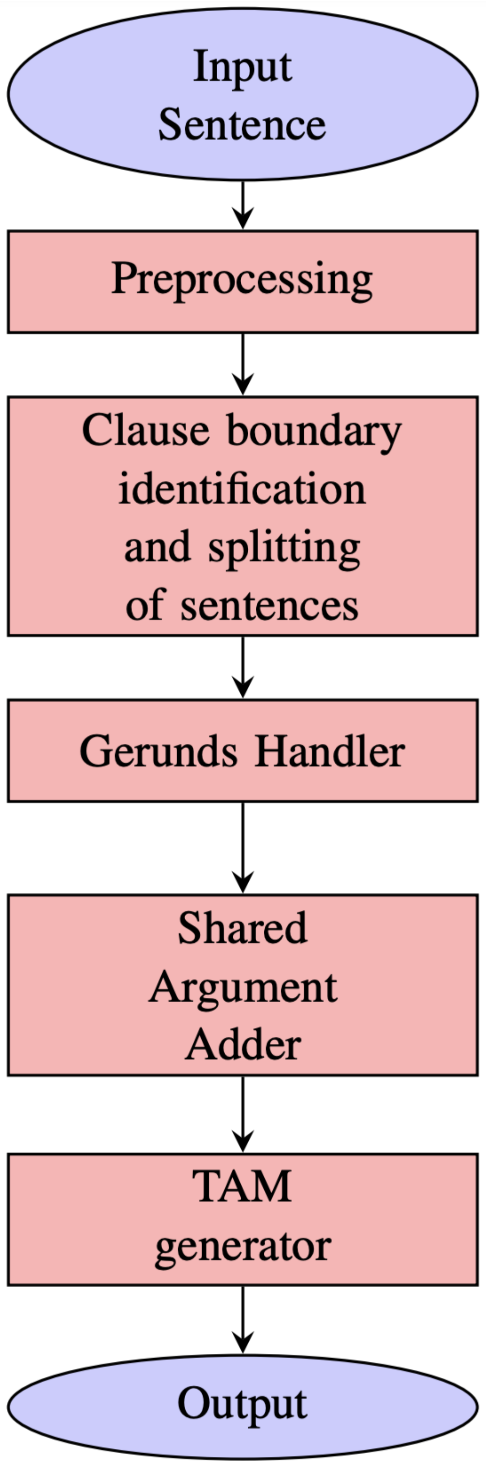


Figure 2.1 Data Flow

```
<Sentence id="1">
1       ((      NP      <fs af='rAma,n,m,sg,3,d,0_ne,0' head="rAma" vpos="vib1_2" name="NP" drel="k1:VGF">
1.1     rAma    NNP     <fs af='rAma,n,m,sg,3,d,0,0' name="rAma">
        ))
2       ((      NP      <fs af='KAnA,n,m,sg,3,d,0,0' head="KAnA" name="NP2" drel="k2:VGF">
2.1     KAnA    NN      <fs af='KAnA,n,m,sg,3,d,0,0' name="KAnA">
        ))
3       ((      VGF     <fs af='KA,v,m,sg,any,,yA,yA' head="KAyA" name="VGF" drel="ccof:CCP">
3.1     KAyA    VM      <fs af='KA,v,m,sg,any,,yA,yA' name="KAyA">
        ))
4       ((      CCP     <fs af='Ora,avy,,,,,,' head="Ora" name="CCP">
4.1     Ora     CC      <fs af='Ora,avy,,,,,,' name="Ora">
        ))
5       ((      NP      <fs af='pAnI,n,m,sg,3,d,0,0' head="pAnI" name="NP3" drel="k2:VGF2">
5.1     pAnI    NN      <fs af='pAnI,n,m,sg,3,d,0,0' name="pAnI">
        ))
6       ((      VGF     <fs af='pI,v,m,sg,any,,yA,yA' head="piyA" name="VGF2" drel="ccof:CCP">
6.1     piyA    VM      <fs af='pI,v,m,sg,any,,yA,yA' name="piyA">
        ))
</Sentence>
```

Figure 2.2 Dependency Parser Output

To illustrate the output generated by this preprocessing step, consider the following input sentence: *Raam ne khaana khaaya aur paani piya*

The output of this preprocessing phase is demonstrated in Figure 2.2 which displays the lexical items annotated in SSF format. This format includes tags that encapsulate both chunk and POS information for each segment of the sentence. Additionally, the 'drel' (dependency relation) feature within the structure meticulously records the various dependency relations present in the sentence, providing a comprehensive overview of its grammatical structure.

2.1.2 Clause Boundary Identification and Splitting of Sentences

The parser provides us verb groups and marks their dependencies. Our next job is to construct small sentences using these verbs and their arguments. We are now looking for sentence-like constructions contained within a sentence, grammatically called, "*clauses*". A clause is a group of words consisting of a verb (or a verb group) and its arguments (explicit and implicit). Depending on the type of the verb, a clause is classified either as finite or non-finite based on the finiteness of the head verb.

For example:

raam khana khaakar ghar gayaa.

Ram food eat+do   home go+past.

'Ram went home after eating.'

In this example, *khana khaakar* is a non- finite clause since *'khaakar'* is a non-finite verb. Similarly, *raam ghar gayaa* is a finite clause as *'gayaa'* is a finite verb. A sentence can have more than one clauses in it. These clauses are classified into two types as:

1. Main clause, which is an independent clause, is also called Superordinate clause

2. Subordinate clause, which is dependent on the main clause.

Below, we discuss some of the clause types mentioned earlier.

1. **Complement Clause** These clauses are introduced by complementizer 'ki' (that) and generally follow the verb of main clause (Koul, 2008). For Example:

    - *Yaha sach hai ki Mohan beemar hai*

    In this example, '*ki mohan bimaar hai'* is a Complement clause and *'ki'* is a complementizer. It must be noted that 'complement clause' may also act an argument of the main clause verb. So, in this example, the main clause is '*yaha sach hai ki mohan bimaara hai'*, which contains the complement clause *'ki mohan bimaara hai'*, in it. This is considered to be a special case where a clause comes as an argument of a verb and becomes a part of the main clause.

2. **Relative Clause** Finite relative clauses occur as a modifier of verb's argument and contain a relative pronoun (Koul, 2009). Such clauses can be either nested or non-nested.

    For example:

    - *Vaha ladaka jo khel raha tha ghar gya*

    In the above example, the nested relative clause is *jo 'khel rahaa thaa'* (who was playing) with *'jo'* as a relative marker. *'jo'* modifies '*vaha'*, the argument of the verb *'gayaa'*.

3. **Coordinate Clause** It is one of the independent clauses in a sentence belonging to a series of two or more independent clauses coordinated by a coordinating conjunction (Koul, 2008). For example:

    - *Main ghar jaunga aur ram dilli jayega*

    'mai ghar jaaungaa' and '*raam dillii jaayegaa'* are two independent clauses with the same status in this sentence. In our work, we consider both clause as coordinate clauses, and the coordinating conjunct is not taken to be part of any of the two clauses. There is thus no hierarchy in these clauses.

For extracting the clauses, we have used method mentioned in (Sharma et al., 2013). They have proposed a two step process in which Step 1 identifies the clause boundary in general, while Step 2 is a post-processing step which do adjustments specifically to handle '*ki*' (that) complement clauses.

2.1.2.1 Step 1

In the initial phase of our analysis, we focus on the extraction and organization of verbs and their dependent elements from the input sentences, a crucial step for constructing a syntactic framework.

- **Verb and Dependent Extraction**: To commence, we systematically identify all verbs present in a sentence using their part-of-speech (POS) tags and chunk information. Following this, we navigate through the dependency tree associated with each verb. This traversal is performed recursively, capturing each dependent of the verb sequentially. The process continues until we either traverse all nodes that are dominated by the verb or encounter another verb within its domain.

- **Handling of Complement Clauses**: A special case arises when a complement clause, introduced by the complementizer *ki*, is annotated as an argument of a verb. In such instances, traversal extends beyond the typical boundaries, continuing until all nodes under the dominion of the complementizer *ki* are thoroughly explored. This approach allows the complement clause to be integrated into the main clause as an embedded structure, ensuring a comprehensive parsing and interpretation of the sentence.
- **Sorting and Clause Boundary Determination**: After extracting the verbs and their dependents, the next step involves sorting these elements by their textual offsets within the sentence. The element with the lowest offset marks the beginning of a clause, while the highest offset determines its endpoint. This methodical sorting helps in accurately delineating the boundaries of each clause, thereby facilitating a structured analysis of the sentence's composition.
- **Example**: To illustrate this process, consider the following input and its analysis
  - *Ram ghar gaya aur khana khaya*
  - Figure 2.3 provides a visual representation of the dependency tree for the given example sentence. This tree is annotated with various relational tags (such as k1, k2, etc.) on the edges connecting the nodes, which represent the grammatical relationships between the verbs and their arguments. Additionally, the offsets of different chunks are indicated in brackets alongside the corresponding words, offering a clear overview of their positions within the sentence.
  - Following the procedures outlined in Step 1, we compile a list of verbs identified in the sentence, which includes *gayaa* and *khaayaa*. We then proceed to traverse the dependency tree for the verb *gayaa*, during which we construct a list comprising the verb *gayaa* and its arguments— *raam* and *ghar*.
  - The elements in the list are sorted according to the offsets of the words they contain. Sorting facilitates the identification of clause boundaries: the word with the lowest offset marks the start of the clause, while the word with the highest offset signifies the end. For the verb *gayaa*, the clause boundaries are established between the words corresponding to the lowest and highest offsets.
  - Similarly, for the verb *khaayaa*, the boundary determination process is conducted with the offsets at positions 5 and 6 marking the limits of the clause.
  - Consequently, the clause boundaries for the example sentence are distinctly marked as follows:
    - ∗ (*raam ghar gayaa*) *aur* (*khaanaa khaayaa*)

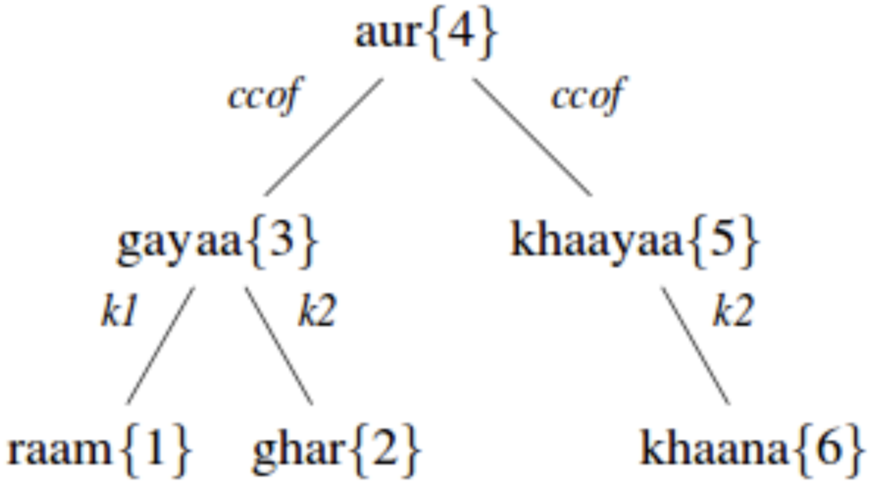


Figure 2.3 Dependency Tree

2.1.2.2 Step 2

- **Postprocessing**: This section of the methodology addresses the postprocessing required for managing complex complement clauses, specifically those introduced by the complementizer *ki* (translated as ”that”). These clauses often present unique challenges due to their potential to be nested and multifaceted within the structure of the main clause.

- **Integration of *ki* Complement Clauses**:

  As established in the initial processing steps, a *ki* complement clause can function as an argument of a verb and, consequently, is considered an integral part of the verb's clause. While Step 1 ensures that these complement clauses are accurately incorporated into the main clause, it does not explicitly delineate the scope of the complement clause if it is inherently complex, comprising more than one subclause.

- **Scope Marking of Complex Complement Clauses**:

  This postprocessing step is specifically designed to address and mark the boundaries of complex complement clauses. These clauses, due to their structural complexity, require detailed analysis to determine their extent and interaction within the overall sentence structure. The procedure utilizes the outputs from Step 1, which include preliminary identifications of clause boundaries and dependencies.

- **Detailed Analysis and Marking Process**:

  The process involves revisiting the *ki* complement clauses identified in Step 1 and examining their internal composition. If these clauses are found to encompass multiple embedded clauses or intricate syntactic constructions, this step will explicitly mark their scope. This is crucial for understanding the complete grammatical and semantic relationships that these complex constructs contribute to the main sentence.

- **Example**:
  - *Raam ne kaha ki tum ghar jaao aur aaram kar lo*
  - After STEP 1, the clause boundaries for the above sentence would be like
    (raam ne kahaa ki (tum ghar jaao) or (aaraam karloo))
  - In STEP 2, we iterate over the output of STEP 1 and mark the boundaries of the complement clause starting from the word immediately following the *'ki'* complementizer and the ending with the end of main clause of which complement clause is a part. The modified boundaries will be:
    - *(raam ne kahaa ki ((tum ghar jaao) or (aaraam karloo)))*

In conclusion, this module takes the input from preprocessing module and identifies the clause boundaries in the sentence. Once clause boundaries are identified, the sentence is divided into different clauses. Once we mark the clause boundaries using this approach, we break the sentence into different simple clauses along those clause boundaries. The example given below illustrates the same.

- *raam jisne khaanaa khaayaa ghar gayaa*

This sentence with clause boundaries marked is,

- *(raam (jisne khaanaa khaayaa) ghar gayaa).*

Once the clause boundaries are marked, we break the sentence using those boundaries. So for this example, split clauses are,

1. *raam ghar gayaa.*
2. *jisne khaanaa khaayaa.*

2.1.3 Gerunds Handler

Since, (Sharma et al., 2013) identifies clause boundary for non-finite and finite verb only, gerunds are not handled in the previous module. This module is used to handle gerunds in the given sentence. In this module, the gerund chunks are first identified and then further processed after getting the arguments.

Consider an example:

logon ko sambodhit karne ke baad dono netaon ne pradhanmantri ko istifa saunpa

In the above example, the clause boundary identifier module marks the entire sentence as a clause but *karne ke baad* is a gerund chunk (verb chunk) here, which is marked as VGNN according to the tagset of the POS tagger used. According to definition of complex sentence given earlier in this report, gerunds also introduce complexity in a sentence. Therefore, in order to simplify such sentences, we use dependency parsing information for extracting the arguments of gerund and splitting the sentence. Here *logon ko* and *sambodhit* are the arguments of verb chunk *karne ke baad*. Here *ke baad* is postposition of verb *karne* so, *ke baad* is splitted from karne and it has been used with the pronoun is to make the sentence more readable.

1. *logon ko sambodhit karne*
2. *iske baad dono netaon ne pradhanmantri ko istifa saunpa*

2.1.4 Shared Argument Adder

After identifying clauses and handling gerunds, the shared arguments are identified between the verbs and sentences are formed accordingly. For example:

- *(ram (chai aur paani peekar) soyaa)*

Here *ram* is the shared argument(k1-karta) of both the verbs *peekar* and *soyaa* . The dependency parser used, marks the inverse dependencies for shared arguments which helps in . So the output of this module is:

- *ram chai aur paani peekar.*
- *ram soyaa.*

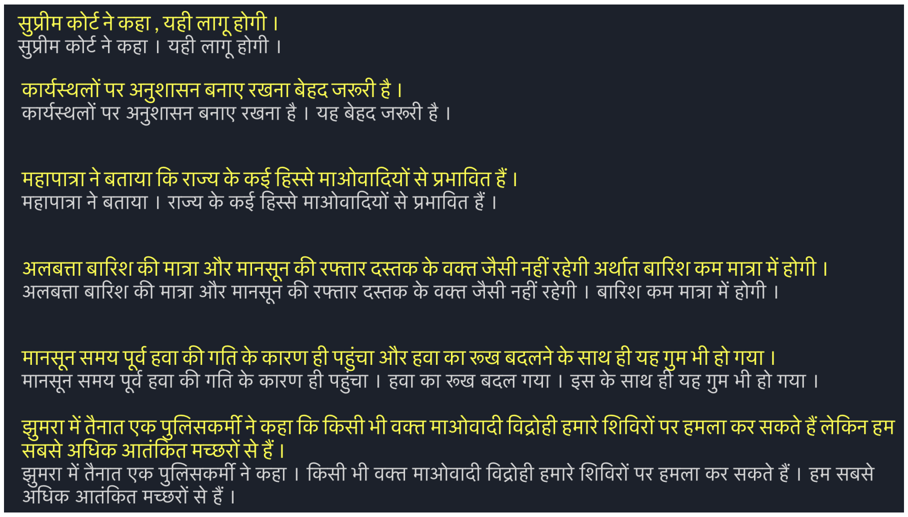
2.1.5 TAM generator

In this stage, the TAM (Tense-Aspect-Modality) generator module plays a critical role in transforming the segmented sentences produced by the previous module into more coherent and readable structures. This conversion ensures that the simplified sentences retain their intended meaning and grammatical correctness.

- **Updating Verb Forms**:

  The primary function of this module is to modify the forms of other verbs in the sentence based on the TAM information of the main verb. This information is provided by the morphological analyzer (morph), which is illustrated in the accompanying figure depicting dependency relations. By aligning the TAM characteristics of the subordinate verbs with those of the main verb, the module ensures temporal and modal consistency across the sentence.

- To exemplify, consider a sentence where the main verb indicates a past tense. The TAM generator will adjust the forms of all relevant verbs within the clause to reflect this past tense, thereby maintaining uniformity and readability. This adjustment not only improves the grammatical flow of the sentence but also enhances the clarity and comprehension for the reader.



सुप्रीम कोर्ट ने कहा , यही लागू होगी ।
सुप्रीम कोर्ट ने कहा । यही लागू होगी ।

कार्यस्थलों पर अनुशासन बनाए रखना बेहद जरूरी है ।
कार्यस्थलों पर अनुशासन बनाए रखना है । यह बेहद जरूरी है ।

महापात्रा ने बताया कि राज्य के कई हिस्से माओवादियों से प्रभावित हैं ।
महापात्रा ने बताया । राज्य के कई हिस्से माओवादियों से प्रभावित हैं ।

अलबत्ता बारिश की मात्रा और मानसून की रफ्तार दस्तक के वक्त जैसी नहीं रहेगी अर्थात बारिश कम मात्रा में होगी ।
अलबत्ता बारिश की मात्रा और मानसून की रफ्तार दस्तक के वक्त जैसी नहीं रहेगी । बारिश कम मात्रा में होगी ।

मानसून समय पूर्व हवा की गति के कारण ही पहुंचा और हवा का रूख बदलने के साथ ही यह गुम भी हो गया ।
मानसून समय पूर्व हवा की गति के कारण ही पहुंचा । हवा का रूख बदल गया । इस के साथ ही यह गुम भी हो गया ।

झुमरा में तैनात एक पुलिसकर्मी ने कहा कि किसी भी वक्त माओवादी विद्रोही हमारे शिविरों पर हमला कर सकते हैं लेकिन हम सबसे अधिक आतंकित मच्छरों से हैं ।
झुमरा में तैनात एक पुलिसकर्मी ने कहा । किसी भी वक्त माओवादी विद्रोही हमारे शिविरों पर हमला कर सकते हैं । हम सबसे अधिक आतंकित मच्छरों से हैं ।

Figure 2.4 Sample output of the system

- Example:
  - INPUT:
    - *ram chai aur paani peekar.*
    - *ram soyaa.*
  - OUTPUT:
    - *ram chai aur paani peeyaa.*
    - *ram soyaa.*

2.2 Evaluation Methodology

2.2.1 Corpus Selection for Evaluation

For the evaluation of our sentence simplification tool, we selected a corpus comprising 100 complex sentences. These sentences were sourced from the Hindi Treebank, as documented by (Bhatt et al., 2009) and (Palmer et al., 2009). The complexity of these sentences makes them ideal candidates for assessing the effectiveness of our simplification tool.

To visually demonstrate the capabilities of our tool, Figure 2.4 presents selected examples from this evaluation. In the figure, each input sentence is highlighted in yellow, and the corresponding simplified outputs are displayed in white.

2.2.2 Nature of Evaluation

The evaluation process for sentence simplification inherently involves both subjective and objective measures, focusing on the readability of the output and the preservation of semantic information. To address these aspects comprehensively, we conducted both manual and automatic evaluations.

2.2.3 Automatic Evaluation

2.2.3.1 Methodology

The automatic evaluation was conducted using the BLEU score metric, established by (Papineni et al., 2002). This metric measures the similarity between the machine-generated text and a set of reference translations, with a score closer to 1 indicating higher resemblance to the reference text, and a score closer to 0 indicating less resemblance.

2.2.3.2 Results

Following the evaluation technique used by (Specia, 2010), our tool achieved a BLEU score of **0.6949** in the sentence simplification task. This score suggests a substantial level of accuracy in maintaining the content and meaning of the original sentences while simplifying them.

2.2.4 Human Evaluation

2.2.4.1 Procedure

To validate the quality of simplification further, a manual evaluation was also performed. Twenty sentences were randomly selected from our test dataset of 100 sentences. The outputs of these sentences were then reviewed by two individuals with foundational knowledge in linguistics.

2.2.4.2 Evaluation Criteria

The reviewers assessed each sentence based on 'Readability' and 'Simplification Quality', using a 0 to 3 scale:

- **0**: No simplification achieved.
- **1**: Partial simplification, with some elements simplified.
- **2**: Most elements within the sentence were simplified.
- **3**: Complete and comprehensive simplification.

2.2.4.3 Outcome

The average score from the evaluations was **2.5**, indicating a high level of effectiveness in simplifying the sentences while retaining their core meaning. This score reflects the tool's capability to produce simplified texts that are easier to read and understand without significantly altering the original information.

The combined insights from both automatic and manual evaluations underscore the robustness and efficiency of our sentence simplification tool, demonstrating its effectiveness in enhancing readability while preserving essential semantic content.

2.3 Effects on Machine Translation

Machine Translation (MT) systems frequently encounter difficulties when translating between highly divergent language pairs due to significant differences in grammar, syntax, and semantic structures. To mitigate these challenges, it is beneficial to simplify complex sentences into more manageable units. This approach aligns with the methodologies employed in phrase-based translation systems, which segment sentences into phrases, translate each segment independently, and then reassemble them. The primary goal in this process is not merely to produce a coherent sentence but to faithfully preserve the semantics of the original text while ensuring readability in the target language.

2.3.1 Evaluating impact of Sentence Simplification in MT Systems

To assess the effectiveness of sentence simplification in enhancing machine translation output, we conducted evaluations both with and without the use of a sentence simplification tool. The results were quantitatively measured using the BLEU score metric, which is widely recognized for evaluating the quality of text translated by machine translation systems.

2.3.2 Quantitative Outcomes

The MT system incorporating the sentence simplification tool achieved a BLEU score of **0.4986**, demonstrating a notable improvement over the system without sentence simplification, which scored 0.4541. This indicates that the simplification tool contributes to a more accurate translation by reducing sentence complexity, which aligns more effectively with the capabilities of the translation system.

2.3.3 Human Evaluation of Machine Translation Quality

In addition to automated metrics, human evaluations were conducted to compare the quality of translations produced by the system with and without the sentence simplification tool. Participants were asked to choose the better translation between the two systems for each sentence. The feedback indicated a preference for the translations produced by the system with the sentence simplification tool, with **12** out of 20 sentences showing improved translation quality post-simplification. In 3 cases, the quality remained unchanged, while in 5 instances, the original non-simplified sentences were judged to have been translated better.

2.3.4 Conclusion

The findings from both quantitative and qualitative evaluations suggest that integrating sentence simplification into machine translation workflows can significantly enhance translation quality, particularly for complex sentences and divergent language pairs. This improvement is primarily attributed to the reduced complexity of the input, which allows the translation algorithms to perform more effectively. However, the results also highlight that the benefits of simplification may vary depending on the specific linguistic features and content of the source text, underscoring the importance of context-aware translation strategies.

Chapter 3

Enhancing the Effectiveness of Sentence Simplification: Development and Evaluation of New Methodologies

Previous chapter, has demonstrated that sentence simplification can significantly improve the efficacy of Hindi to English machine translation systems. We have predominantly utilized a rule-based approach to segment sentences along clause boundaries.

While this method is generally effective for sentences containing two to three verb chunks, it tends to falter with more complex sentence structures, often resulting in a loss of semantic integrity. Additionally, the current system does not modify the grammatical cases (*vibhakti*) of words in the simplified sentences, which can lead to misinterpretations or a loss of the original meaning.

In response to these challenges, this study introduces novel approaches aimed at mitigating these limitations and enhancing the overall quality of sentence simplification. We have developed and utilized a parallel corpus of complex and simplified sentences to facilitate sentence simplification through monolingual machine translation, assessing the effectiveness of these new methodologies in comparison to the original approach.

3.1 Quality Analysis

This section of the thesis delicates to the evaluation of the sentence simplification system developed in our research. To comprehensively assess the quality and effectiveness of the simplification approach, we conducted a structured experiment involving participants with relevant expertise in Natural Language Processing and Computational Linguistics.

3.1.1 Participants

The evaluation involved five participants who are currently pursuing research in the domain of Natural Language Processing or Computational Linguistics. All participants are native Hindi speakers and possess proficient language skills in Hindi, essential for understanding the nuances in the test sentences. Furthermore, each participant has completed basic and some advanced courses in linguistics, providing them with the necessary analytical skills to assess and critique the simplification outcomes effectively.

Data Set	No of Verb Chunk per Sentence
Data Set 1	2-3
Data Set 2	4-5
Data Set 3	More than 5

Table 3.1 Testing Datasets

3.1.2 Materials

To ensure a comprehensive evaluation, we prepared three distinct sets of test data, each comprising 100 sentences extracted from the Hindi Treebank (Bhatt et al., 2009). These sets were categorized based on the complexity of the sentences, measured by the number of verb chunks:

- Set 1: Sentences containing 2 to 3 verb chunks.
- Set 2: Sentences containing 4 to 5 verb chunks.
- Set 3: Sentences containing more than 5 verb chunks.

This classification aimed to test the robustness of the sentence simplification system across varying levels of linguistic complexity. Table 3.1 provides a detailed breakdown of the testing dataset division, illustrating the distribution of sentences across the three categories.

3.1.3 Procedure

Participants were instructed to evaluate the output from each dataset using a predefined scale ranging from 0 to 3, where 0 indicates the worst outcome and 3 indicates the best, reflecting the degree of simplification achieved:

- **0 - No Simplification**: None of the expected simplifications were performed.
- **1 - Partial Simplification**: Some of the expected simplifications were performed.
- **2 - Substantial Simplification**: Most of the expected simplifications were performed.
- **3 - Complete Simplification**: Complete and thorough simplification was achieved.

After rating each sentence, participants were also encouraged to provide qualitative feedback, particularly for outputs that received low ratings. This feedback is crucial for identifying specific issues and challenges faced by the sentence simplification tool, allowing further refinement and adjustments to enhance its performance.

This structured evaluation approach not only quantifies the effectiveness of the sentence simplification process but also provides valuable insights into the practical challenges and limitations encountered, facilitating targeted improvements in future iterations of the system.

Data Set	Average Rating
Data Set 1	2.5
Data Set 2	2.0
Data Set 3	1.2

Table 3.2 Average ratings

3.1.4 Results

The average ratings for data set 1 was **2.5**, dataset 2 was **2.0** and dataset 3 was **1.2**. (Table 3.2)

3.2 Major Issues with the Current System

As per the feedback provided, the three main reasons of low ratings came out to be ”loss of naturality/readability”, ”loss of semantic information” and ”grammatical errors”.

3.2.1 Loss of Naturality

”Loss of naturality” was tagged with more than 60% of the low rated outputs. Naturality is the natural flow of a sentence. It is the result of all the experience we have had of that language (including reading, writing, speaking and listening). Any sentence that do not fit in that frame seems anomalous to the brain and is termed as not-natural. The simplest example for explaining the naturality is gender in Hindi. Based on our training of the language, we naturally assign genders of non-living things and any deviation from our assigned gender leads to the feeling that something is wrong. In our case splitting the sentence at verb chunks containing verbs like ”*kaha, mana, bola, aadesh diya* etc.” takes away the naturality. For Example:

- *Ram ne mana ki wo bahut mehanati hai*

simplified to :

- *Ram ne mana.*
- *Wo bahut mehanati hai.*

Another example,

- *Sita ne pucha ki sooryodaya kab hoga.*

simplified to:

- *Sita ne pucha.*
- *Sooryodaya kab hoga.*

In all such cases, though the output is grammatically correct, the first sentence sounds incomplete or *not-natural*.

Devoid of naturality also arises when the sentence is broken at each and every verb chunk. Example:

- *yah poochne par ki kya we dobaara congress mein lautenge sangama ne kaha ki na to iski zarurat hai aur na hi peeche lautane ka sawal hi uthta hai*

simplified to:

- *Kya we dobaara congress mein lautenge.*
- *yah poochane par sangama ne kaha.*
- *Na to iski zarurat hai.*
- *Na hee peeche lautana hai.*
- *Iska sawal uthta hai.*

It is clearly observable that the simplified sentences neither sound natural nor succeed to preserve the meaning.

3.2.2 Grammatically Incorrect Output

In certain instances, the output generated by our sentence simplification system exhibited grammatical inaccuracies. These errors predominantly originated from issues related to the inflexibility of case markers, also known as '*vibhakti*' in Hindi. This phenomenon typically arises when the static nature of case markers in the input sentences is not adequately adjusted during the simplification process.

3.2.2.1 Major Source of Grammatical Errors

We have developed specific rules to modify the verb's Tense-Aspect-Modality (TAM) in the course of simplifying sentences. However, their system does not currently extend these modifications to the case markers. As a result, the system often produces sentences where there is a misalignment between the verb forms and their corresponding case markers.

Example:

- *machharon ke katne ke baad wo beemar hue*

is simplified to

- *machharon ke kata.*
- *is ke baad wo beemar hue.*

In the first simplified sentence the vibhakti '*ke*' should have been changed to '*ne*' for the formation of a valid sentence.

3.2.3 Loss of Semantic Information

The process of simplifying sentences inherently involves certain trade-offs, particularly concerning the preservation of contextual information. In our study, we observed that segmenting sentences strictly at each verb chunk can sometimes lead to a loss of contextual coherence, which may confuse the reader or convey inaccurate information.

3.2.3.1 Challenges with Context Preservation

Our approach involves breaking down complex sentences into simpler units by splitting them at each verb chunk. While this method effectively reduces sentence complexity, it can inadvertently strip away the contextual framework that links various parts of the sentence. This issue is particularly pronounced in sentences where the logical or causal relationships between clauses are crucial for understanding the overall meaning.

3.2.3.2 Example

- *Ram ne jhuth bola ki Mohan mar gya hai aur Rakesh Mumbai chala gya.*

is simplified to:

- *Ram ne jhuth bola.*
- *Mohan mar gya hai.*
- *Rakesh Mumbai chala gya.*

In this example, while each of the resulting sentences is grammatically correct, the crucial contextual link between them is lost. Specifically, the implication that the statements about Mohan and Rakesh might not be true—originally introduced by "*Ram ne jhuth bola*" (Ram lied)—is obscured in the simplification. Therefore, a reader of the simplified sentences may not realize that the assertions regarding Mohan and Rakesh could be false, leading to potential misunderstandings.

3.3 Enhancements

3.3.1 Maintaining Naturality & Preserving the Meaning

The concept of sentence naturality, while subjective, plays a crucial role in the quality of sentence simplification. Our analysis indicates that the principal factor contributing to the loss of naturality is the methodical segmentation of sentences at each verb chunk. Research in cognitive and psychological domains regarding human reading suggests that the criteria for sentence complexity should differ between machines and the human mind. Notably, even advanced translation systems like Google, Bing, and ILMT struggle with sentences containing more than two verb chunks, whereas humans typically manage sentences with four to five verb ridges without significant difficulty.

To address these findings and enhance the naturality of simplified sentences, we have implemented specific improvements in our sentence simplification approach:

3.3.1.1 Selective Non-Breaking at Certain Verbs

Feedback from evaluations has highlighted that breaking sentences at certain verbs can disrupt the natural flow, readability, and contextual integrity of the text. Particularly, this issue arises with "factive" verbs, which inherently imply the truth of the subordinate clause they introduce. Such verbs include *kaha* (said), *mana* (accepted), *bola* (spoke), *bataya* (told), *samjhaya* (explained), *pucha* (asked), among others. Our enhanced approach avoids segmenting sentences at these critical junctures to maintain a coherent narrative flow.

3.3.1.2 Maintaining Integrity at Comma-Separated Verbs

Another improvement involves not segmenting sentences at commas that separate verb phrases. This tactic helps to preserve the linguistic and structural unity of sentences, particularly in complex constructions where commas play a pivotal role in demarcating subtle shifts in narrative or explanatory sequences.

These targeted enhancements are designed to respect the inherent properties of human language processing, thereby making simplified sentences more natural and comprehensible. By adapting our simplification strategy to more closely mimic the way humans parse and understand complex sentences, we aim to reduce cognitive strain and improve the accessibility of the text.

3.3.2 Dealing with the *vibhakti* Change

Whenever the breaking point verb is a transitive verb, the *vibhakti* will change from "*ke*" to "*ne*"
Example:

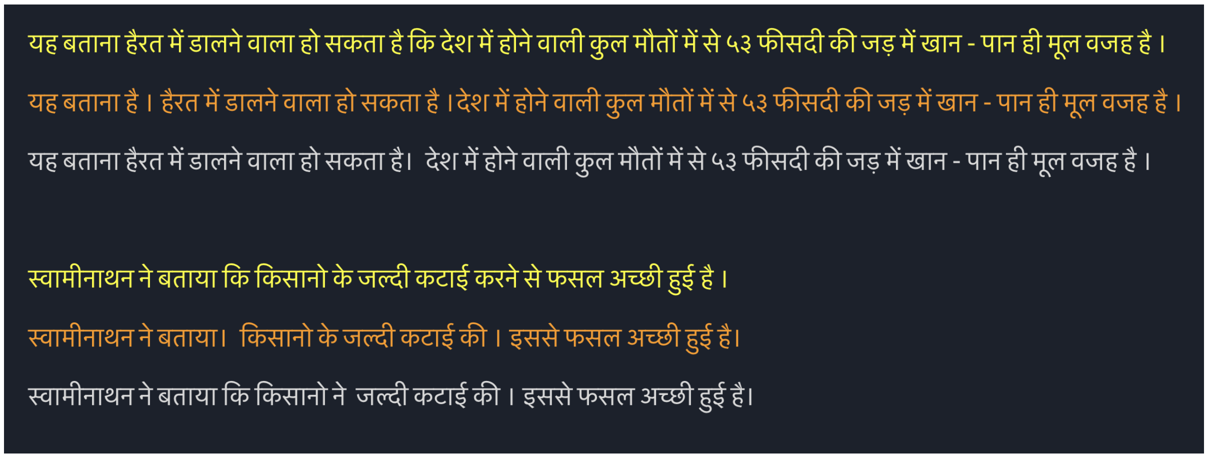
- *machharon ke katne ke baad wo beemar hue*

Output:

- *machharon ke kata.* $\rightarrow$ *machharon ne kata*
- *is ke baad wo beemar hue.*

3.4 Illustration of Enhanced Sentence Simplification

Figure 3.1 presents a comparative visualization of the improvements made in sentence simplification, showcasing two specific examples that highlight the effectiveness of the recent enhancements. This



यह बताना हैरत में डालने वाला हो सकता है कि देश में होने वाली कुल मौतों में से ५३ फीसदी की जड़ में खान - पान ही मूल वजह है ।

यह बताना है । हैरत में डालने वाला हो सकता है ।देश में होने वाली कुल मौतों में से ५३ फीसदी की जड़ में खान - पान ही मूल वजह है ।

यह बताना हैरत में डालने वाला हो सकता है। देश में होने वाली कुल मौतों में से ५३ फीसदी की जड़ में खान - पान ही मूल वजह है ।

स्वामीनाथन ने बताया कि किसानो के जल्दी कटाई करने से फसल अच्छी हुई है ।

स्वामीनाथन ने बताया। किसानो के जल्दी कटाई की । इससे फसल अच्छी हुई है।

स्वामीनाथन ने बताया कि किसानो ने जल्दी कटाई की । इससे फसल अच्छी हुई है।

Figure 3.1 System output after improvements

figure is designed to clearly demonstrate the impact of the modifications on the quality of the simplified sentences.

3.4.1 Color Coding and Layout:

- **Input Sentence (Yellow)**: Each example begins with the original complex sentence highlighted in yellow. This color coding helps to distinguish the starting point of our simplification process.

- **Output from the Earlier System (Orange)**: Following the input, the second sentence in each example is shown in orange. This represents the output generated by the previous version of our sentence simplification system, before the implementation of the recent enhancements.

- **Enhanced Output Sentence (White)**: The final sentence in each set is displayed in white, representing the output after applying the new enhancements. This color contrast is used to emphasize the improvements in naturality and coherence achieved through the modifications.

3.5 Evaluation

A similar round of evaluation was carried out with three data sets containing sentences with 2 to 3 verb chunks, 4 to 5 verb chunks and more than 5 verb chunks respectively. The average rating of the first data set almost remained unchanged whereas there was a minor improvement from 2.0 to 2.2 in the second data set. A significant improvement in the rating of the third data set was observed which jumped from 1.2 to 2.1. Table 3.3 shows a comparison between the average ratings given to the system with and without the enhancements.

For further enhancing the system a very deep knowledge of grammar and coding of more rules is required. And with more rules, more disambiguation problems arises.

Data Set	Average Ratings (Without Improvements)	Average Ratings (With Improvements)
Data Set 1	2.5	2.5
Data Set 2	2.0	2.2
Data Set 3	1.2	2.1

Table 3.3 Average Rating Comparison

3.6 Moving Towards Statistical Approach

3.6.1 Sentence Simplification as Monolingual Machine Translation

Sentence simplification is fundamentally a process of transforming complex sentences into simpler versions. This transformation involves modifying both the lexicon and grammatical structures of the sentences while ensuring the preservation of their semantic content. Effectively, this can be likened to a form of machine translation, where the source 'language' consists of complex grammatical structures and the target 'language' is their simplified counterpart.

3.6.2 Integration of Machine Translation Techniques

Recognizing sentence simplification as a translation task allows us to leverage statistical methods from machine translation. These methods are particularly advantageous because they do not necessitate a deep understanding of grammar; rather, they rely primarily on the availability of high-quality training data. The quality of this data is crucial, as it directly influences the effectiveness of the resulting simplification model.

3.6.3 Development of Training Data

For our study, we meticulously enhanced the output of our existing rule-based system to prepare our training dataset. This dataset comprises over 20,000 manually corrected sentences, specifically designed to train our model to produce accurate and contextually appropriate simplifications.

3.6.4 Employment of Phrase-Based Machine Translation (PBMT)

We use the Moses software to train a PBMT model (Koehn et al., 2007). In general, a statistical machine translation model finds a best translation e' of a text in language f to a text in language e by combining a translation model that finds the most likely translation $p(f \mid e)$ with a language model that outputs the most likely sentence

$$e' = \underset{e \in e^*}{\arg\max} \{p(f \mid e) \cdot p(e)\}$$

Data Set	Average Ratings (Without Enhancements)	Average Ratings (With Enhancements)	Average Ratings (Using Machine Translation)
Data Set 1	2.5	2.5	2.7
Data Set 2	2.0	2.2	2.6
Data Set 3	1.2	2.1	2.5

Table 3.4 Compiled Results

3.6.5 Utilization of GIZA++ for Word Alignment

The alignment of words between complex and simplified sentences is facilitated by GIZA++, which employs IBM Models 1 to 5 and an HMM-based alignment model. These models are instrumental in identifying statistically significant alignments that inform the phrase alignment processes in the Moses pipeline.

3.6.6 Operational Workflow

- During the training phase, the GIZA++ aligner processes the pairs of complex and simplified sentences under standard settings without optimization. This process generates a phrase table that encapsulates pairs of phrases—both from complex and simplified sentence structures—and their associated conditional probabilities.

- For actual sentence simplification, the Moses decoder utilizes this phrase table to generate simplified versions of sentences in the test set, ensuring that the outputs are not only simpler but also maintain the integrity of the original messages.

By integrating advanced machine translation methodologies into the sentence simplification process, we aim to enhance the system's ability to handle a variety of complex linguistic structures effectively. This approach not only broadens the applicability of sentence simplification techniques but also improves their accuracy and reliability in producing readable and understandable texts.

3.6.7 Results

Results Table 3.4 shows how monolingual machine translation further improved the ratings given by participants.

3.7 Conclusion and Implications of the Study on Sentence Simplification

This study has critically evaluated the current systems designed for sentence simplification in Hindi, highlighting their numerous limitations and the need for a more robust and effective approach. Through our research, we have developed and proposed a rule-based methodology tailored to address these shortcomings by systematically simplifying complex sentences into multiple, simpler sentences.

3.7.1 Key Features of the Proposed Approach

- **Clause Boundary Utilization**: Our system employs a strategic analysis of clause boundaries combined with dependency parsing. This method effectively identifies the various arguments associated with verbs, facilitating a structured breakdown of complex sentences.

- **Preservation of Semantic Integrity**: By restructuring the grammatical components of sentences without altering their inherent meaning, our approach ensures that the core semantic information of the original sentences is preserved. This is crucial for maintaining the accuracy and relevance of the simplified text.

3.7.2 Integration of Phrase-Based Machine Translation

In addition to the rule-based approach, our study has also ventured into the application of phrase-based machine takes (PBMT) specifically adapted for the task of sentence simplification. This innovative application seeks to harness the strengths of machine translation frameworks to further enhance the process of sentence simplification.

3.7.3 Impact on Machine Translation

We have extensively explored how sentence simplification can positively impact the field of machine translation, particularly when dealing with complex sentence structures. Simplification can significantly improve the translation process by reducing the cognitive load on the translation system, leading to outputs that are not only more accurate but also more natural.

3.7.4 Conclusion

The advancements presented in this study signify a substantial step forward in the field of sentence simplification for Hindi. By combining rule-based techniques with advanced translation technologies, we have created a system that not only addresses the immediate linguistic challenges but also contributes to broader applications in natural language processing. The implications of these enhancements extend beyond simplification alone, offering potential improvements in related areas such as machine translation, text summarization, and automated content generation.

Overall, this thesis underscores the importance of developing sophisticated, context-aware technologies in language processing, and it sets the stage for further research and development in the optimization of sentence simplification systems.

Bibliography

Akshar Bharati, Vineet Chaitanya, Rajeev Sangal, and KV Ramakrishnamacharyulu. Natural language processing: a paninian perspective. *(No Title)*, 1995.

Akshar Bharati, Dipti Misra Sharma, Samar Husain, Lakshmi Bai, Rafiya Begum, and Rajeev Sangal. Anncorra: Treebanks for indian languages, guidelines for annotating hindi treebank (version–2.0). *LTRC, IIIT Hyderabad, India*, 2009.

Tej K Bhatia. Professor yamuna kachru and creativity in hindi linguistics. *World Englishes*, 34(1): 31–36, 2015.

Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. A multi-representational and multi-layered treebank for hindi/urdu. In *proceedings of the third linguistic annotation workshop (LAW III)*, pages 186–189, 2009.

Raman Chandrasekar and Bangalore Srinivas. Automatic induction of rules for text simplification. *Knowledge-Based Systems*, 10(3):183–190, 1997.

Raman Chandrasekar, Christine Doran, and Srinivas Bangalore. Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*, 1996.

Takao Doi and Eiichiro Sumita. Input sentence splitting and translating. In *Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 104–110, 2003.

Sambhav Jain, Naman Jain, Aniruddha Tammewar, Riyaz Ahmad Bhat, and Dipti Misra Sharma. Exploring semantic information in hindi wordnet for hindi dependency parsing. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 189–197, 2013.

Gary A Klein and Frank Kurkowski. Effect of task demands on relationship between eye movements and sentence complexity. *Perceptual and motor skills*, 39(1):463–466, 1974.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Omkar Nath Koul. *Modern Hindi Grammar*. Dunwoody Press Springfield, USA, 2008.

Vilson J Leffa. Clause processing in complex sentences. In *Proceedings of the First International Conference on Language Resources and Evaluation*, volume 1, pages 937–943, 1998.

Ryan McDonald and Joakim Nivre. Characterizing the errors of data-driven dependency parsing models. In Jason Eisner, editor, *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `https://aclanthology.org/D07-1013`.

Martha Palmer, Rajesh Bhatt, Bhuvana Narasimhan, Owen Rambow, Dipti Misra Sharma, and Fei Xia. Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure. In *The 7th International Conference on Natural Language Processing*, pages 14–17, 2009.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

C Poornima, V Dhanalakshmi, KM Anand, and KP Soman. Rule based sentence simplification for english to tamil machine translation system. *International Journal of Computer Applications*, 25(8): 38–42, 2011.

Rahul Sharma, Soma Paul, Riyaz Ahmad Bhat, and Sambhav Jain. Automatic clause boundary annotation in the hindi treebank. In *Proceedings of the 27th Pacific Asia Conference on Language, Information and Computation*, pages 499–504. Waseda University, 2013.

Advaith Siddharthan. An architecture for a text simplification system. In *Language Engineering Conference, 2002. Proceedings*, pages 64–71. IEEE, 2002.

Advaith Siddharthan. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4:77–109, 2006.

Ankush Soni, Sambhav Jain, and Dipti Misra Sharma. Exploring verb frames for sentence simplification in hindi. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1082–1086, 2013.

Lucia Specia. Translating from complex to simplified sentences. In *Computational Processing of the Portuguese Language: 9th International Conference, PROPOR 2010, Porto Alegre, RS, Brazil, April 27-30, 2010. Proceedings 9*, pages 30–39. Springer, 2010.

Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, Tsutomu Hirao, and Masaaki Nagata. Divide and translate: Improving long distance reordering in statistical machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 418–427, 2010.