CHAPTER – 1

INTRODUCTION

1.1 OVERVIEW

Semantics, as a branch of linguistics, aims to study the meaning in language. As one knows that a language exhibits a meaningful message because of the semantic interaction with the different linguistic levels phonology, lexicon and syntax. However, the field of semantics, too, contributes towards stylization. That means any discussion of the semantic features of literary style implies a discussion of the nature of the semantics in literary texts. Semantic analysis deals with the meaning of words and sentences, the ways that words and sentences refer to elements in the world. "Meaning" in these discussions is usually associated with semantic. Semantic knowledge is a method of representing knowledge. The goal is to reduce the syntactic structures and provide the meaning.

Once the computer has arrived at an analysis of the input sentence's syntactic structure, a semantic analysis is needed to ascertain the meaning of the sentence. The syntactic structure of a sentence, the NLP system will attempt to produce the logical form of the sentence. The basic or primitive unit of meaning for semantic will be not the word but the sense, because words may have different senses, like those listed in the dictionary for the same word. Thus different senses can be organized into a set of classes of objects; this representation is called ontology such as substance, quantity, quality, relation, place, time, position, state, action, and affection, events, ideas, concepts, and plans.

Actions and events are especially influential. Events are important in many theories because they provide a structure of organizing the interpretation of sentences. Actions are carried out by agents. Also important is the already mentioned notion of a situation. Ex: rat eat cat. This sentence follows the Syntactical rule, but semantically the sentence is meaningless.

The product we are trying to build is called "Semantic parser for Kannada". This product does the semantic parsing at Kaaraka level. It will find out who is playing the role of each Kaaraka in a given sentence. Translating from one language to another by doing word by word direct translation will not be meaningful. So, this product comes handy at this stage. Once we have the Kaarakas, the sentence can be translated easily and meaningfully. This product will be useful to any person interested in languages and grammar.

Kaaraka is the name given to a relation subsisting between noun and a verb. There are 6 kaarakas mentioned by Paanini, namely:

- Kartaa (Doer)
- Karma (Experiencer)
- KaraNa (Instrument)
- Sampradaana (Recipient)
- Apaadaana (Separation)
- AdhikaraNa(location)

We added "Destination" to the above list.

1.1.1 VIBHAKTHI:

Case (ವಿಭಕ್ತಿ)

Kannada has seven cases:

- Nominative case
- Accusative case (ಕರ್ಮವಿಭಕ್ತಿ)
- Instrumental-ablative case (ಕರಣವಿಭಕ್ತಿ)
- Dative case (ಸಂಪ್ರದಾನವಿಭಕ್ತಿ)
- Genitive case (ಸಂಭಂದವಿಭಕ್ತಿ)
- Locative case (ಅಧಿಕರಣವಿಭಕ್ತಿ)
- Vocative case (ಸಂಬೋಧನಾವಿಭಕ್ತಿ).

Because the traditional study of Kannada grammar is based on Sanskrit grammar, a fifth case (since the dative case is the fourth case and the genitive case is the sixth in the traditional order of the cases) is sometimes considered: the ablative case (ಅಪಾದಾನವಿಭಕ್ತಿ).

This case is formed periphrastically by combining the genitive case of the noun supposedly in the ablative with the instrumental-case form of the noun 'ದೆಸೆ', meaning 'cause, vicinity, place, point'. Thus the Kannada ablative literally translates to 'from/by the cause/point of the {noun}'. However, this 'ablative' form is not commonly used colloquially, and exists only for propriety—it is not a true case, serving only to provide a parallel to the Sanskrit ablative. In its place, the third case, the instrumental-ablative case, is normally used

The **nominative case** (abbreviated **NOM**) is one of the grammatical cases of a noun or other part of speech, which generally marks the subject of a verb or the predicate noun or predicate adjective, as opposed to its object or other verb arguments. Generally, the noun "that is doing something" is in the nominative, and the nominative is the dictionary form of the noun.

The reference form (more technically, the *least marked*) of certain parts of speech is normally in the nominative case, but this is often not a complete specification of the reference form, as it may also be necessary to specify the number and gender. Thus the reference or least marked form of an adjective might be the nominative masculine singular. The parts of speech which are often declined and therefore may have a nominative case are nouns, adjectives, pronouns and less frequently numerals and participles. The nominative case often indicates the subject of a verb but sometimes does not indicate any particular relationship with other parts of a sentence. In some languages the nominative case is unmarked, it may be said to be marked by a zero morpheme. Moreover, in most languages with a nominative case, the nominative form is the lemma; that is, it is the reference form used to cite a word, to list it as a dictionary entry, etc.

Nominative cases are found in Arabic, Estonian, Slovak, Ukrainian, Hungarian, Lithuanian, Georgian, German, Latin, Greek, Icelandic, Old English, Old French, Polish, Serbian, Czech, Romanian, Russian, and Pashto, among other languages. English still retains some nominative pronouns, which are contrasted with the accusative (comparable to the oblique or disjunctive in some other languages): *I* (accusative *me*), *we* (accusative *us*), *he* (accusative *him*), *she* (accusative *her*), *they* (accusative *them*) and *who* (accusative *whom*). A usage that is archaic in most, but not all, current English dialects is the singular second-person pronoun *thou* (accusative *thee*). A special case is the word *you*: Originally, *ye* was its nominative form and *you* the accusative, but over time *you* has come to be used for the nominative as well.

The term "nominative case" is most properly used in the discussion of nominative–accusative languages, such as Latin, Greek, and most modern Western European languages.

In active–stative languages there is a case sometimes called nominative which is the *most* marked case and is used for the subject of a transitive verb or a voluntary subject of an intransitive verb but not for an involuntary subject of an intransitive verb; since such languages are a relatively new field of study, there is no standard name for this case.

The **accusative case** (abbreviated ACC) of a noun is the grammatical case used to mark the direct object of a transitive verb. The same case is used in many languages for the objects of (some or all) prepositions. It is a noun that is having something done to it, usually used together (such as in Latin) with the nominative case. For example, "they" in English is nominative; "them" is accusative. The sentence "They like them" clearly shows the nominative case and accusative case working in conjunction using the same base word. The syntactic functions of the accusative consist of designating the immediate object of an action, the intended result, the goal of a motion, and the extent of an action.

The accusative case existed in Proto-Indo-European and is present in some Indo-European languages (including Latin, Sanskrit, Greek, German, Polish, Romanian, Russian, Ukrainian), in the Uralic languages, in Altaic languages, and in Semitic languages (such as Hebrew and Classical Arabic). Finnic languages, such as Finnish and Estonian, have two cases to mark objects, the accusative and the partitive case. In morphosyntactic alignment terms, both perform the accusative function, but the accusative object is telic, while the partitive is not.

Modern English, which almost entirely lacks declension in its nouns, does not have an explicitly marked accusative case even in the pronouns. Such forms as *whom*, *them*, and *her* derive rather from the old Germanic dative forms, of which the -m and -r endings are characteristic. This conflation of the old accusative, dative, instrumental, and (after prepositions) genitive cases is the *oblique case*. Most modern English grammarians no longer use the Latin accusative/dative model, though they tend to use the terms *objective* for oblique, *subjective* for nominative, and *possessive* for genitive *(see Declension in English). Hine*, a true accusative masculine third person singular pronoun, is attested in some northern English dialects as late as the 19th century.

The **instrumental case** (abbreviated **INS** or **INSTR**; also called the *eighth case*) is a grammatical case used to indicate that a noun is the *instrument* or means by or with which the subject achieves or accomplishes an action. The noun may be either a physical object or an abstract concept.

Modern English expresses the instrumental meaning by use of adverbial phrases that begin with the words *with*, *by*, or *using* then followed by the noun indicating the *instrument*:

I wrote the note with a pen.

I wrote the note (by) using a pen.

Technical descriptions often use the phrase "by means of", which is similar to "by use of", as in:

I wrote the note by means of a pen.

I wrote the note by use of a pen.

This can be replaced by "via", which is a Latin ablative of the nominative (viā) *via*, meaning road, route, or way. In the ablative this means *by way of*.

The **dative case** (abbreviated **DAT**, or sometimes **D** when it is a core argument) is a grammatical case generally used to indicate the noun to which something is given, as in "Maria gave **Jacob** a drink". Here, Jacob is an indirect dative.

In general, the dative marks the indirect object of a verb, although in some instances, the dative is used for the direct object of a verb pertaining directly to an act of giving something. This may be a tangible object (e.g., "a book" or "a tapestry"), or an intangible abstraction (e.g., "an answer" or "help").

Sometimes the dative has functions unrelated to giving. In Scottish Gaelic and Irish, the term *dative case* is used in traditional grammars to refer to the prepositional case-marking of nouns following simple prepositions and the definite article. In Georgian, the dative case also marks the subject of the sentence with some verbs and some tenses. This is called the dative construction.

The dative was common among early Indo-European languages and has survived to the present in the Balto-Slavic branch and the Germanic branch, among others. It also exists in similar forms in several non-Indo-European languages, such as the Uralic family of languages, and Altaic languages. In some languages, the dative case has assimilated the functions of other now-extinct cases. In Ancient Greek, the dative has the functions of the Proto-Indo-European locative and instrumental as well as those of the original dative.

Under the influence of English, which uses the preposition "to" for both indirect objects (*give to*) and directions of movement (*go to*), the term "dative" has sometimes been used to describe cases that in other languages would more appropriately be called lative.

In grammar, **genitive** (abbreviated **GEN** also called the **possessive case** or **second case**) is the grammatical case that marks a noun as modifying another noun. It often marks a noun as being the possessor of another noun; however, it can also indicate various other relationships than possession: certain verbs may take arguments in the genitive case, and it may have adverbial uses (*see* Adverbial genitive).

Placing the modifying noun in the genitive case is one way to indicate that two nouns are related in a genitive construction. Modern English typically does not morphologically mark nouns for a genitive case in order to indicate a genitive construction; instead, it uses either the *'s* clitic or a preposition (usually *of*). However, the personal pronouns do have distinct possessive forms. There are various other ways to indicate a genitive construction, as well. For example, many Afroasiatic languages place the head noun (rather than the modifying noun) in the construct state.

Locative (abbreviated **LOC**) is a grammatical case which indicates a location. It corresponds vaguely to the English prepositions "in", "on", "at", and "by". The locative case belongs to the general local cases together with the lative and separative case.

The locative case exists in many language groups.

The **vocative case** (abbreviated **VOC**, voc.) is the case used for a noun that identifies a person (animal, object, etc.) being addressed or, occasionally, the determiners of that noun. A vocative expression is an expression of direct address where the identity of the party spoken to is set forth expressly within a sentence. For example, in the sentence, "I don't know, John", *John* is a vocative expression that indicates the party being addressed—as opposed to the sentence, "I don't know John", where *John* is the direct object of the verb "know."

1.2 GOALS

The non-technical goals to be achieved by this project are:

- Given a Kannada sentence, the program should find the Kaaraka by doing the semantic level mapping.
- To determine the Kaarakas from the given sentence depending on the set of rules associated with the mappings.
- To handle the ambiguities based on two methods:
  - Aakanksha (Expectation) of the Verb.
  - Yogyatha (Eligibility) of the Noun.

The technical goals to be achieved by this project are:

- To process each Sentence from the Input file one by one, along with the Morphological Output of that Sentence, which is Input to the program, Categorize words in each sentences as Naamapada and Kriyapada along with the Tag set of that word.
- Find the Aakansha of each verb in a sentence and assign the associated Kaaraka expected for that verb.
- Assign the score to each Naamapada in the Sentence based on the rules and delete the non-matching Kaaraka for that Naamapada in the Sentence.
- Represent the Kaarakas in GUI form by having the verb in a sentence at the center and related Kaaraka for the Namapada in sentence around the verb.

1.3 MOTIVATION

Kannada Language being one of the major Dravidian languages of India and it has 27th place in most spoken language in the world. But still it does not yet have Semantic checking methods for a given Kannada sentence. When Computational Linguistic is concerns Kannada is lagging far behind compared to Telugu and Kannada. Writing the Semantic level Parser for any south Indian language is bit difficult. Because the languages are highly inflected with three gender forms and two number forms. In most of the Indian languages including Kannada a verb ends with a token which indicates the gender of the person (Noun/ Pronoun).
Therefore the Morphological Parsed Sentence (Input to the program) will give the tag-set for each Word in the Sentence.
The Kannada example below shows a plain sentence and its tagged form as generated by the saara system.

raamanu manege hoodanu

raamanu||raama||N-PRP-PER-M.SL-NOM manege||mane||N-COM-COU-N.SL-DAT

hoodanu||hoogu||V-IN-ABS-PAST-P3.M.SL Here, for each word form, its root form and the tag are shown, separated by two vertical bars. The tag for the first word indicates that the first word is a noun, it is a proper noun, it indicates a (singular) person in masculine gender, and in nominative case. The second tag should be read as noun-common-countable-neuter_gender-singular-dative_case. The third tag should be read as verb intransitive (akarmaka)-absolute_aspect-third_person-masculine - singular in agreement. Thus tagged sentences contain a lot of useful information for carrying out various NLP tasks. Based on the Morphological Parser output, the mapping from Vibhaktis to Kaaraka is done by the Algorithm which we have implemented.

1.4 PROJECT SCOPE

- The program Input (100 Sentence file) is in both Unicode (Kannada) as well as Roman (English) format.
- Metaphoric language cannot be given as an Input to the Parser.

1.5 METHODOLOGY

- The File which Consists of 100 Kannada Sentences with the Morphological Tags for each word in the Sentence is the Input to the Program.
- Based on the Tag-Set, the words are classified as Nouns and Verbs.
- Create a datastructure for each Naamapada and Kriyapaada.
- The datastructure is in this form:
  Naamapada : { person : p1, gender: M, count: SL, anim: PROH }
- Read the yaml file which has the kriyapaada and expected Kaarakas in JSON format and then classify the verbs and associated Aakanksha

  Aakanksha can be as follows:

  - Kartaa (Doer)
  - Karma (Experiencer)
  - KaraNa (Instrument)
  - Sampradaana (Recipient)
  - Apaadaana (Separation)
  - AdhikaraNa(location)
  - Destination
- Process Each Naamapada in the Sentence and Set Scores for the Kaarakas based on the rules (Algorithm).
- Then perform Constraint Propagation by setting the Score to “0” to non-matching Kaaraka once all the Naampada’s are processed.
- Kaaraka backtracking if the verb expects any other Kaaraka from previous naamapada.
- Then once the Kaaraka is Identified for each Naamapada in the sentence, the output is shown in the GUI format with the Kriye(Verb) of the Sentence placed at the center encircled by the Kaarakas of that Sentence.

CHAPTER – 2

PROBLEM DEFINITION

Kannada is one of the four major Dravidian languages, having almost forty million speakers and its own independent script and long documented histories. Kannada has been estimated to be over 2, 500 years old and the third oldest Indian language after Sanskrit and Tamil. To create any natural language processing tool requires the application of many types of knowledge, such as lexical, morphological, syntactical, semantic etc., in order to resolve different kind of ambiguities. Our project aims at processing each Kannada sentence with the Morphological tags (Vibhatkis) to associated Kaaraka for those Vibhatkis which helps in Translation from Kannada to other Language, as Word-to-Word translation does not work in Kannada language, as the meaning and Context of the words varies from Kannada to other Language(Telugu). Therefore this approach helps in finding the Kaarakas for Naamapadas in the sentence which can result in lots of Ambiguities which is handled through:

- Eligibility of the Noun (Yogyatha).
- Expectation of the Verb (Eligibility).

CHAPTER-3

LITERATURE SURVEY

The karakas are recognized by most scholars as basic semantic notions that pivot sentence constructions. They are similar to the case roles/relations proposed in the case grammars. But karakas are much more than these, and their crucial role as a common substratum of ontology, cognition, and grammar can be understood only if we regard them as a manner of classifying 'actions' in the real world.

It may not be however inappropriate to suggest that the karaka notions are conceived as properties of the world corresponding to, though independent of their grammatical/morphological manifestations. Panini himself was probably merely projecting the karakas (literally, 'a factor of action') from morphological occurrences in the form of cases to a set of possible actions in the world. This point has been most aptly made by a recent commentator:

Panini identifies six karakas corresponding to six cases, viz., the nominative, accusative, dative, instrumental, locative, and ablative. Possessive and vocative are conspicuous by their absence in Panini's grammar. This is how Panini defines the six karakas (Ashtadhyayi, I.4.24-54):

1. Apadana (lit. 'take off'): "(that which is) firm when departure (takes place). This is the equivalent of the ablative notion which signifies a stationary object from which movement proceeds.
2. Sampradana ('bestowal'): "he whom one aims at with the object". This is equivalent to the dative notion which signifies a recipient in an act of giving or similar acts.
3. Karana ("instrument") "that which effects most". This is equivalent to the instrumental notion.
4. Adhikarana ('location'): or "substratum". This is equivalent to the locative notion.
5. Karman ('deed'/'object'): "what the agent seeks most to attain". This is equivalent to the accusative notion.
6. Karta ('agent'): "he/that which is independent in action". this is equivalent to the case of the subject or the nominative notion.

Nominative Case (prathamaa vibhakti): Nominative Case (prathamaa vibhakti) represents the noun-form to represent the "who" or "what" on the *verb* in the sentence. In other words *nominative case* represents the *subject* or kartaa of the sentence.
The following sentences are in *nominative case*.

1. The boy is going.
2. Poet is writing.
3. Teachers are speaking.
4. Girl is reading.
5. River is flowing.
6. I am asking.
7. Sita is singing.
8. Fruit is falling.
9. Vehicle is moving.
10. The book is there.
11. There is no water.
12. Flower is blooming.

Table 3.1 Examples demonstrating the Prathama Vibhatki

In sentence 1 above, if we ask the question "who is going?", the answer that comes is "boy". So, boy is the *subject* in the sentence and the noun-form is in *nominative case* or *prathamaa vibhakti*. Similarly in sentence 8, if we ask the question "what is falling? The answer that comes is "fruit". In this fruit is the *subject*.

So, answer to the question "who" or "what" is the *subject* of the sentence and is always in *nominative case*. The verb form follows the *number* of the noun. The followings are few rules where the *nominative case* must be used as the noun-form. In active voice statements the subject in the sentence is always in nominative case. In passive voice statements the object follows the nominative case. The subject follows the instrumental case.

Accusative Case (dvitiiyaa vibhakti): Accusative Case (dvitiiyaa vibhakti) of noun-form represents the "to what" or "to whom" of the sentence. In other words *accusative case* represents the *object* or Karma in the sentence.
The following sentences are in *accusative case*

1. The painter is painting a portrait.
2. The carpenter is chopping wood.
3. Poets are writing songs.
4. Enemies are attacking the city.
5. Students are singing the stanzas.
6. Committee gave the permission.
7. Boy is going to school.
8. Girl is reading the book.

Table 3.2 Examples demonstrating Dvithaya Vibhatki

In sentence 1 above, if we ask the question "what is the painter painting?", the answer that comes is "the portrait". So, "portrait" is the *object* in the sentence and the noun-form is in *accusative case* or *dvitiiyaa vibhakti*. Similarly in sentence 6, if we ask the question "what did the committee gave?", the answer that comes is "permission". In case of sentence 7, if we ask the question "where is the boy going to?", the answer that comes is "school".
So, answer to the question "to what", "to whom" or "to where" etc., is the *object* in the sentence and is always in *accusative case*. The verb form is independent of the *number* or वचन (vachana) of the *object*.

In active voice sentences the object always in accusative case.

Instrumental Case (tRRitiiya vibhakti): Instrumental Case or tRRitiiya vibhakti of *noun-form* represents the instrumental form in a sentence. The answer that comes from the question "by what" or "with what" is the *instrument* in the sentence. The *instrument* is with what the *subject* or kartaa is doing something. The *instrument* is independent of the *number* and gender of the *subject* or *object*.

The following sentences are in *instrumental case*.

1. Farmer is ploughing with plough.
2. I am cutting with axe.
3. Teacher wrote using pen.
4. You are touching with hand.
5. Snakes bites with teeth.
6. Guest is happy with food.
7. Garland is made using flower.
8. You live with knowledge.
9. (I) will decorate God with garland.

Table 3.3 Examples Demonstrating Tritiya Vibhatki

In sentence 1 above, if we ask the question "what is the farmer is ploughing with?", the answer that comes is "with the plough". So, "plough" is the *instrument* in the sentence and the noun-form is in *instrumental case* or *tRRitiiya vibhakti*. Similarly in sentence 4, if we ask the question "what are you touching with?", The answer that comes is "with hand". In case of sentence 6, if we ask the question "what is the guest happy with?" the answer that comes is "with food". So, hand and food are in *instrumental case*.

<u>Grammatical Rule</u>

The word expressing "by what" or "with what" pertaining to the verb will be in instrumental case.

Sentences 1 and 9 also follow this rule.

Besides the above rules there are some special rules where *instrumental case* is used. Followings are the examples of these.

10. Women played with the children.

11. Without effort knowledge is not achievable.

12. Without water life is impossible.

13. What is life without knowledge?

14. He is blind with eyes.

21. What shall mirror do to one without eyes?

22. He is deaf with ears.

23. He is lame with foot.

24. Rama is naughty by nature.

25. He is running speedily.

26. Rama is living happily.

27. Named Rama

28. The girl is beautiful by face.

29. Simple by nature.

30. Hunters are normally poor.

31. Let the boy go quickly.

32. They came in one queue.

33. I am kshatriya (warrior) by caste.

34. Boys are roaming by their wish.

35. Rama is younger to you by one year.

36. Gopala is younger to Shyama by a month.

37. He is elder to me by a month.

38. Sage by matted lock.

39. Brahmin by sacred-thread?

40. Body shivers due to cold.

41. He is heartbroken due to sadness.

42. Hari is seen through merit.

43. He is suffering due to fever.

44. He is crying due to hunger.

45. I have need of study.

46. What is the need of quarrelling?

47. Quarrelling is waste.

48. Do not have need in wealth?

49. Less by knowledge.

50. Less by wealth.

51. He is less by wealth.

52. The temple was built in a year.

53. Rama read Sanskrit in a month.

54. I bought the book with five coins.

55. He bought the toy with ten coins.

56. The lustful person is giving money to the maid.

57. Rama is going on a different route.

Table 3.4 Example demonstrating Tritiya Vibhatki

Grammatical Rule

Grammatical Rule

In sentence 17 - the word "effort" is in *instrumental case* as the word is used to express that without effort knowledge cannot be achieved. Sentences 18 and 19 also follow this rule.

Grammatical Rule

If any word expressing lameness in any organ or body part is used then the word representing the organ or body part will be in instrumental case.

In sentence 20 - the word "eyes" is in *instrumental case* as this is the organ that has the lameness. Sentences 21, 22 and 23 also follow this rule.

Grammatical Rule

The word expressing the nature or characteristics of someone or something will be in instrumental case.

In sentence 24 the word "nature" represents a characteristics of Rama. So, the word "nature" or is in *instrumental case*. Similarly in sentence 25 the word "speedily" is an attribute or characteristics of him running. Hence the word speedily or is in *instrumental case*. In sentence 33 also the word caste represents an attribute about me. So this word is in *instrumental case*. Sentences 24 to 34 follow this rule.

<u>Grammatical Rule</u>

The word expressing any mark of identification of a person will be in instrumental case.

In sentence 38 - sage is identified by "matted lock". In other words "matted lock" is the identification symbol to identify a sage. So, the word "matted lock" is in *instrumental case*. Sentences 39 also follows this rule.

<u>Grammatical Rule</u>

The word expressing the cause of something will be in instrumental case.

In sentence 40 "cold" is the reason for which body shivers. So, the word "cold" is in *instrumental case*. Similarly, in sentence 41 "sadness" is the reason for his broken heart. Hence the word is in *instrumental case*. Sentences 42, 43 and 44 also follow this rule.

<u>Grammatical Rule</u>

In sentence 45 "study is what I have need for". So, the word study is in *instrumental case*. Similarly in sentence 47 - "quarrelling is waste" also means there is no need or necessity of quarrelling. Hence the word quarrelling or is in *instrumental case*. Sentences 46 and 48 also follow this rule.

<u>Grammatical Rule</u>

In sentence 49 - there is a lack of "knowledge". Similarly in sentence 50 there is a lack of "wealth". So, these words are in *instrumental case*. Sentence 51 also follows this rule.

<u>Grammatical Rule</u>

If the sentence means achievement of some result after some time then the word expressing the time will be in instrumental case.

In sentence 52 the temple was built in a year. In other words the result was achieved in a year. The word "year" is expressing the time after which the result was achieved. So, the word "year” is in *instrumental case*. Sentence 53 also follows this rule.

<u>Grammatical Rule</u>

The word expressing the value or cost of something will be in instrumental case.

In sentence 54 "five coins" express the cost or value of the book. So, it is in *instrumental* *case*. Similarly in sentence 55 "ten coins" is the cost of the toy. So, "ten coins" is in *instrumental case*.

Grammatical Rule

The word expressing the person to whom something is given to perform some immoral or indecent act will be in instrumental case and not in dative case.

In sentence 56 the lust person is giving money to the maid in return of an immoral. So, the word "maid" is in *instrumental case* instead of *dative case*.

Grammatical Rule

Word expressing the path that is followed will be in instrumental case.

Dative Case / (chaturthii vibhakti): Dative Case (chaturthii vibhakti) of noun-form represents the "to whom" or "for whom" of the sentence. In other words *dative case* represents the *dative* in the sentence.

The following sentences are in *dative case*.

1. Give rice to the beggar.
2. I am giving prize to him.
3. Doctor is giving medicine to the patient.
4. Let the rich give money to poor.
5. The lustful person is giving money to the maid.

Table 3.5 Examples demonstrating Chaturthi Vibhatki

In sentence 1 above, if we ask the question "give rice to whom?", the answer that comes is "the beggar". So, "beggar" is the *dative* in the sentence and the noun-form is in *dative case* or *chaturthii vibhakti*. Similarly in sentence 3, if we ask the question "doctor gave medicine to whom? the answer that comes is "patient". So, answers to the question "to whom" or "for whom" etc., is the *dative* in the sentence and is always in *dative case*. The verb form is independent of the *number* or वचन (vachana) of the *subject* or *object*.

<u>Grammatical Rule</u>

The word expressing the person to whom something is given will be in dative case.
Sentence 1, 2, 3 and 4 follow this rule.

<u>Grammatical Rule</u>

The word expressing the person to whom something is given to perform some immoral or indecent act will be in instrumental case and not in dative case.
In sentence 5 the lust person is giving money to the maid in return of an immoral. So, the word "maid" is in *instrumental case* instead of *dative case*. Besides the above rules there are few special rules where *dative case* is used. Followings are examples of these.

6. Let the author write for the paper.
7. Gold is for necklace.
8. Wife should be accepted for merit/dharma.
9. Wealth becomes egoism.
10. Power is to torment others.
11. Let power be for others' good.
12. Lakshmi hates the knowledgeable.
13. He hates the clever.
14. King is angry on the enemies.
15. Gopala is angry on Rama.
16. Do you like sweets?
17. Girls like flowers.
18. Let everyone like knowledge.
19. Ladies like flowers.
20. I like milk.
21. Child likes to play.
22. Who does not like sweets?
23. Salute to teacher.
24. Salute to Narayana.

25. Offered to Indra.

26. Offered to fire.

27. Let good happen to all.

28. Let the guest come home.

29. I will go to village.

30. Shyama is borrowing one hundred from Rama.

31. I do not owe anything to anyone.

32. (Lord Vishnu) Hari owes liberation to worshippers.

33. Mother is showing moon to son.

34. Tell Gangadatta about me.

Table 3.6 Table demonstrating Chaturthi Vibhaktis

Grammatical Rule

The word expression the purpose for which certain action is taken or something is needed will be in dative case.

Sentence 6 above means the author should write for the paper. In other words the writing action is for the "paper". So, paper takes the *dative case*. Similarly in sentence 7 gold is needed for necklace. So, necklace is in the *dative case*. Sentence 8 also follows the same rule.

Grammatical Rule

The word expressing the object into which another object transforms will be in dative case.

In sentence 9 above - wealth turns or transforms into egoism. So, "egoism" is in *dative case* the object into which wealth transforms. Sentence 10 and 11 also follow the same rule.

Grammatical Rule

The word expressing the person (or thing) against whom (or which) anger or hatred is shown will be in dative case.

In sentence 12 above - Laxmi shows hatred towards the "knowledgeable". So, the word knowledgeable is in *dative case*. Similarly in sentence 14 the King shows anger towards enemies. So, the word enemy is in *dative case*. Sentence 13 and 15 also follow the same rule.

<u>Grammatical Rule</u>

when verbs meaning "liking" are used, the word expressing the person to whom it is a matter of liking will be in dative case.

In sentence 17 above - girls like flowers or in other words flowers are a matter of liking to the girls. So, "girl" is in *dative case*. Similarly in sentence 21 play is a matter of liking to "child". So, the child is in *dative case*. Sentence 16, 18, 19, 20, and 22 also follow the same rule.

<u>Grammatical Rule</u>

In sentence 23 above "teacher" is the object of reference of the word salute. So, the word teacher is in *dative case*. Sentence 24, 25, 26 and 27 also follow the same rule.

<u>Grammatical Rule</u>

When verbs meaning going or coming are used the words expressing the destination will be either in accusative case or dative case. In sentence 28 above "home" is in *dative case* (or *accusative case*) as home is the destination. Sentence 29 also follows the same rule.

<u>Grammatical Rule</u>

When something is borrowed from others the person from whom the thing is borrowed will be in dative case. In sentence 30 above Shyam has borrowed money from Rama. So, "Rama" is in *dative case*. Similarly in sentence 32 Hari (Lord Vishnu) borrows from the worshipper. So, the word worshipper is in *dative case*. Sentence 31 also follows the same rule.

<u>Grammatical Rule</u>

The person with whom the subject relates something through his or her action will be in dative case. In sentence 33 above mother is showing the moon to son. In other words mother the *subject* in the sentence with her action of showing is relating son with the moon. So, the word son is in *dative case*.

Ablative Case / (paJNchamii vibhakti): Ablative Case (paJNchamii vibhakti) of noun-form represents the "from whom/what" of the sentence. In other words *ablative case* represents the *ablative* in the sentence.

These are in *ablative case*.

1. Leaf fell from the tree.
2. Fruits are falling from the trees.
3. He fall from the running horse.
4. I am coming from the school.
5. Gods come from heaven.

Table 3.7 Example demonstrating Panchami Vibhatki

In sentence 1 above, if we ask the question "leaf fell from what?", the answer that comes is "the tree". So, "tree" is the *ablative* in the sentence and the noun-form is in *ablative case* or *paJNchamii vibhakti*. Similarly in sentence 4, if we ask the question "where am I coming from?", the answer that comes is "school".
So, answer to the question "from whom/what/where" is the *ablative* in the sentence and is always in *ablative case*. The *verb-form* is independent of the *number* of the *subject* or *object*. Followings are the rules where *ablative case* is used.

<u>Grammatical Rule</u>
When an object is separated from another the word expressing the object from which the separation happened will be in ablative case.
Sentence 1, 2 and 3 follow this rule.
<u>Grammatical Rule</u>
When someone or something is coming from a place the word expressing the source will be in ablative case.
Sentence 4 and 5 follow this rule.

Genitive Case (ShaShThii vibhakti): Genitive Case (ShaShThii vibhakti) of *noun-form* represents the "whose" of the sentence. In other words *genitive case* represents the *genitive* in the sentence.

The following sentences are in *genitive case*.

1. Son of Dasaratha.
2. Krishna's friend.
3. Rise of sun.
4. Water of river.
5. Whiteness of moon.
6. Heat of fire.
7. Vrihaspati is the teacher of Gods.
8. Demons are the disciples of Sukracharya.
9. Women's jewellery.
10. This is my opinion.
11. Ganapati has four hands.

Table 3.8 Example demonstrating the Shashti Vibhatkis

In sentence 1 above, if we ask the question "whose son?", the answer that comes is "Dasaratha". So, "Dasaratha" is the *genitive* in the sentence and the *noun-form* is in *genitive case* or *ShaShThi vibhakti*. Similarly in sentence 4, if we ask the question "whose water?", the answer that comes is "river".

So, answer that comes from the question "whose" is the genitive in the sentence and is always in *genitive case*. Followings are the rules where *genitive case* is used.

<u>Grammatical Rule</u>

The word denoting a person or thing whose relationship with another is being expressed will be in genitive case.

In sentence 1 to 11 the words expressing relationship to someone or something are in *genitive case*.

Besides the above rules there are few more special rules where *genitive case* is used.

Followings are the examples of these.

Locative Case / (saptamii vibhakti): Locative Case (saptamii vibhakti) of *noun-form* represents the "where" of the sentence. In other words *locative case* represents the *locative* in the sentence.

These are in *locative case*.

1. Lotuses are in the lake.
2. Flowers bloom in the creeper.
3. Elephants roam in the forest.
4. Offices are in the city.
5. Child is sleeping on the bed.
6. Leaves fall on the ground.
7. There are many stories in Mahabharata.
8. Atma is in everybody.

Table 3.9 Examples demonstrating the Sapthami Vibhatkis

In sentence 1 above, if we ask the question "where are the lotuses?", the answer that comes is "lake". So, "lake" is the *locative* in the sentence and the *noun-form* is in *locative case* or *saptamii vibhakti*. Similarly in sentence 3, if we ask the question "elephants roam where? the answer that comes is "forest".

So, answer that comes from the question "where" the locative in the sentence is and is always in *locative case*. Followings are the rules where *locative case* is used.

<u>Grammatical Rule</u>
The word expressing the location of something or someone will be in locative case.
Sentence 1 to 8 follow this rule.
Besides the above rules there are few more special rules where locative case is used.
Followings are the examples of these.

9. He is sitting on the seat.

10. Students are running on the road.

11. Peacocks are dancing on the mountain.

12. Young boys are swimming in the river.

13. Girls are playing in the playground.

14. I will go in the evening.

15. Holiday is on Sunday.

16. When sun rises, lotus blooms.

17. When moon rises, lilies bloom.

18. Among animals human beings are the best.

19. Do good as if death is holding by hair.

20. Friend made me sit by holding my hands.

21. Father loves son.

22. You trust me.

23. Respect father.

Table 3.10 Examples demonstrating the Sapthami Vibhatkis

Grammatical Rule

The word expressing something on which someone or something sits or stands will be in locative case.

In sentence 9 if we ask the question "where is he seating?" or "on what is he seating?", the answer that comes is on the seat. So, the word "seat" is in *locative case* as it is expressing the thing on which someone is seating. Sentence 10 to 13 also follow this rule.

Grammatical Rule

The word expressing the time of action in response to the question on verb as "when" will be in locative case. In sentence 14 answer to the question "when shall I go?" is "in the evening" or "evening". So, the word "evening" is in *locative case*. Sentence 15 also follows this rule.

<u>Grammatical Rule</u>

When the action of one results in action of another, the thing whose action occurs first will be in locative case. In sentence 16 - lotus blooms when sun rises. In other words the action of sun's rising results in the action of the lotus blooming. So, the word "sun" is in *locative case*. Sentence 17 also follow this rule.

<u>Grammatical Rule</u>

If a comparison in a group is being made then the word expressing the group will be in locative case or in genitive case. In sentence 18 the word "animal" is in *locative case* (or alternatively *genitive case*), as it represents the group in which human beings are the best.

<u>Grammatical Rule</u>

If a part of body is held separately then the word expressing the part will be in locative case. In sentence 20 - my friend is holding my hand (separately) to make me seat. So, the word "hand" is in *locative case* as this is the part of body which is being held separately. Sentence 19 also follows this rule.

<u>Grammatical Rule</u>

If verbs meaning trust, fondness, love, anger, worship etc., are used, the person in whom such trust or fondness is shown will be in locative case.
In sentence 21 the verb is used to express father's love in his son. So, the word "son" is in *locative case*. Sentence 22 and 23 also follow this rule.

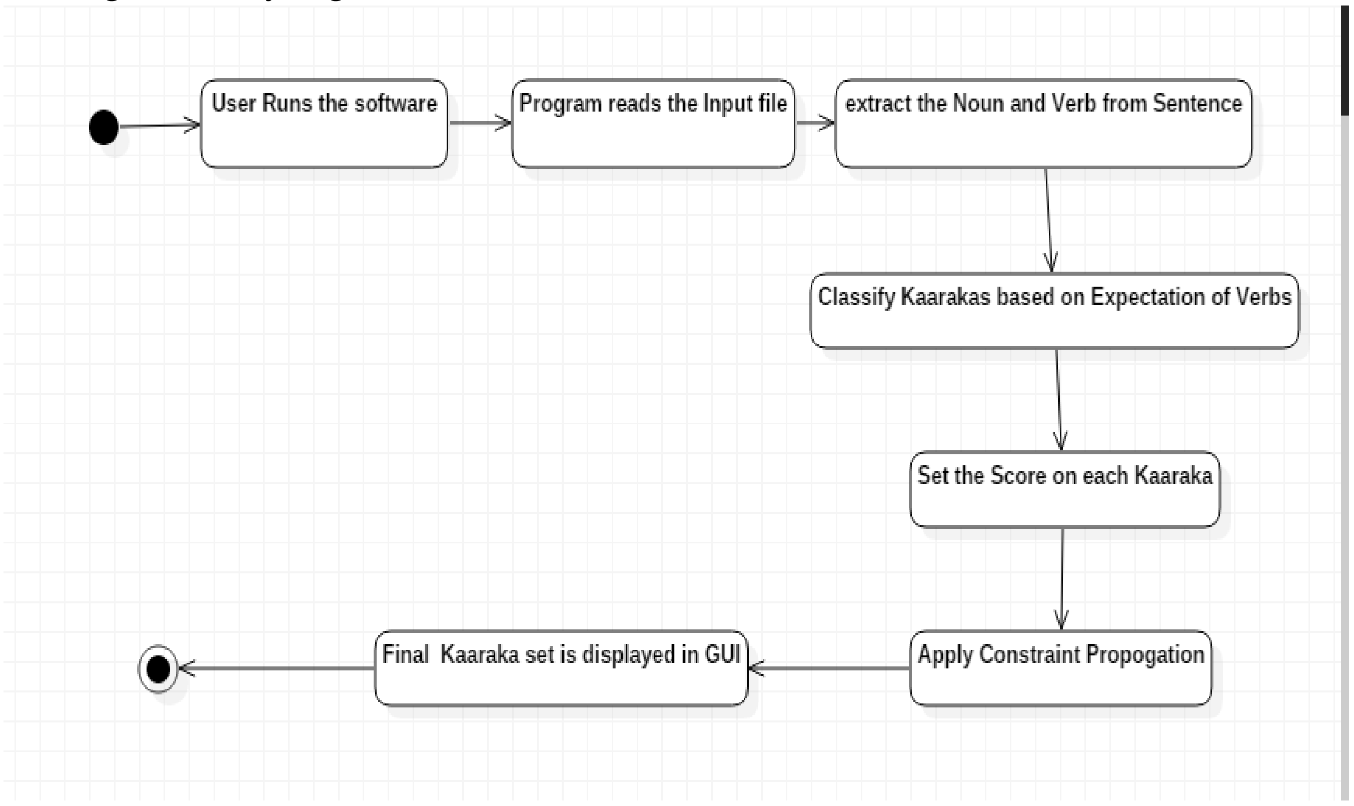
CHAPTER-4

PROJECT REQUIREMENT SPECIFICATION

4.1 ACTIVITY DIAGRAM

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Below is the activity diagram of the project “Semantic Parser for Kannada”.

Fig 4.1 Activity diagram for the Semantic Parser


User Runs the software
Program reads the Input file
extract the Noun and Verb from Sentence
Classify Kaarakas based on Expectation of Verbs
Set the Score on each Kaaraka
Apply Constraint Propogation
Final Kaaraka set is displayed in GUI

4.2 WORKFLOW DIAGRAMS

Workflow may be a view or representation of real work, thus serving as a virtual representation of work. The flow being described may refer to a service, document or product being transferred from one step to another.

4.2.1 Manually Coded Modules:

These are the modules that were implemented by the developing team of the project and they reflect the functionality of the project. These modules are shown below.

4.2.1.1 Reading input from the file

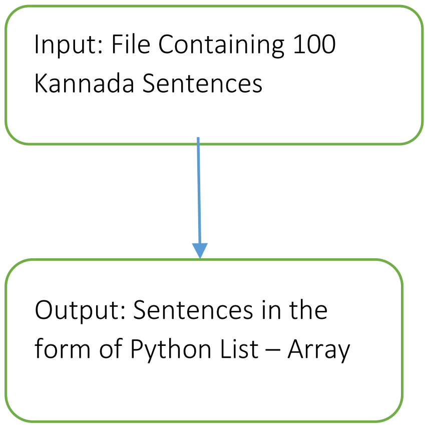

Input: File Containing 100 Kannada Sentences
Output: Sentences in the form of Python List – Array

Fig 4.2 Reading the text file which consists of Kannada sentences with tags

4.2.1.2 Process each Naamapada and Kriyapada from the sentence

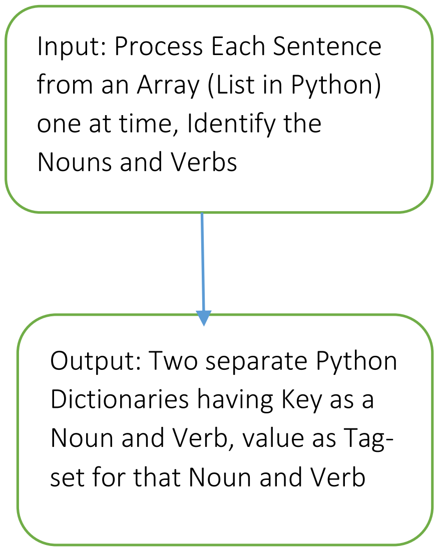

Input: Process Each Sentence from an Array (List in Python) one at time, Identify the Nouns and Verbs
Output: Two separate Python Dictionaries having Key as a Noun and Verb, value as Tag-set for that Noun and Verb

Fig 4.3 Identify the Nouns and Verbs from the tags

4.2.1.3 Create a data structure for Noun and Verb

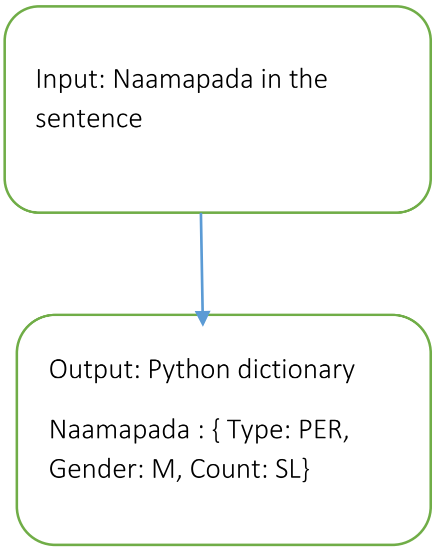

Input: Naamapada in the sentence
Output: Python dictionary
Naamapada : { Type: PER, Gender: M, Count: SL}

Fig 4.4 Data structure which has the attributes for each naamapada and kriyapaada

4.2.1.3 Get the Aakanskha of each verb from that sentence and classify it into Kaarakas

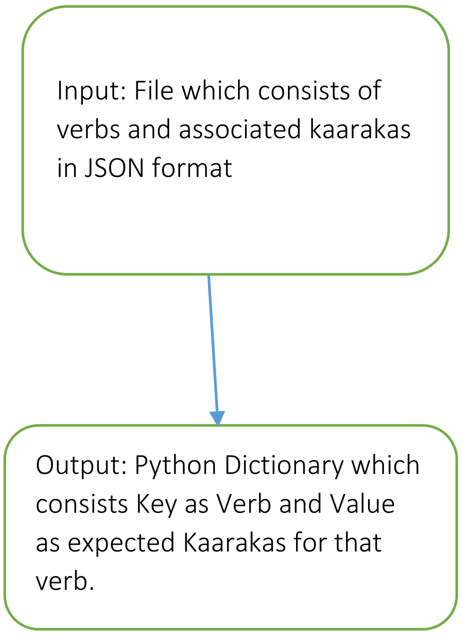

Input: File which consists of verbs and associated kaarakas in JSON format
Output: Python Dictionary which consists Key as Verb and Value as expected Kaarakas for that verb.

Fig 4.5 Read the Aakanksha from the file and create the dictionary for storing them

4.2.1.4 Set the corresponding score for each Naamapada against the matched Kaaraka based on rule.

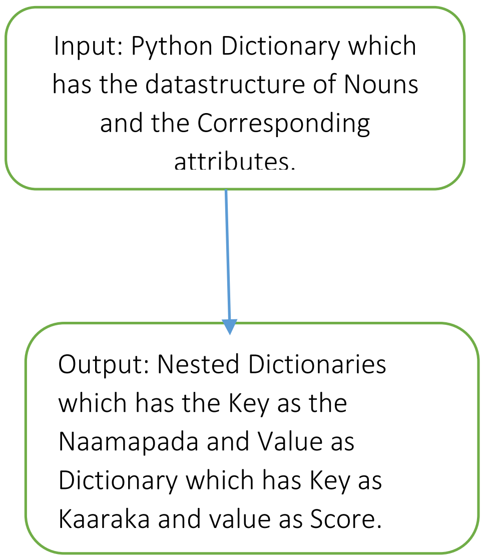

Input: Python Dictionary which has the datastructure of Nouns and the Corresponding attributes.
Output: Nested Dictionaries which has the Key as the Naamapada and Value as Dictionary which has Key as Kaaraka and value as Score.

Fig 4.6 Process each naamapada in the sentence and assign score based on the rule

4.2.1.5 Apply Constraint Propogation on the Naamapada and remove the Kaaraka which does not belong to the Set.

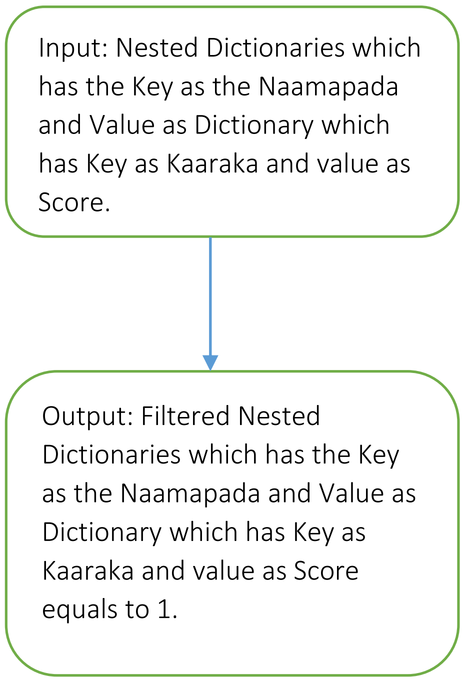

Input: Nested Dictionaries which has the Key as the Naamapada and Value as Dictionary which has Key as Kaaraka and value as Score.
Output: Filtered Nested Dictionaries which has the Key as the Naamapada and Value as Dictionary which has Key as Kaaraka and value as Score equals to 1.

Fig 4.7 Perform Constraint Propagation on the naamapada’s so that the non-matching Kaaraka is deleted from the Naamapada

4.3 CONSTRAINT:

This section describes what the project intends to do and what it does not in detail.

4.3.1 WHAT DOES THE PROJECT DO

Project **"Semantic Parser for Kannada"** is intended to perform the following tasks:

- A Program accepts a text file which consists of 100 Kannada Sentences with the Morphological tags.
- For classifying words as Nouns and Verbs, the tag set is really important for that word which is Output from Morphological Parser.
- Accepts Complex Kannada Sentences having Sentence with more than 1 verb (100 Kannada Sentences).
- The project intends to show the clear Classifications of Kaaraka for each Naamapada in the GUI form with the action (verb) placed at center and Kaaraka encircled with that action.
- The Mapping is only constrained from Vibhatkis to Kaarakas.

4.3.2 WHAT DOES THE PROJECT NOT DO

Project **"Semantic Parser for Kannada"** is intended not to perform the following tasks:

- The Program cannot take Random Input Sentences (Words which are not present in the dictionary) because the Morphological tags will not be generated.
- The Metaphoric language are not recognized.

CHAPTER - 5

SYSTEM REQUIREMENTS SPECIFICATION

The software and hardware requirements specify a complete set of prerequisites that are required to build the system. It describes the behaviour of the system to be developed. The document includes functional and non-functional requirements for the software to be developed. The functional requirements define what functions the software should perform and non-functional requirements specify the constraints on design and implementation and other performance and scalability related details. Requirements must be measurable, testable, and related to identify the needs or opportunities, and be defined to a level of detail sufficient for system design. The assumption and dependencies describe a set of constraints that determine the success of the project. Assumption lists various factors or system environmental states that has to be fulfilled for the success of the project.

5.1 HARDWARE REQUIREMENTS:

The hardware requirements for Semantic Parser for Kannada are:

- Minimum 512 MB of secondary memory space should be available for storing the files and processing them.
- For faster processing speed, a minimum of 2 GB RAM is needed.
- A 64 bit system is preferred for faster processing.
- System with Ubuntu OS 12.04 or higher version

5.2 SOFTWARE REQUIREMENTS:

The software requirements for this project are:

- Python 3
- Python packages needed are:
  - Python-Yaml
- Google Chrome or Mozilla Browser supporting HTML5 and CSS5.

5.3 NON-FUNCTIONAL REQUIREMENTS:

Non-functional Requirements specify the quality of the system, is mostly related to the satisfiability of the user. They cover all the remaining requirements which are not covered by the functional requirements. They specify criteria that judge the operation of a system, rather than specific behaviours.

5.3.1 DEPENDENCIES

5.3.1.1 Security and Privacy Dependencies:

- The user is expected to ensure that the input file is not corrupt and is responsible to maintain the integrity of the files.
- The application has access to the directories that contain the Files. Hence, the user has to ensure he/she does not specify a critical directory (such as root) and also not keep any other files other than that of the Kannada Sentences.
- The application requires permission to read and write files, and create directories. Hence, it is expected that the user permissions for the mentioned directories are appropriately specified.

5.3.1.2 Performance dependencies:

- Performance is not of concern, as processing 100 Sentences will take less time and writing the output into the file is not much time consuming.
- The GUI is quick and faster in terms of loading the HTML page.

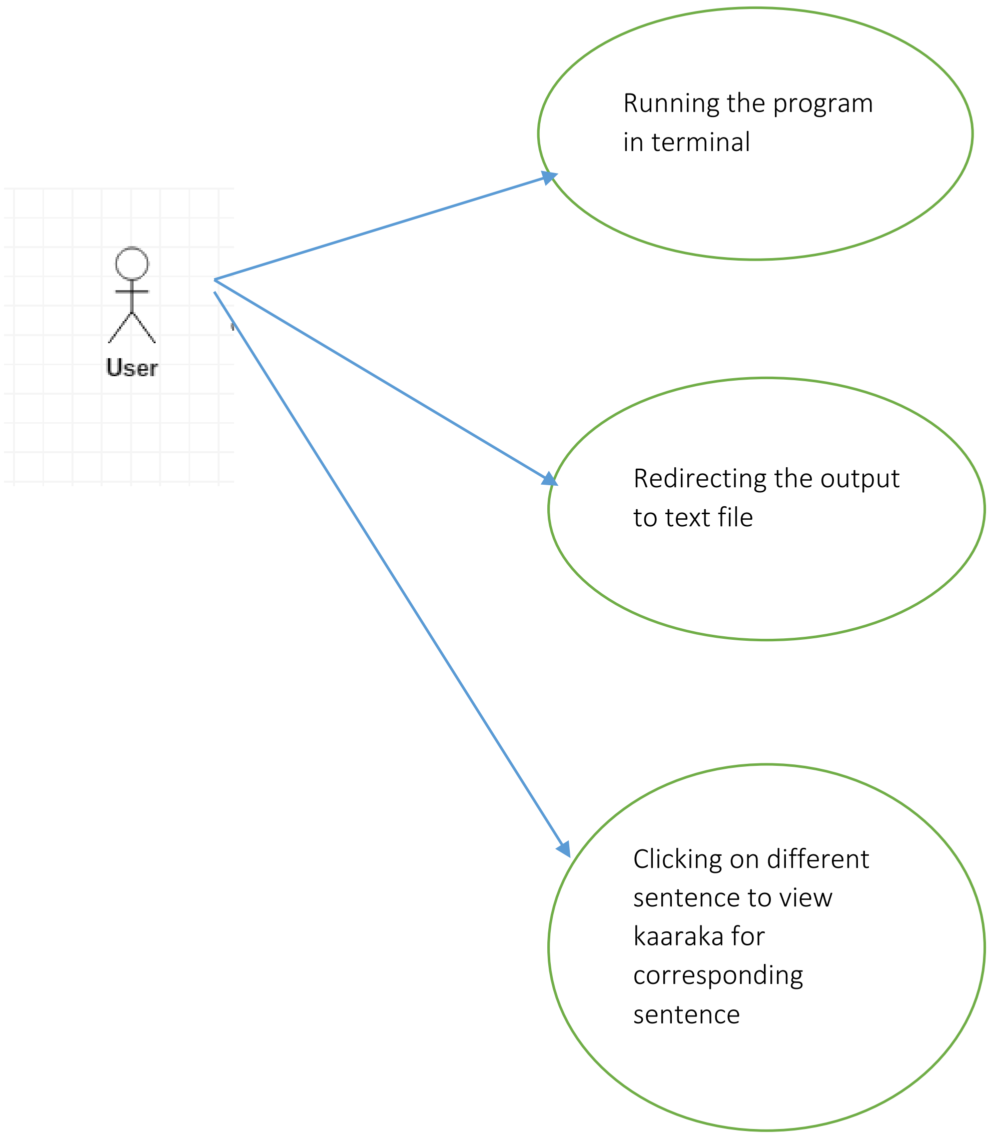
5.3.2 ASSUMPTIONS

The project assumes the following conditions to be true:

- The software dependencies have all been accurately installed without any errors or warnings.
- All Kannada Sentences with the Morphological tags are accurate and not corrupt.
- The hardware on which the program is running has power backup.
- The system has the ability to cache the model and decode it when necessary.

5.3.3 USER REQUIREMENTS

The user requirements is being specified by the use case diagram. Use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. So, to model the entire system numbers of use case diagrams are used. Use case diagrams are used to gather the requirements of a system including internal and external influences. The use case diagram for Semantic Parser for Kannada is shown below:



User
Running the program in terminal
Redirecting the output to text file
Clicking on different sentence to view kaaraka for corresponding sentence

Fig 5.1 Use case diagram for Semantic Parser

CHAPTER – 6

GANTT CHART

Task name	Start date	End date	Duration
Research on the domain	01/01/2016	14/01/2016	12
Selection and finalization of the topic	15/01/2016	20/01/2016	6
Presentation of the project topic	21/01/2016	21/01/2016	1
Literature Survey	21/01/2016	02/02/2016	10
Feasibility Analysis	02/02/2016	10/02/2016	9
System Requirements and Specification	11/02/2016	18/02/2016	7
Presentation and Architecture Designs	19/2/2016	03/03/2016	17
Detailed Design	03/03/2016	26/03/2016	24
Presentation and Test Planning	27/03/2016	04/04/2016	8
Test Plan Execution	04/04/2016	10/04/2016	8
Presentation and Report Writing	11/04/2016	27/04/2016	16
Report Approval and Submission	27/4/2016	05/05/2016	4

Table 6.1 GANTT Chart representing Project activities and the number of days spent on each activity

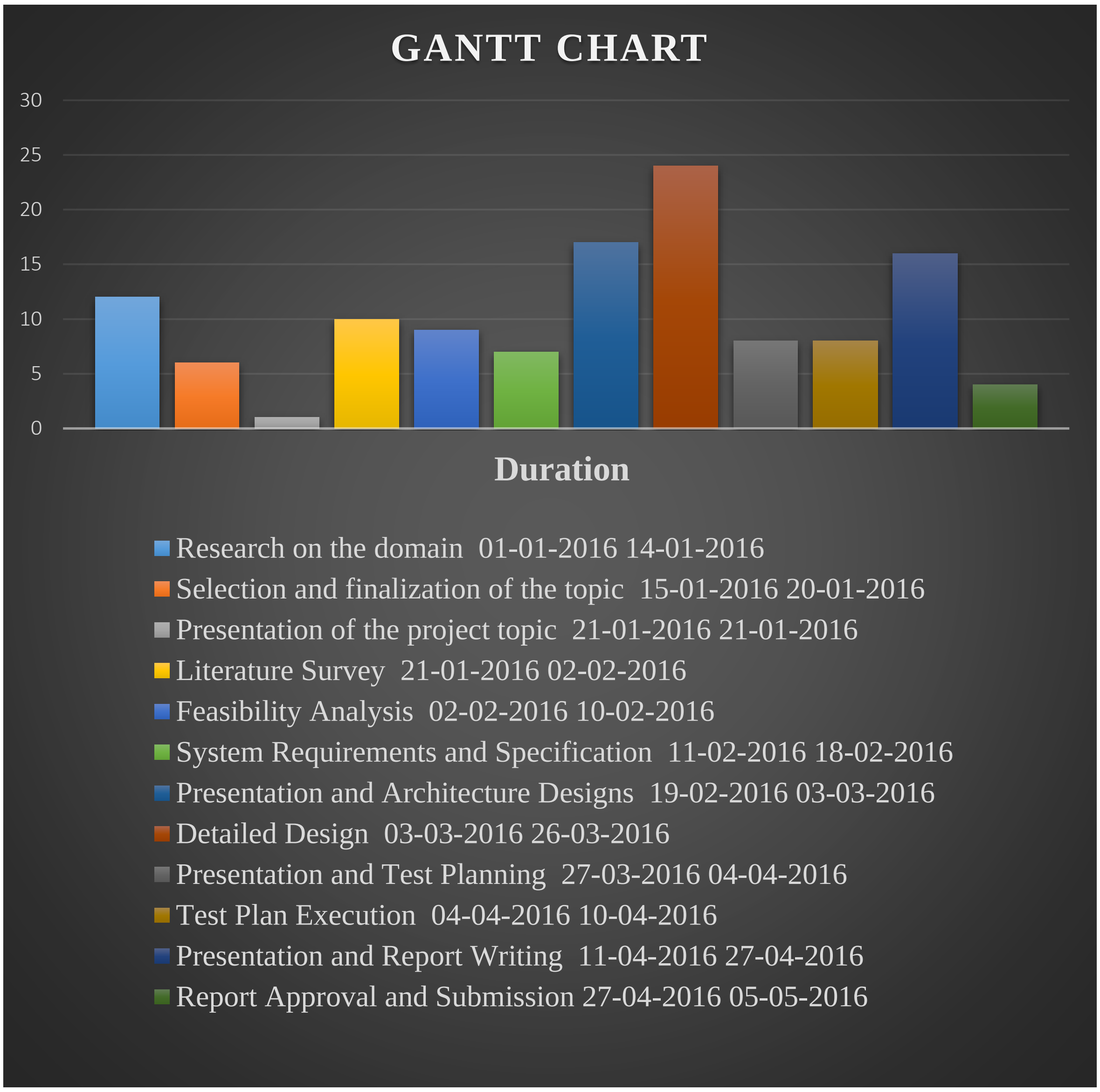


Fig 6.1 GANTT Chart representing activities carried out during JAN – MAY 2016

CHAPTER – 7
SYSTEM DESIGN

This section describes the high level design aspects of the project and also covers the low level description of the underlying components and their design structure. Topics covered include the following:

- System Overview
- System Architecture
- Data Design

7.1 BLOCK DIAGRAM

Below is the block diagram for the project "Semantic Parser for Kannada".

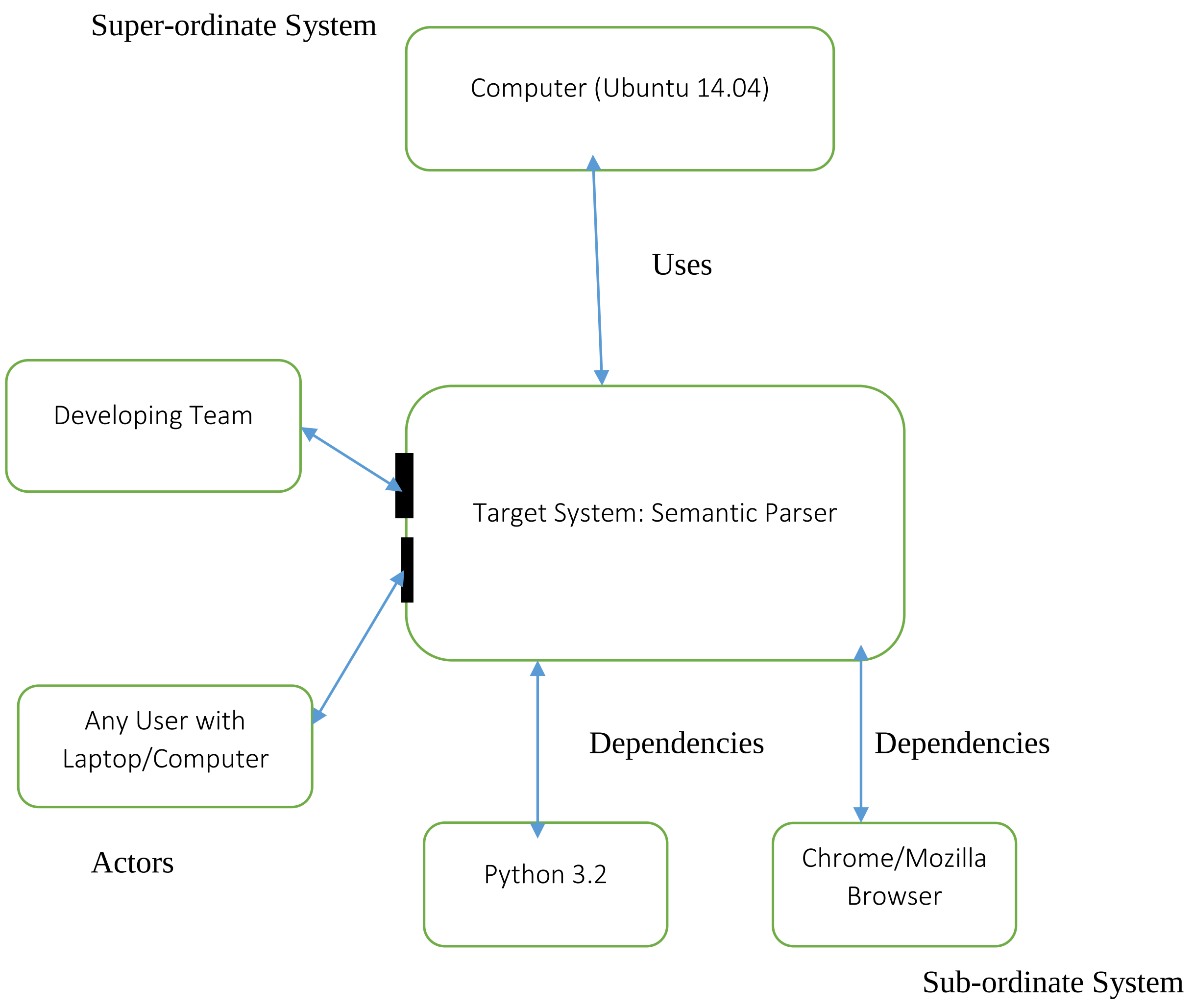


Fig 7.1 Diagram representing the overall architecture of the system

7.1.1 SYSTEM OVERVIEW

Keeping in mind the architectural design, the entire system is divided into four main components, namely:

1. Super-ordinate system
2. Sub-ordinate system
3. Target system
4. Actors

The *Target system* is the system that has the project source code and all the project requirements. The systems use the target system. The system that provides the necessary data processing support to the target system to complete the functionality is called the *Sub-ordinate system*, which is this case all the software and hardware support. *Actors* are the potential users of the system.

7.2 IMPLEMENTED MODULES

MODULE-1:

Classification of Nouns and Verbs from the given Input Sentence

Description:

This module Processes Each Sentence from an Array (List in Python) one at time, Identify the Nouns and Verbs from the tag-set.

Tags are short labels attached to words to indicate their lexical, morphological, syntactic, semantic or other grammatical properties. A tag-set is a set of tags designed for a particular language or class of languages with a particular purpose in mind. In the saara system, the tag-set is designed for the purposes of syntactic parsing, glossing/translation, language teaching/learning etc. The saara system uses a hierarchical design for tags. This makes the tags more flexible and convenient, while at the same time providing detailed, fine-grained information at various levels. The Kannada example below shows a plain sentence and its tagged form as generated by the saara system.

raamanu manege hoodanu

raamanu||raama||N-PRP-PER-M.SL-NOM manege||mane||N-COM-COU-N.SL-DAT

hoodanu||hoogu||V-IN-ABS-PAST-P3.M.SL

Here, for each word form, its root form and the tag are shown, separated by two vertical bars. The tag for the first word indicates that the first word is a noun, it is a proper noun, it indicates a (singular) person in masculine gender, and in nominative case. The second tag should be read as noun-common-countable-neuter_gender-singular-dative_case. The third tag should be read as verb-intransitive(akarmaka)-absolute_aspect-third_person-masculine-singular in agreement. Thus tagged sentences contain a lot of useful information for carrying out various NLP tasks.

MODULE – 2: ASSIGNING THE SCORE BASED ON THE SEMANTIC RULE

Based on the rules, the nature of the Tag-set decides on the Kaaraka.

Eg:

NOM – Nominative: Karta

ACC – Accusative – Karma

INST – Instrument – Karana

LOC – Location – Adhikarana

DAT – Dative – Sampradhana

GEN – Genitive – ABL – Ablative - Aapadhana

Depending on the tag, the score is assigned to each of the Naamapada.

7.3 DATA DESIGN

The data design of the project can be depicted using a data flow diagram. A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel. Below shown diagram depicts the data flow diagram for Semantic Parser for Kannada.

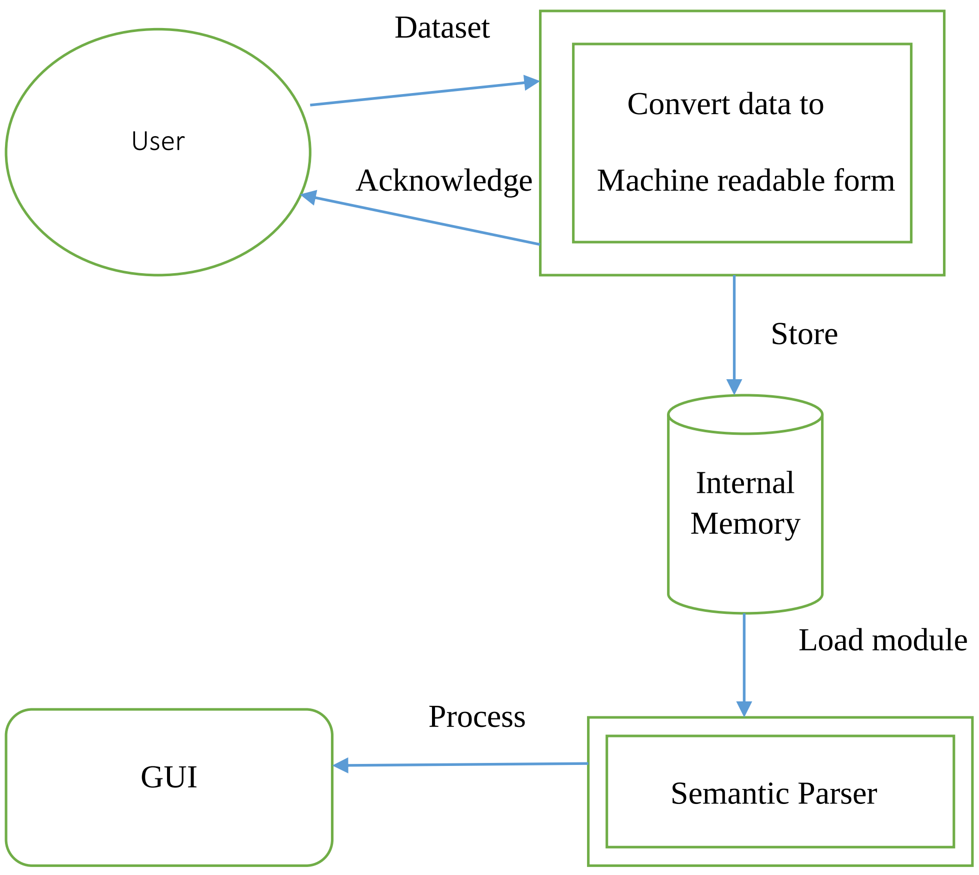


Fig 7.2 Diagram representing the flow of data between different entities

CHAPTER - 8

DETAILED DESIGN

This section of detailed design consists of low level data flow diagrams. Each module is further split into its low granular modules and data processing and data processing at each level is described in the below sections.

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel

A detailed data flow diagram consists of three sections:

• **Input**: The input to the module.

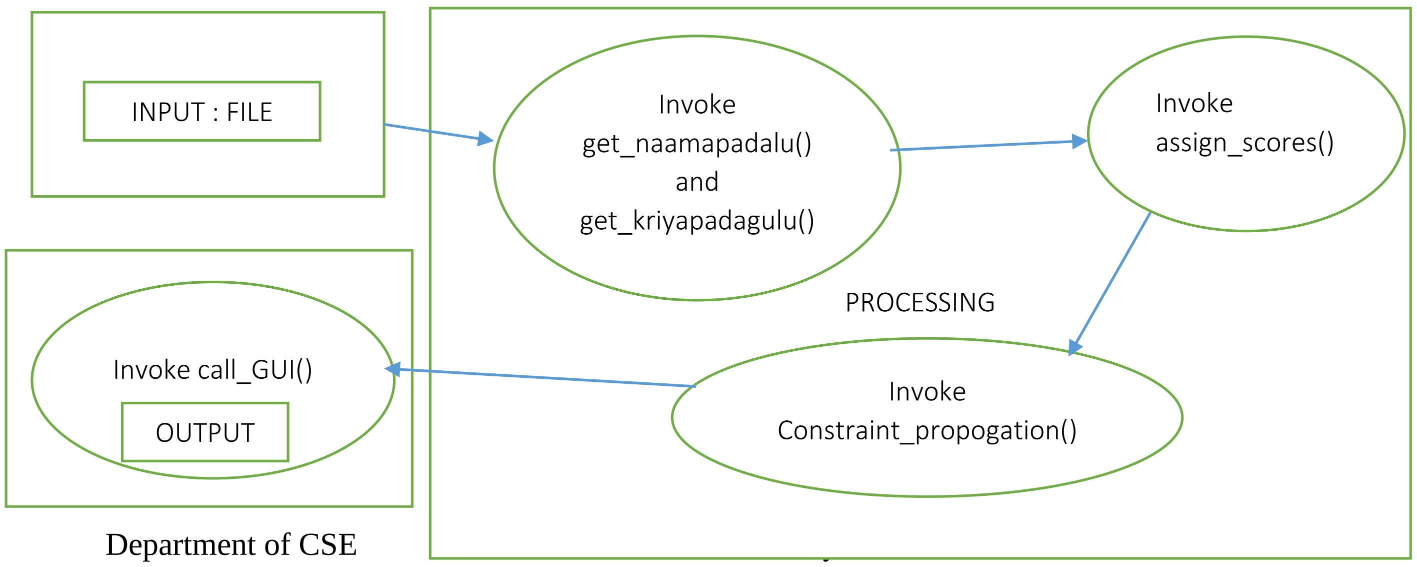
• **Control Transform**: The Method/API call that initiates the core functionality call.

• **Options**: The flow of all the core functionality calls.

• **Output**: Output of the module.

8.1 DATA FLOW DIAGRAM – LEVEL-1

Fig 8.1 Diagram representing the Data flow between different modules


INPUT : FILE
Invoke get_naamapadalu() and get_kriyapadagulu()
Invoke assign_scores()
PROCESSING
Invoke Constraint_propogation()
Invoke call_GUI()
OUTPUT

CHAPTER - 9
IMPLEMENTATION

File Name: Class_Kaaraka.py:

Description:

This program is written in Python Language.

Steps are as follows:

- The program initially tries to read the text file of 100 Kannada input sentences along with tag set which is processed one by one. The Sentences are then converted into a form of Python List which is represented in [ ] format (Array).
- The tag set is split based on the Noun and the Verb and the words are classified based on Verbs and Nouns in the sentence which is then stored in the form of Python Dictionary (Key as Noun and value as tag-set), same with the Verb.
- Then the file which consisted of verb and the expected Kaaraka for that verb is read as Input to the Aakanksha() function which in-turn will save the results in the form of Python Dictionary(Key as Verb and value as Expected Kaarakas).
- The datastructure is created for each naamapada and kriyapaada which has the following attributes:
    - Naamapada : { Type: PER, Count: SL, Gender: M, ANIM: PROH}
- Then Each naampada data structure is being processed and the Scores are assigned to each of the Naamapada based on the rule based Algorithm which will assign score of “1” if its best match and “0.5” if it’s a Partial match in the form of nested dictionary(Key as Naamapada and Value as Dictionary which in turn Key as Kaaraka and value as Score).
- Then this Python dictionary is then processed through Constraint Propagation function which will eliminate the non-matching Kaaraka from the list.
- The backtrack function will add the Kaaraka to naamapada (if not added by constraint propogation), if the verb expectation matches the kaaraka list.
- Then the final kaaraka list is being processed to display the result in GUI.

Pseudocode:

Step 1: Read the input files – Sentences, Sentences with Morphological tags.

Step 2: Find all the nouns and verbs in the sentences.

Step 3: Read the aakanksha of the verbs from the yaml file (Manual classification)

Step 4: Create a datastructure for all Naamapada which has the following format:

Naamapada_datastructure = { Naamapada : { GEN : M, Type: PER, Person: P1, Count: SL, Anim: Proh } for all Naamapada in the sentence.

```
Step 5: process_naamapadagalu() //assign scores for each kaaraka it can take
{
  for all kriyaapada in kriyaapada_list: // This list has all the kriyapaada and associated tags in it.
      kriyaapada_dict = self.kriyaapada_data[kriyaapada] //kriyapaada dict has the attributes of the Gender, Count, Person, Type etc for all Kriyapaada's
      yogyatha[kriyaapada] = {} // Dictionary which has all the yogyatha of naamapada's.
for naamapada , naamapada_dict in kriyaapada_naamapad_assoc(): //Dictionary which has verb and naamapada pair along with naampada_dictionary which has attirbutes
     if naamapada_dict['type'] == 'CARD': // CARD represents Cardinal(Count) of the naamapada (person, thing etc)
              continue // Do not iterate if you encounter this condition
     self.yogyatha[kriyaapada][naamapada] = {}
      if naamapada_dict['case'] == "NOM": //Check if the Naamapada is in Prathama Vibhakti
          if "PER" in naamapada_dict['type'] or "ANIM" in naamapada_dict['type']: // If the Naamapada has the attribute person or ANIM in it, then iterate this.
```

```
                    if (kriyaapada_dict['count'] ): // Check if the Naamapada Count is Singular or Plural
                        if (kriyaapada_dict['gender']): // Check if the Naamapada is Male or Frmale or Neutral
                            if naamapada_dict['gender'] in kriyaapada_dict['gender'] and naamapada_dict['count'] == kriyaapada_dict['count']: // Check if these attributes match both in kriyapada and naamapada
                                self.yogyatha[kriyaapada][naamapada]['kartha'] = 1 // Confirm the score of the naamapada as Kartha to 1
                            else:
                                self.yogyatha[kriyaapada][naamapada]['karma'] = 0.5 // If the attribute didn't match, assign the score karma to 0.5


                    else
                            if naamapada_dict['count'] == kriyaapada_dict['count']: // If the count is same in both Naamapada and Kriyapada, then assign kartha to 0.75
                                self.yogyatha[kriyaapada][naamapada]['kartha'] = 0.75
                            else: // if the condition of count didn't match, then assign 05 to karma
                                self.yogyatha[kriyaapada][naamapada]['karma'] = 0.5
                else:
                    self.yogyatha[kriyaapada][naamapada]['karma'] = 0.5
                    self.yogyatha[kriyaapada][naamapada]['kartha'] = 0.75
elif "LOC" in naamapada_dict['type']: // If the LOC in the tag of Naamapada, then fix the adhikaraNa to 1
    self.yogyatha[kriyaapada][naamapada]['adhikaraNa'] = 1
```

```
else:
        if 'F' in naamapada_dict['gender'] or 'M' in naamapada_dict['gender']: // if Female in tag
of naamapada then assign kartha to 0.5
              if naamapada_dict['count'] == kriyaapada_dict['count']:
                    self.yogyatha[kriyaapada][naamapada]['kartha'] = 0.5
                    self.yogyatha[kriyaapada][naamapada]['karma'] = 0.25
            else: // if the Female not in the tag, then assign kartha and karma to 0.25
                    self.yogyatha[kriyaapada][naamapada]['kartha'] = 0.25
                    self.yogyatha[kriyaapada][naamapada]['karma'] = 0.25
elif 'N' in naamapada_dict['gender']: // if the Neutral in the tag of naamapada, then assign
0.25 to kartha
        self.yogyatha[kriyaapada][naamapada]['kartha'] = 0.25
       self.yogyatha[kriyaapada][naamapada]['karma'] = 0.5
elif naamapada_dict['case'] == "ACC": //If  the case is Accusative then assign Karma to 1
       self.yogyatha[kriyaapada][naamapada]["karma"] = 1
elif naamapada_dict['case'] == "ABL": // If case is Ablative, then assign Karana score to 1
      if naamapada_dict['type'] == "BODY": // if the tag has BODY as attribute in it, then
fix it to Karana by assigning score to "1"
            self.yogyatha[kriyaapada][naamapada]["karaNa"] = 1
            self.yogyatha[kriyaapada][naamapada]["apaadaana"] = 0.5
            self.yogyatha[kriyaapada][naamapada]["karaNa"] = 0.5
elif naamapada_dict['case'] == "DAT": // if the case is Dative
           if "PER" in naamapada_dict['type']: // if tag has PERSON as attribute type
                if "sampradana" not in self.aakanksha[kriyaapada]: // if sampradhana not a part
of Aakanksha
```

```
                self.yogyatha[kriyaapada][naamapada]["karma"] = 1 //Then the kaaraka is karma
          else:
                self.yogyatha[kriyaapada][naamapada]["sampradana"] = 1 // Fix it to sampradhana
elif "LOC" in naamapada_dict['type']: // LOCATION In the tag of naamapada as case
    if 'PLA' in naamapada_dict['type']: // if PLACE is part of LOCATION
        self.yogyatha[kriyaapada][naamapada]["destination"] = 1 //then fix the kaaraka as destination
    elif 'TIM' in naamapada_dict['type']: // if the TIME is part of attribute type of Naamapada
        self.yogyatha[kriyaapada][naamapada]["adhikaraNa"] = 1 //then fix the kaaraka – adhikaraNa for naamapada by assigning score as 1
    else: // Partially assign 0.5 score to adhikaraNa and destination
        self.yogyatha[kriyaapada][naamapada]["adhikaraNa"] = 0.5
        self.yogyatha[kriyaapada][naamapada]["destination"] = 0.5
    else:
        self.yogyatha[kriyaapada][naamapada]["destination"] = 0.5
elif naamapada_dict['case'] == "LOC" //if LOCATION in case attribute of naamapada
      if naamapada_dict['type'] == "BODY": //if BODY is part of attribute type of Naamapada
          self.yogyatha[kriyaapada][naamapada]["karaNa"] = 1
     else:
          naamapada_dict['gender'] == 'M' or naamapada_dict['gender'] == 'F':
          self.yogyatha[kriyaapada][naamapada]["karma"] = 1
```

```
else:
        self.yogyatha[kriyaapada][naamapada]["adhikaraNa"] = 1
}
```

Step 6: constraint_propagation() // **Perform the Constraint propagation on the kaaraka list by eliminating non-matching kaaraka from the list**

```
{
        for each kaaraka in aakanksha:
                Find its max value and assign the corresponding noun to that kaaraka and
                eliminate that kaaraka's possibility from other nouns. If still nouns are present
                and if it fits for any other kaarakas other than aakanksha assign them
                correspondingly
}
```

Step 7: Kaaraka Backtrack()

```
{
      for eack kaaraka in final_kaaraka:
         if the kaaraka is present in aakanksha of kriyapada, then backtrack and add the
      kaaraka accordingly to the naamapada
}
```

CHAPTER - 10

TESTING

The purpose of this chapter is to enable to project management to know the testing status of the project and the quality level of the application. It provides an insight on the various test cases and the scope of activity that has led to the development of this document. Testing process was divided into unit testing and integration testing. Unit testing phase was used to perform testing on individual units of the application. After this, the entire application was tested to verify that the individual units of the application were in precision when run together.

The unit testing involved the checking of each unit namely the frontend GUI, the navigation, content of the webpages. In the testing of the control logic, compilation, exception handling and other errors were checked for. The file system interface was tested to verify the read-write process. The integration testing was performed once the unit testing was completed. The application was tested for various Kannada Sentences. All types of flows were tested throughout.

10.1 ASSUMPTIONS

The following assumptions have been made for the testing process:

- File system has sufficient storage available.
- Python Package is installed on working fine.
- The CPU usage available is very high (In terms of percentage ~ 90%).

10.2 CONSTRAINTS

- Secondary data storage should be quiet enough to hold all the data and test data.
- The program should process file also as there is no restriction specified on size of the file to be processed.
- Python program will be running all the time. It should never automatically stop or be stopped by the user manually.
- CPU usage availability must be high.

10.3 RISKS

The following are the possible risks that could occur in the project development phase that may affect the quality of the project:

• **Schedule Risks**: Project schedule gets slipped when project tasks and schedule release risks are not addressed properly. They can be due to wrong time estimation, resources not being tracked properly, failure to identify complex functionalities and the time required to implement these.

• **Operational Risks**: The type of laws that occur due to improper process implementation or failed system come under Operational Risks category. This can be caused due to failure to resolve responsibilities, no resource planning or no communication in the team.

• **Technical Risks**: Technical risks lead to failure of functionality and performance of the software. This may be due to complexity of the project implementation.

10.4 TEST SUMMARY ASSESSMENT

Sl.No	Test Case	Expected Result	Actual Result	Outcome
1	ಬಾ	Verb : ಬಾ||ಬಾ kartha ['ನೀನು||ನೀನು']	Verb : ಬಾ||ಬಾ kartha ['ನೀನು||ನೀನು']	Success
2	ಬನ್ನಿ	Verb : ಬನ್ನಿ||ಬನ್ನಿ kartha ['ನೀವು||ನೀವು']	Verb : ಬನ್ನಿ||ಬನ್ನಿ kartha ['ನೀವು||ನೀವು']	Success
3	ತಿನ್ನು	Verb : ತಿನ್ನು||ತಿನ್ನು kartha ['ನೀನು||ನೀನು']	Verb : ತಿನ್ನು||ತಿನ್ನು kartha ['ನೀನು||ನೀನು']	Success
4	ನೀನು ಬಾ	Verb : ಬಾ||ಬಾ kartha ['ನೀನು||ನೀನು']	Verb : ಬಾ||ಬಾ kartha ['ನೀನು||ನೀನು']	Success
5	ಅವನು ಬಂದನು	Verb : ಬಂದನು||ಬರು kartha ['ಅವನು||ಅವನು']	Verb : ಬಂದನು||ಬರು kartha ['ಅವನು||ಅವನು']	Success
6	ಅವಳು ಮನೆಗೆ ಹೋಗುವಳು	Verb : ಹೋಗುವಳು||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಅವಳು||ಅವಳು']	Verb : ಹೋಗುವಳು||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಅವಳು||ಅವಳು']	Success

7	ರಾಮನು ಹಣ್ಣನ್ನು ತಿಂದನು	Verb : ತಿಂದನು||ತಿನ್ನು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಹಣ್ಣನ್ನು||ಹಣ್ಣು']	Verb : ತಿಂದನು||ತಿನ್ನು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಹಣ್ಣನ್ನು||ಹಣ್ಣು']	Success
8	ನನಗೆ ನಿದ್ದೆ ಬರುತ್ತಿದೆ	Verb : ಬರುತ್ತಿದೆ||ಬರು kartha ['ನಿದ್ದೆ||ನಿದ್ದೆ'] karma ['ನನಗೆ||ನನಗೆ']	Verb : ಬರುತ್ತಿದೆ||ಬರು kartha ['ನಿದ್ದೆ||ನಿದ್ದೆ'] karma ['ನನಗೆ||ನನಗೆ']	Success
9	ಅವನು ನನಗೆ ದುಡ್ಡು ಕೊಡಬೇಕು	Verb : ಕೊಡಬೇಕು||ಕೊಡು kartha ['ಅವನು||ಅವನು'] karma ['ದುಡ್ಡು||ದುಡ್ಡು'] sampradana ['ನನಗೆ||ನನಗೆ']	Verb : ಕೊಡಬೇಕು||ಕೊಡು kartha ['ಅವನು||ಅವನು'] karma ['ದುಡ್ಡು||ದುಡ್ಡು'] sampradana ['ನನಗೆ||ನನಗೆ']	Success
10	ದೇವರು ಒಳ್ಳೆಯದನ್ನು ಮಾಡಲಿ	Verb : ಮಾಡಲಿ||ಮಾಡು kartha ['ದೇವರು||ದೇವರು'] karma ['ಒಳ್ಳೆಯದನ್ನು||ಒಳ್ಳೆಯದು']	Verb : ಮಾಡಲಿ||ಮಾಡು kartha ['ದೇವರು||ದೇವರು'] karma ['ಒಳ್ಳೆಯದನ್ನು||ಒಳ್ಳೆಯದು']	Success
11	ಅಕ್ಕ ಎಲ್ಲಿಗೆ ಹೋಗಿದ್ದಾಳೆ?	Verb : ಹೋಗಿದ್ದಾಳೆ||ಹೋಗು destination ['ಎಲ್ಲಿಗೆ||ಎಲ್ಲಿ'] kartha ['ಅಕ್ಕ||ಅಕ್ಕ']	Verb : ಹೋಗಿದ್ದಾಳೆ||ಹೋಗು destination ['ಎಲ್ಲಿಗೆ||ಎಲ್ಲಿ'] kartha ['ಅಕ್ಕ||ಅಕ್ಕ']	Success
12	ಅಮ್ಮ ಯಾರಿಗೆ ಮೊದಲು ದೋಸೆ ಕೊಡುತ್ತಾಳೆ?	Verb : ಕೊಡುತ್ತಾಳೆ||ಕೊಡು kartha ['ಅಮ್ಮ||ಅಮ್ಮ'] karma ['ದೋಸೆ||ದೋಸೆ'] sampradana ['ಯಾರಿಗೆ||ಯಾರು'] adhikaraNa ['ಮೊದಲು||ಮೊದಲು']	Verb : ಕೊಡುತ್ತಾಳೆ||ಕೊಡು kartha ['ಅಮ್ಮ||ಅಮ್ಮ'] karma ['ದೋಸೆ||ದೋಸೆ'] sampradana ['ಯಾರಿಗೆ||ಯಾರು'] adhikaraNa ['ಮೊದಲು||ಮೊದಲು']	Success
13	ಚಿಕ್ಕಮ್ಮ ಹೇಗೆ ಅಡುಗೆ ಮಾಡಬಲ್ಲಳು?	Verb : ಮಾಡಬಲ್ಲಳು||ಮಾಡು kartha ['ಚಿಕ್ಕಮ್ಮ||ಚಿಕ್ಕಮ್ಮ'] karma ['ಅಡುಗೆ||ಅಡುಗೆ']	Verb : ಮಾಡಬಲ್ಲಳು||ಮಾಡು kartha ['ಚಿಕ್ಕಮ್ಮ||ಚಿಕ್ಕಮ್ಮ'] karma ['ಅಡುಗೆ||ಅಡುಗೆ']	Success

14	ಅಡುಗೆಮನೆಯಲ್ಲಿ ಏನೂ ಇಲ್ಲವಂತೆ	Verb : ಇಲ್ಲವಂತೆ||ಇಲ್ಲವಂತೆ karma ['ಏನೂ||ಏನೂ'] adhikaraNa ['ಅಡುಗೆಮನೆಯಲ್ಲಿ||ಅಡುಗೆಮನೆ	Verb : ಇಲ್ಲವಂತೆ||ಇಲ್ಲವಂತೆ karma ['ಏನೂ||ಏನೂ'] adhikaraNa ['ಅಡುಗೆಮನೆಯಲ್ಲಿ||ಅಡುಗೆಮನೆ	Success
15	ಮರದಿಂದ ಹಣ್ಣು ಕೆಳಕ್ಕೆ ಬಿದ್ದಿತು	Verb : ಬಿದ್ದಿತು||ಬೀಳು destination ['ಕೆಳಕ್ಕೆ||ಕೆಳಕ್ಕೆ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] apaadaana ['ಮರದಿಂದ||ಮರ']	Verb : ಬಿದ್ದಿತು||ಬೀಳು destination ['ಕೆಳಕ್ಕೆ||ಕೆಳಕ್ಕೆ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] Success apaadaana ['ಮರದಿಂದ||ಮರ']	Success
16	ಯಾರೂ ಮನೆಯಿಂದ ಹೊರಗೆ ಹೋಗಲೇಬಾರದು	Verb : ಹೋಗಲೇಬಾರದು||ಹೋಗು kartha ['ಯಾರೂ||ಯಾರು'] apaadaana ['ಮನೆಯಿಂದ||ಮನೆ'] adhikaraNa ['ಹೊರಗೆ||ಹೊರಗೆ']	Verb : ಹೋಗಲೇಬಾರದು||ಹೋಗು kartha ['ಯಾರೂ||ಯಾರು'] apaadaana ['ಮನೆಯಿಂದ||ಮನೆ'] adhikaraNa ['ಹೊರಗೆ||ಹೊರಗೆ']	Success
17	ಕೆಲವರು ಮನೆಯಿಂದ ಮನೆಗೆ ಅಲೆಯುತ್ತಾರೆ	Verb : ಅಲೆಯುತ್ತಾರೆ||ಅಲೆ destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಕೆಲವರು||ಕೆಲವರು'] apaadaana ['ಮನೆಯಿಂದ||ಮನೆ']	Verb : ಅಲೆಯುತ್ತಾರೆ||ಅಲೆ destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಕೆಲವರು||ಕೆಲವರು'] apaadaana ['ಮನೆಯಿಂದ||ಮನೆ']	Success
18	ನಿಮಗೆ ಏನು ಬೇಕು	Verb : ಬೇಕು||ಬೇಕು karma ['ಏನು||ಏನು'] sampradana ['ನಿಮಗೆ||ನಿಮಗೆ']	Verb : ಬೇಕು||ಬೇಕು karma ['ಏನು||ಏನು'] sampradana ['ನಿಮಗೆ||ನಿಮಗೆ']	Success
19	ನಾಯಿಗಳು ಕಾಲಿನಿಂದ ನೆಲವನ್ನು ಕೆರೆಯುತ್ತವೆ	Verb : ಕೆರೆಯುತ್ತವೆ||ಕೆರೆ kartha ['ನಾಯಿಗಳು||ನಾಯಿ'] karma ['ನೆಲವನ್ನು||ನೆಲ'] karaNa ['ಕಾಲಿನಿಂದ||ಕಾಲು']	Verb : ಕೆರೆಯುತ್ತವೆ||ಕೆರೆ kartha ['ನಾಯಿಗಳು||ನಾಯಿ'] karma ['ನೆಲವನ್ನು||ನೆಲ'] karaNa ['ಕಾಲಿನಿಂದ||ಕಾಲು']	Success
20	ಪೂರ್ವದ ಕಡೆ ನೋಡು	Verb : ನೋಡು||ನೋಡು kartha ['ನೀನು||ನೀನು']	Verb : ನೋಡು||ನೋಡು kartha ['ನೀನು||ನೀನು']	Success

		adhikaraNa ['ಕಡೆ||ಕಡೆ']	adhikaraNa ['ಕಡೆ||ಕಡೆ']	
21	ಸೂರ್ಯ ನೆತ್ತಿಯ ಮೇಲೆ ಬಂದಿದ್ದಾನೆ	Verb : ಬಂದಿದ್ದಾನೆ||ಬರು kartha ['ಸೂರ್ಯ||ಸೂರ್ಯ'] adhikaraNa ['ಮೇಲೆ||ಮೇಲೆ']	Verb : ಬಂದಿದ್ದಾನೆ||ಬರು kartha ['ಸೂರ್ಯ||ಸೂರ್ಯ'] adhikaraNa ['ಮೇಲೆ||ಮೇಲೆ']	Success
22	ರಾಮನು ಸೀತೆಯ ಜೊತೆ ಕಾಡಿಗೆ ನಡೆದನು	Verb : ನಡೆದನು||ನಡೆ destination ['ಕಾಡಿಗೆ||ಕಾಡಿಗೆ'] kartha ['ರಾಮನು||ರಾಮ']	Verb : ನಡೆದನು||ನಡೆ destination ['ಕಾಡಿಗೆ||ಕಾಡಿಗೆ'] kartha ['ರಾಮನು||ರಾಮ']	Success
23	ಹುಡುಗರು ಬಾವಿಯಲ್ಲಿ ಇಣಕಿದರು	Verb : ಇಣಕಿದರು||ಇಣಕು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] adhikaraNa ['ಬಾವಿಯಲ್ಲಿ||ಬಾವಿ']	Verb : ಇಣಕಿದರು||ಇಣಕು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] adhikaraNa ['ಬಾವಿಯಲ್ಲಿ||ಬಾವಿ']	Success
24	ಸಂಜೆ ಹೋಗೋಣ	Verb : ಹೋಗೋಣ||ಹೋಗು kartha ['ನಾವು||ನಾವು'] adhikaraNa ['ಸಂಜೆ||ಸಂಜೆ']	Verb : ಹೋಗೋಣ||ಹೋಗು kartha ['ನಾವು||ನಾವು'] adhikaraNa ['ಸಂಜೆ||ಸಂಜೆ']	Success
25	ನಾನು ನಾಳೆ ನಿಮ್ಮ ಮನೆಗೆ ಬರುವುದಿಲ್ಲ	Verb : ಬರುವುದಿಲ್ಲ||ಬರು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ನಾನು||ನಾನು'] adhikaraNa ['ನಾಳೆ||ನಾಳೆ']	Verb : ಬರುವುದಿಲ್ಲ||ಬರು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ನಾನು||ನಾನು'] adhikaraNa ['ನಾಳೆ||ನಾಳೆ']	Success
26	ಹೇಳಲಾರೆ	Verb : ಹೇಳಲಾರೆ||ಹೇಳು kartha ['ನಾನು||ನಾನು or ನೀನು||ನೀನು’]	Verb : ಹೇಳಲಾರೆ||ಹೇಳು kartha ['ನಾನು||ನಾನು or ನೀನು||ನೀನು	Success
27	ನನಗೆ ಜ್ವರ ಬಂದಿದೆ	Verb : ಬಂದಿದೆ||ಬರು kartha ['ಜ್ವರ||ಜ್ವರ'] karma ['ನನಗೆ||ನನಗೆ']	Verb : ಬಂದಿದೆ||ಬರು kartha ['ಜ್ವರ||ಜ್ವರ'] karma ['ನನಗೆ||ನನಗೆ']	Success
28	ನನಗೆ ಬೆಲ್ಲ ಬೇಕು	Verb : ಬೇಕು||ಬೇಕು karma ['ಬೆಲ್ಲ||ಬೆಲ್ಲ'] sampradana ['ನನಗೆ||ನನಗೆ']	Verb : ಬೇಕು||ಬೇಕು karma ['ಬೆಲ್ಲ||ಬೆಲ್ಲ'] sampradana ['ನನಗೆ||ನನಗೆ']	Success

29	ಹುಡುಗರು ಸೀಬೆ ಹಣ್ಣುಗಳನ್ನು ತಿಂದರ	Verb : ತಿಂದರು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣುಗಳನ್ನು||ಹಣ್ಣು']	Verb : ತಿಂದರು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣುಗಳನ್ನು||ಹಣ್ಣು']	Success
30	ಹುಡುಗರು ಹಣ್ಣು ತಿಂದು ಮನೆಗೆ ಹೋದರು	Verb : ತಿಂದು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] Verb : ಹೋದರು||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Verb : ತಿಂದು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] Verb : ಹೋದರು||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Success
31	ಹುಡುಗರು ಹಣ್ಣು ತಿಂದು ಮನೆಗೆ ಹೋಗಿ ಮುಖ ತೊಳೆದುಕೊಂಡು ಆಟಕ್ಕೆ ಓಡಿದರು	Verb : ತಿಂದು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] Verb : ಹೋಗಿ||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ'] Verb : ತೊಳೆದುಕೊಂಡು||ತೊಳೆ kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಮುಖ||ಮುಖ'] Verb : ಓಡಿದರು||ಓಡು destination ['ಆಟಕ್ಕೆ||ಆಟ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Verb : ತಿಂದು||ತಿನ್ನು kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಹಣ್ಣು||ಹಣ್ಣು'] Verb : ಹೋಗಿ||ಹೋಗು destination ['ಮನೆಗೆ||ಮನೆ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ'] Verb : ತೊಳೆದುಕೊಂಡು||ತೊಳೆ kartha ['ಹುಡುಗರು||ಹುಡುಗ'] karma ['ಮುಖ||ಮುಖ'] Verb : ಓಡಿದರು||ಓಡು destination ['ಆಟಕ್ಕೆ||ಆಟ'] kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Success
32	ಹುಡುಗರು ಹಾಲು ಮತ್ತು ಹಣ್ಣನ್ನು ಸೇವಿಸಿದರು	Verb : ಸೇವಿಸಿದರು||ಸೇವಿಸು kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Verb : ಸೇವಿಸಿದರು||ಸೇವಿಸು kartha ['ಹುಡುಗರು||ಹುಡುಗ']	Success

		karma ['ಹಣ್ಣನ್ನು||ಹಣ್ಣು', 'ಹಾಲು||ಹಾಲು']	karma ['ಹಣ್ಣನ್ನು||ಹಣ್ಣು', 'ಹಾಲು||ಹಾಲು']	
33	ಆ ದಿನ ರಾಮನು ಬೇಗ ಎದ್ದು ಶೌಚ ಸ್ನಾನ ಇತ್ಯಾದಿಗಳನ್ನು ಮುಗಿಸಿಕೊಂಡು ದೇವರ ಪೂಜೆ ಮಾಡಿ ಕೈಯಲ್ಲಿ ಚೀಲ ಹಿಡಿದು ಸಾಮಾನು ತರಲು ಪೇಟೆಗೆ ಹೊರಟನು	Verb : ಎದ್ದು||ಏಳು kartha ['ರಾಮನು||ರಾಮ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಮುಗಿಸಿಕೊಂಡು||ಮುಗಿಸು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಇತ್ಯಾದಿಗಳನ್ನು||ಇತ್ಯಾದಿ', 'ಸ್ನಾನ||ಸ್ನಾನ', 'ಶೌಚ||ಶೌಚ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಮಾಡಿ||ಮಾಡು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಪೂಜೆ||ಪೂಜೆ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಹಿಡಿದು||ಹಿಡಿ kartha ['ರಾಮನು||ರಾಮ'] karma ['ಚೀಲ||ಚೀಲ'] adhikaraNa ['ದಿನ||ದಿನ'] karaNa ['ಕೈಯಲ್ಲಿ||ಕೈ'] Verb : ತರಲು||ತರು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಸಾಮಾನು||ಸಾಮಾನು'] adhikaraNa ['ದಿನ||ದಿನ']	Verb : ಎದ್ದು||ಏಳು kartha ['ರಾಮನು||ರಾಮ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಮುಗಿಸಿಕೊಂಡು||ಮುಗಿಸು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಇತ್ಯಾದಿಗಳನ್ನು||ಇತ್ಯಾದಿ', 'ಸ್ನಾನ||ಸ್ನಾನ', 'ಶೌಚ||ಶೌಚ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಮಾಡಿ||ಮಾಡು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಪೂಜೆ||ಪೂಜೆ'] adhikaraNa ['ದಿನ||ದಿನ'] Verb : ಹಿಡಿದು||ಹಿಡಿ kartha ['ರಾಮನು||ರಾಮ'] karma ['ಚೀಲ||ಚೀಲ'] adhikaraNa ['ದಿನ||ದಿನ'] karaNa ['ಕೈಯಲ್ಲಿ||ಕೈ'] Verb : ತರಲು||ತರು kartha ['ರಾಮನು||ರಾಮ'] karma ['ಸಾಮಾನು||ಸಾಮಾನು'] adhikaraNa ['ದಿನ||ದಿನ']	Success

		Verb : ಹೊರಟನು||ಹೊರಡು destination ['ಪೇಟೆಗೆ||ಪೇಟೆ'] kartha ['ರಾಮನು||ರಾಮ'] adhikaraNa ['ದಿನ||ದಿನ']	Verb : ಹೊರಟನು||ಹೊರಡು destination ['ಪೇಟೆಗೆ||ಪೇಟೆ'] kartha ['ರಾಮನು||ರಾಮ'] adhikaraNa ['ದಿನ||ದಿನ']	
34	ದಾರಿಯಲ್ಲಿ ಹಳೆಯ ಸ್ನೇಹಿತನನ್ನು ಕಂಡು ಅವನನ್ನು ಮನೆಗೆ ಕರೆತಂದನು	Verb : ಕಂಡು||ಕಾಣು karma ['ಸ್ನೇಹಿತನನ್ನು||ಸ್ನೇಹಿತ'] adhikaraNa ['ದಾರಿಯಲ್ಲಿ||ದಾರಿ'] Verb : ಕರೆತಂದನು||ಕರೆತರು destination ['ಮನೆಗೆ||ಮನೆ'] karma ['ಅವನನ್ನು||ಅವನು'] adhikaraNa ['ದಾರಿಯಲ್ಲಿ||ದಾರಿ']	Verb : ಕಂಡು||ಕಾಣು karma ['ಸ್ನೇಹಿತನನ್ನು||ಸ್ನೇಹಿತ'] adhikaraNa ['ದಾರಿಯಲ್ಲಿ||ದಾರಿ'] Verb : ಕರೆತಂದನು||ಕರೆತರು destination ['ಮನೆಗೆ||ಮನೆ'] karma ['ಅವನನ್ನು||ಅವನು'] adhikaraNa ['ದಾರಿಯಲ್ಲಿ||ದಾರಿ']	Success
35	ಇಬ್ಬರೂ ಬಿಸಿ ಬಿಸಿ ಕಾಫಿ ಮಾಡಿಕೊಂಡು ಕುಡಿದರು	Verb : ಮಾಡಿಕೊಂಡು||ಮಾಡು kartha ['ಇಬ್ಬರೂ||ಇಬ್ಬರು'] karma ['ಕಾಫಿ||ಕಾಫಿ'] Verb : ಕುಡಿದರು||ಕುಡಿ kartha ['ಇಬ್ಬರೂ||ಇಬ್ಬರು'] karma ['ಕಾಫಿ||ಕಾಫಿ']	Verb : ಮಾಡಿಕೊಂಡು||ಮಾಡು kartha ['ಇಬ್ಬರೂ||ಇಬ್ಬರು'] karma ['ಕಾಫಿ||ಕಾಫಿ'] Verb : ಕುಡಿದರು||ಕುಡಿ kartha ['ಇಬ್ಬರೂ||ಇಬ್ಬರು'] karma ['ಕಾಫಿ||ಕಾಫಿ']	Success
36	ಹಾವು ಕಚ್ಚಿ ಹುಡುಗ ಸತ್ತನು	Verb : ಕಚ್ಚಿ||ಕಚ್ಚು kartha ['ಹಾವು||ಹಾವು'] Verb : ಸತ್ತನು||ಸಾಯು karma ['ಹುಡುಗ||ಹುಡುಗ']	Verb : ಕಚ್ಚಿ||ಕಚ್ಚು kartha ['ಹಾವು||ಹಾವು'] Verb : ಸತ್ತನು||ಸಾಯು karma ['ಹುಡುಗ||ಹುಡುಗ']	Success
37	ಸೊಳ್ಳೆ ಕಚ್ಚಿದರೆ ಮಲೇರಿಯ ಬರುವುದಂತೆ	Verb : ಕಚ್ಚಿದರೆ||ಕಚ್ಚು kartha ['ಸೊಳ್ಳೆ||ಸೊಳ್ಳೆ'] Verb : ಬರುವುದಂತೆ||ಬರು	Verb : ಕಚ್ಚಿದರೆ||ಕಚ್ಚು kartha ['ಸೊಳ್ಳೆ||ಸೊಳ್ಳೆ'] Verb : ಬರುವುದಂತೆ||ಬರು	Success

		kartha ['ಸೊಳ್ಳೆ||ಸೊಳ್ಳೆ'] karma ['ಮಲೇರಿಯ||ಮಲೇರಿಯ']	kartha ['ಸೊಳ್ಳೆ||ಸೊಳ್ಳೆ'] karma ['ಮಲೇರಿಯ||ಮಲೇರಿಯ']	
38	ಇಲ್ಲಿ ಯಾರೂ ಮೂತ್ರ ವಿಸರ್ಜನೆ ಮಾಡಬಾರದ	Verb : ಮಾಡಬಾರದು||ಮಾಡು kartha ['ಯಾರೂ||ಯಾರು'] karma ['ವಿಸರ್ಜನೆ||ವಿಸರ್ಜನೆ', 'ಮೂತ್ರ||ಮೂತ್ರ'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ']	Verb : ಮಾಡಬಾರದು||ಮಾಡು kartha ['ಯಾರೂ||ಯಾರು'] karma ['ವಿಸರ್ಜನೆ||ವಿಸರ್ಜನೆ', 'ಮೂತ್ರ||ಮೂತ್ರ'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ']	Success
39	ಇಲ್ಲಿ ಹೊಲಸು ಮಾಡಿದವರನ್ನು ಶಿಕ್ಷೆಗೆ ಗುರಿಪಡಿಸಲಾಗುತ್ತದೆ	Verb : ಮಾಡಿದವರನ್ನು||ಮಾಡು karma ['ಹೊಲಸು||ಹೊಲಸು'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ'] Verb : ಗುರಿಪಡಿಸಲಾಗುತ್ತದೆ||ಗುರಿಪಡಿಸು destination ['ಶಿಕ್ಷೆಗೆ||ಶಿಕ್ಷೆ'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ']	Verb : ಮಾಡಿದವರನ್ನು||ಮಾಡು karma ['ಹೊಲಸು||ಹೊಲಸು'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ'] Verb : ಗುರಿಪಡಿಸಲಾಗುತ್ತದೆ||ಗುರಿಪಡಿಸು destination ['ಶಿಕ್ಷೆಗೆ||ಶಿಕ್ಷೆ'] adhikaraNa ['ಇಲ್ಲಿ||ಇಲ್ಲಿ']	Success
40	ನೀನು ಹೇಳಿದರೆ ನಾನು ಬರುತ್ತೇನೆ	Verb : ಹೇಳಿದರೆ||ಹೇಳು kartha ['ನೀನು||ನೀನು'] Verb : ಬರುತ್ತೇನೆ||ಬರು kartha ['ನಾನು||ನಾನು']	Verb : ಹೇಳಿದರೆ||ಹೇಳು kartha ['ನೀನು||ನೀನು'] Verb : ಬರುತ್ತೇನೆ||ಬರು kartha ['ನಾನು||ನಾನು']	Success
41	ನಿನ್ನಷ್ಟೆ ಬೇಗ ನಾನು ನಡೆಯಲಾರೆ	Verb : ನಡೆಯಲಾರೆ||ನಡೆ kartha ['ನಿನ್ನಷ್ಟೆ||ನಿನ್ನ', 'ನಾನು||ನಾನು']	Verb : ನಡೆಯಲಾರೆ||ನಡೆ kartha ['ನಿನ್ನಷ್ಟೆ||ನಿನ್ನ', 'ನಾನು||ನಾನು']	Success
42	ಮುದುಕರು ಮೂಲೆಯಲ್ಲಿ ಕುಳಿತು ಹರಟೆ ಹೊಡೆದರು	Verb : ಕುಳಿತು||ಕುಳಿರು kartha ['ಮುದುಕರು||ಮುದುಕ'] adhikaraNa ['ಮೂಲೆಯಲ್ಲಿ||ಮೂಲೆ'] Verb : ಹರಟೆಹೊಡೆದರು||ಹರಟೆಹೊಡೆ	Verb : ಕುಳಿತು||ಕುಳಿರು kartha ['ಮುದುಕರು||ಮುದುಕ'] adhikaraNa ['ಮೂಲೆಯಲ್ಲಿ||ಮೂಲೆ'] Verb : ಹರಟೆಹೊಡೆದರು||ಹರಟೆಹೊಡೆ	Success

		kartha ['ಮುದುಕರು||ಮುದುಕ'] adhikaraNa ['ಮೂಲೆಯಲ್ಲಿ||ಮೂಲೆ']	kartha ['ಮುದುಕರು||ಮುದುಕ'] adhikaraNa ['ಮೂಲೆಯಲ್ಲಿ||ಮೂಲೆ']	
43	ಅವನಿಗೆ ಬರುವಂತೆ ಹೇಳು	Verb : ಬರುವಂತೆ||ಬರು karma ['ಅವನಿಗೆ||ಅವನು'] Verb : ಹೇಳು||ಹೇಳು kartha ['ನೀನು||ನೀನು'	Verb : ಬರುವಂತೆ||ಬರು karma ['ಅವನಿಗೆ||ಅವನು'] Verb : ಹೇಳು||ಹೇಳು kartha ['ನೀನು||ನೀನು'	Success
44	ಪ್ರಜೆಗಳು ರಾಜನಲ್ಲಿ ಬೇಡಿಕೊಂಡರು	Verb : ಬೇಡಿಕೊಂಡರು||ಬೇಡು kartha ['ಪ್ರಜೆಗಳು||ಪ್ರಜೆ'] karma ['ರಾಜನಲ್ಲಿ||ರಾಜ']	Verb : ಬೇಡಿಕೊಂಡರು||ಬೇಡು kartha ['ಪ್ರಜೆಗಳು||ಪ್ರಜೆ'] karma ['ರಾಜನಲ್ಲಿ||ರಾಜ']	Success
45	ಕಾರು ಬಸ್ಸು ಡಿಕ್ಕಿ ಹೊಡೆದು ಐದು ಜನ ಸತ್ತರು	Verb : ಡಿಕ್ಕಿಹೊಡೆದು||ಡಿಕ್ಕಿಹೊಡೆ kartha ['ಕಾರು||ಕಾರು', 'ಬಸ್ಸು||ಬಸ್ಸು'] Verb : ಸತ್ತರು||ಸಾಯು karma ['ಜನ||ಜನ']	Verb : ಡಿಕ್ಕಿಹೊಡೆದು||ಡಿಕ್ಕಿಹೊಡೆ kartha ['ಕಾರು||ಕಾರು', 'ಬಸ್ಸು||ಬಸ್ಸು'] Verb : ಸತ್ತರು||ಸಾಯು karma ['ಜನ||ಜನ']	Success

Table 10.1 Test cases for Semantic Parser for Kannada with the Results

The Test cases had 100% accuracy and the kaaraka classification is shown in the GUI form having the verb in the middle, encircled by the Kaarakas.

CHAPTER-11

CONCLUSION

All Natural language processing (NLP) is primarily concerned with getting computers to perform useful and understanding tasks with human languages. In natural language processing system first the words are placed into a structured form that leads to semantic correct sentence. Typical applications for natural language processing include the following.

- A better human-computer interface that could convert from a natural language into a computer language and vice versa.
- A translation program that could translate from one human language to another (Kannada to Telugu, for example). Even if programs that translate between human languages are not perfect, they would still be useful in that they could do the rudimentary translation first, with their work checks and corrected by a human translator. This cuts down on the time for the translation.
- Programs that could check for semantics and writing techniques in a word processing document.

CHAPTER-12

FUTURE ENHANCEMENTS

The following can be the future enhancements that can be thought of:

- Finding the Kaaraka using different methodologies.
- Based on the results, Kannada to telugu translation can be performed.
- Trying to test with the chapters with more sentences which has metaphoric meaning and checking out the results.