A Dialog Act Tagger for Telugu

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science by Research
in
Computer Science and Engineering

by

Dowlagar Suman
201307678
suman.d@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2016

Copyright © Dowlagar Suman, 2016
All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis titled "A Dialog Act Tagger for Telugu" by Dowlagar Suman has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Dr. Radhika Mamidi

To my loving Husband, Parents and In-Laws

Acknowledgments

I would like to express my deep gratitude to my supervisor Dr. Radhika Mamidi for her guidance and encouragement. I must be very fortunate to have an advisor who gave me the freedom to explore on my own, and at the same time her guidance helped me to recover when ever my steps faltered. Her vast knowledge in Computational Linguistics especially in Dialog Systems helped me to build two strong papers which in-turn contributed in completing my Masters. Her patience and support helped me overcome many crisis situations and finish this thesis.

I am indebted to Language Technologies Research Center (LTRC) faculty - Prof Dipti Misra Sharma, Dr. Soma Paul and Dr. Manish Shrivastava for guiding me through the basic concepts of natural language processing and computational linguistics which helped me to understand my research area better. They also helped me to understand the basic tools which played a crucial role in my research work.

I am also grateful to my friends who helped me to build the corpus necessary for my research and also their guidance helped me to understand various concepts which benefited my thesis.

Most importantly, none of this would have been possible without the love and patience of my family. I formally thank them for their support and strength through these years. I would like to personally thank my husband. From the beginning, he is the one who had the confidence that I can join such an esteemed institution and successfully complete my research work. He guided me through the difficult times in my life and never let me fail. I thank him immensely for that.

Abstract

In a task oriented domain, recognizing the intention of a speaker is important so that the conversation can proceed in the correct direction. This is possible only if there is a way to label the utterance with its proper intent. One such labeling technique is Dialog Act(DA) tagging. The main goal of this thesis is to build a Dialog Act tagger for the Telugu corpus. This work focuses on discussing various n-gram DA tagging techniques so as to tag the Telugu data.

The n-gram DA tagging methods proposed earlier for English will not work for free word order languages like Telugu as English language follows strict subject–verb–object(SVO) syntax. This thesis explains in detail about the two DA tagging methods for tagging free word order languages preferentially Telugu. In this thesis, we propose a method to perform DA tagging for the Telugu corpus using advanced machine learning techniques combined with karaka dependency relation modifiers.

The use of karaka dependencies for free word order languages like Telugu helps in extracting the modifier-modified relationships between words or word clusters for an utterance. The modifier-modified relationships remain fixed even though the word order in an utterance changes. These extracted modifier-modified relationships appear similar to n-grams. Later, statistical machine learning methods are applied to predict DA for an utterance in a dialog.

The first method uses n-gram karakas with back-off as n-gram language modeling technique at n-gram level and Memory Based Learning at utterance level. In the second method, we use syntactic features such as anaphora resolution, conjunct identification and using modifier-modified relationship rules we automatically extract n-gram karakas. Then we apply language modeling (LM) with Hidden Markov Model (HMM) method for DA tagging.

The proposed methods are compared with several baseline tagging algorithms. The results show that the proposed methods perform better DA tagging for free word order languages like Telugu.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

Dialog is the most fundamental and privileged arena of language. It is certainly the first kind of language we most commonly indulge in. The term 'dialog' origins from the Greek word *dialogos* which means conversation. The study of dialog which is the communication between two or more participants, both spoken (either face-to-face or at a distance via telephone) and written remains an interesting challenge within the field of Computational Linguistics. Dialog has been studied from a variety of perspectives, including linguistic and psychological, but it is relatively recently that this study has extended to the computational.

1.1 Dialog System

A dialog system or conversational agent (CA) [13, 28, 1, 2] is a computer system intended to converse with a human, with a coherent structure. The crucial use of a dialog system is to convert simple yet complicated tasks from manual to automated.

Figure 1.1 shows a typical dialog system architecture from [13]

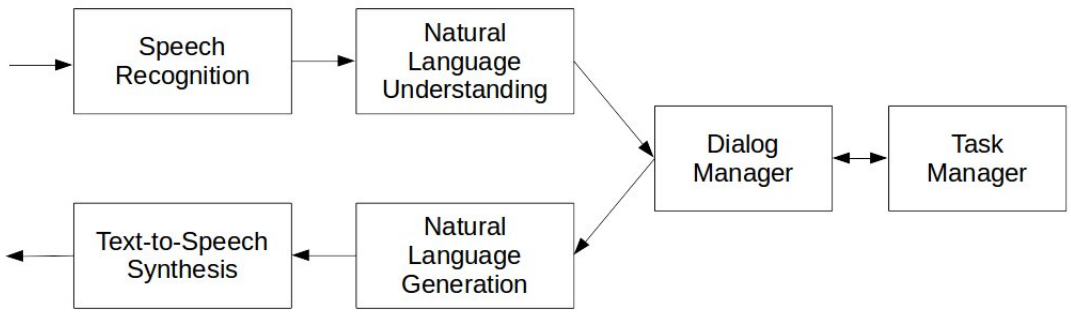

Figure 1.1 Figure showing Dialog system architecture

1. The user speaks, and the input is converted to plain text by the system's input recognizer/decoder.

2. The text is analyzed by a natural language understanding unit(NLU), which may include: proper name identification, part of speech tagging, syntactic/semantic parser.

3. The semantic information is analyzed by the dialog manager, that maintains the history and the state of the dialog and manages the general flow of the conversation. Usually, the dialog manager contacts one or more task managers, that have knowledge of the specific task domain.

4. The dialog manager produces output using natural language generator(NLG). It could be said an NLG system is like a translator that converts a computer based representation into a natural language representation.

5. A text-to-speech (TTS) system converts normal language text into speech.

The basic dialog system are only capable of limited domain specific conversations. Such dialog systems are based only on what is needed to fill slots and only a particular pattern is followed while responding the questions. In order to be useful for more than just form filling applications a dialog system must be able to do things like decide when a user has asked a question, made a proposal, rejected a suggestion and need to be able to ask clarification questions and suggest plans. So the conversational agent needs sophisticated models of interpretation and generation. For better interpretation, the dialog system must understand the actual intent of what the user says. This can be done with the help of Dialog acts.

1.2 Dialog Acts

To make dialog managers and core components of the Natural Language Understanding and Natural Language Generation more manageable and reusable a description of human dialogs is essential.

A dialog or conversation consists of a sequence of turns, each of which consists of one or more utterance. According to [3] an utterance in a dialog is a kind of action being performed by the speaker. Austin called these kinds of actions as **speech acts**.

Later on, [24] suggested that all speech acts can be classified into one of five major classes.

- **Assertives:** Commit the Speaker to the truth of some proposition (e.g. stating, claiming, reporting, announcing);

- **Directives:** Attempts to bring about some effect through the action of the Hearer (e.g. ordering, requesting, demanding);

- **Commissives:** Commit Speaker to some future action (e.g. promising, offering, swearing to do something);

- **Expressives:** The expression of some psychological state (e.g. thanking, apologizing, congratulating);
- **Declarations:** Speech acts whose successful performance brings about the correspondence between the propositional content and reality (e.g. resigning, sentencing, dismissing)

A **Dialog Act** is a specialized speech act. *For example, Question is a speech act, but Question_on_hotel is a dialog act.* Dialog acts are different in different dialog systems.

The process of understanding and generating the dialogs is known as dialog modeling. In dialog modeling, to understand the dialogs, speaker's intent must be recognized. The recognition of the speaker's intent is done with the help of Dialog Acts. Dialog Acts is a tagset that classifies utterances based on pragmatic, semantic and syntactic features.

The dialog acts represent the meaning of an utterance in the context of a dialog, where the context is divided into several types, with both global and local views: linguistic, semantic, physical, social and cognitive. A human conversation with its corresponding dialog acts is given in figure 1.2.

Speaker		Dialog in Telugu with gloss and its English Translation	DA tag
విద్యార్థి	:	నమస్కారం సర్.	
student	:	hello sir	GREETINGS
Student	:	Hello sir.	
లైబ్రేరియన్	:	నమస్కారం.	
Librarian	:	hello	GREETINGS_REPLY
Librarian	:	Hello.	
విద్యార్థి	:	ఎన్ని రోజులలో ఈ పుస్తకం ఇవ్వేయాలి?	
student	:	how_many days_in this book give_should	INFO_REQUEST
Student	:	By when this book has to be returned?	
లైబ్రేరియన్	:	ఒక్క నెలలో.	
Librarian	:	one month_in	ANSWER
Librarian	:	In one month.	
విద్యార్థి	:	ధన్యవాదములు.	
student	:	thanks	GREETINGS_EOC
Student	:	Thank you.	

Figure 1.2 An example dialog with DA tags

1.3 Motivation

In order to write a dialog system, the understanding of dialogs is very important. Dialog acts make the understanding easier. Assigning a DA tag to an utterance covers not only the explicit

information obtained from the words in the utterance but also the implicit information present in the utterance. This task of identifying the implicit information conveyed by the user is known as Intention Recognition. Currently, the generic dialog systems developed focus on the words in the utterance but not the intent of the utterance. Dialogue acts can be used to recognize the intention of the user and thus differentiate situations where the user is requesting some information from situations where the user is simply giving some information or back-channels. In dialogs, there comes a situation such that, even the utterance has the same words its intent might be different when the same utterance is given in different contexts. These problems cannot be tackled by considering just the words and their meanings in the utterance. Hence recognition of intent is necessary for understanding and further processing of the utterance in a dialog.

1.4 Literature Survey

Earlier, research in DA tagging was limited to linguistic domain, but now with the help of statistics, machine learning and pattern matching, automated DA tagging with various DA recognition approaches [16] have come into existence. Out of all the statistical approaches, the use of n-grams for DA tagging gave better accuracy [12].

1.4.1 DA Tagging using N-gram methods

Some of the DA tagging methods using n-gram cues include:

- Word based DA tagging in [11] proved that Dialogue acts can be inferred from their constituent words to an accuracy of around 50% using a very simple unigram model, implying that better performance should be possible using a more involved N-gram Markov model.
- [19] considers n-gram dialog act tagging method by taking n-grams of order 1 to 3.
- On the other hand, [26] used n-grams with *predictivity criterion* for DA tagging which shows that instead of considering all n-grams, to take only those which surpass the threshold. But [26] pointed out that the threshold concept works only if the corpus is large enough.

1.4.2 DA Tagging using Syntactic features

- [15] proved that both the syntactic relations as well as the semantic VerbNet-based relations included in the utterances can be extracted and added to the feature sets for the recognition task.

- [17] explicitly proposed global syntactic features derived from automatic parsing of the sentence which will help in DA tagging of the sentence.

[18] and [23], proved that memory based learning techniques can be used for DA tagging. Other prominent methods for DA tagging include Naive Bayesian interpretation [22], and Hidden Markov Models [25].

The methods discussed above were developed by considering the dialog corpus in English which follows a strict SVO syntax. Here we test these DA tagging techniques by considering our dialog corpus in Telugu. We observe that the DA tagging techniques initially proposed for English cannot be directly applied to free word order languages like Telugu. In Telugu, like most other South Asian languages, the word order of grammatical functions like subject and objects are largely free. Internal changes in the sentences or the phrases will not affect the grammatical functions of the nominals. For free word order language we cannot directly apply statistical n-gram methods. Before we use n-gram methods, a grammatical model must be implemented such that the model restricts the movement of the words or the word clusters by establishing some relationships between them [6].

The grammatical model used to establish the relationships between the words or the word clusters is based on *Paninian framework* [5]. Paninian framework treats a sentence as a series of modifier-modified elements. A sentence is supposed to have a primary modified, which is generally the main verb of the sentence. The elements modifying the verb participate in the action specified by the verb. The participant relations with the verb are called *karakas*. The appropriate mapping of the syntactic cues helps in identifying the appropriate *karakas* ('participants in an action'). *Karakas* are the grammatical relations that bind the modifier and the modified elements. Through Paninian framework we know that the relationship between words does not change even though the word order changes. Therefore this framework would be better suited for sentence analysis of Telugu, which is relatively a free word order language.

Why karaka based dependencies are chosen over Stanford dependency relations?

According to [4] Indian Languages(ILs) are morphologically rich and have a relatively flexible word order. For such languages syntactic subject-object positions do not always elegantly explain the varied linguistic phenomena. In fact, there is a debate in the literature whether the notions 'subject' and 'object' can at all be defined for ILs [20]. Behavioral properties are the only criteria based on which one can confidently identify grammatical functions in Telugu. It is difficult to exploit such properties computationally. Marking semantic properties such as thematic role as dependency relation is also problematic. Thematic roles are abstract notions and will require higher semantic features which are difficult to formulate and to extract as well. So, thematic roles are not marked. On the other hand, the notion of karaka relations provides us a level which while being syntactically grounded also helps in capturing some semantics.

1.5 Contribution of the Thesis

The major contribution to the thesis are:

1. We proposed a model to perform DA tagging for free word order languages
   (a) We extracted modifier-modified karaka dependency relationships between words (or word clusters) in the utterance.
   (b) The extracted modifier-modified karaka dependencies follow an n_gram_karaka format so that they appear similar to n-grams. Hence DA tagging approaches proposed for positional based languages can be applied.
   (c) For the obtained n_gram_karakas we used various Bayesian language modeling approaches such as Katz's back-off in combination with k Nearest Neighbor for DA tagging.
2. We tested our approach on the library corpus. The corpus is known as ASKLIB and it consists of interactions between students and a librarian.
3. Later we extended the approach by adding a context handler module.
   (a) Here we applied Hidden Markov Model for Contextual DA tagging
   (b) Preprocessing such as speaker recognition using anaphora resolution is done before extracting karaka dependencies
   (c) A combination of language modeling approach is applied for DA tagging.

1.6 Thesis Organization

The remainder of the thesis is organized as follows:

- **Chapter 2** - A Semi Supervised Dialog Act Tagging for Telugu: This chapter explains about various classification algorithms that were proposed for English and how they respond when they are applied to Telugu corpus. Karaka dependency approach with language modeling and memory based learning is discussed here.
- **Chapter 3** - A Contextual Dialog Act Tagging for Telugu: This chapter about the importance of contextual information in Dialog Act Tagging. Further, we also present a method to identify contextual information.
- **Chapter 4** - Applications: This chapter explains the use of Dialog Act Tagging in various fields.
- **Chapter 5** - Conclusions and Future Work: A brief summary of the contributions of this research and the scope for future work in this direction is presented in this chapter.

Chapter *2*

Dialog Corpus and Tagset

2.1 ASKLIB Corpus

At present, there is no available corpus related to task oriented Telugu dialogs. Our work started with the construction and acquisition of the dialogs in Telugu. The corpus is developed by us using various observations, techniques and interactions.

The focus was on task oriented, domain dependent dialogs with *'Library'* as the domain. We named the corpus as ASKLIB. ASKLIB consists of nearly 225 dialogs that took place between students and the librarian. This corpus is also collected by frequently visiting different libraries and observing how people interact with the librarian. The data acquisition was also done through the *Wizard of Oz* technique. 27 active participants were told to assume the scenario of a library and were asked to write a few generic 2 party conversations. After the corpus acquisition, we observed that the dialogs pertaining to the library domain could be broadly classified into 4 types, viz. ISSUE, REISSUE, RETURN, ENQUIRY. In other words, a person's interaction with a librarian can result in either issue, reissue or return of a book or any enquiry related to a book/the library. The data that was collected has undergone various layers of automated spell checking using CALTSLAB Spell Checker (`http://caltslab.uohyd.ernet.in/spell\_checker.php`) and manual spell checking to make the corpus reliable and correct.

Words	**Dialogs**	**Utterances per Dialog**
12826	225	7-9

Table 2.1 Corpus Statistics

Table 2.1 gives the information about the number of words, dialogs and utterances per dialog present in ASKLIB corpus. The information the percentage of tag in the corpus, description of each tag with an example (written originally in Telugu) translated to English.

2.2 Modified DAMSL Tagset

The DA tagset is based on DAMSL (Dialogue Act Markup in Several Layers) [8] and some domain dependent tags [10].

DAMSL

A major problem with speech act theory is that it attempts to capture the utterance's purpose with one label [8]. This is a problem because utterances can simultaneously respond, promise, request and inform. DAMSL addresses this problem by allowing multiple labels in multiple layers to be applied to the utterance. Thus an utterance might simultaneously perform actions such as responding to a question, confirming understanding, promising to perform an action and informing.

DAMSL is one of the domain independent tagsets used for DA Tagging. DAMSL was initially designed to be universal. Its annotation scheme is composed of four levels (or dimensions): communicative status, information level, forward looking functions and backward looking functions. The communicative status states whether the utterance is uninterpretable, abandoned or it is a self-talk. This feature is not used for most of the utterances. The information level provides an abstract characterization of the content of the utterance. It is composed of four categories: task, task-management, communication-management and other-level. In the forward looking functions the annotators are allowed to look ahead in the dialog to determine the effect an utterance has on dialog. The backward looking functions show the relationship between the current utterance and the previous dialogue acts, such as accepting a proposal or answering the question. DAMSL is composed of 42 DA classes.

DAMSL tags are high-level and designed to be applicable to various types of dialogs. The idea is that for a particular domain, these classes could be further subdivided into acts that are relevant to the domain, The common level of abstraction across domains, would allow researchers to share data in a way that would not be possible if everyone developed their own scheme.

The focus of DAMSL has primarily been on task-oriented dialogs, where the participants are focused on accomplishing a specific task.

Domain Dependent Tags

The ASKLIB corpus is first tagged with the DAMSL tagset. It is observed that all the 42 tags are not used for the data. So the tags that are relevant to our tagset are chosen from the DAMSL tagset. After choosing a set of applicable tags to the DAMSL tagset it was clear that DAMSL "a generic tagset" is not enough for domain specific applications. So, in our tagset, a few domain dependent tags are included along with the DAMSL tagset. The reason is that, after the corpus acquisition, we observed that the dialogs pertaining to the library domain could be broadly classified into 4 types, viz. ISSUE, REISSUE, RETURN, ENQUIRY. To make the tagset more application specific, the above three categories i.e. ISSUE, RETURN, REISSUE are included as tags. The fourth category i.e. ENQUIRY is already a part of the DAMSL tagset which is present in the DAMSL tagset as INFO REQUEST.

At present, our tagset consists of a total of 21 tags. The tagset and its related information is given in table 2.2.

Tag list	**%**	**Description of Tags**	**Example of utterances translated to English**
RETURN	2.93	Utterance intent is to return the book	*I am returning this book.*
TIME_ASSERT	0.55	Speaker makes a claim with respect to library timings	*Now the time is 6.30pm.*
ISSUE_INFO_REQUEST	1.92	Utterance is bound to provide answer related to issue of book	*Do you want to issue this book?*
ISSUE	3.88	Utterance intent is to issue the book	*Issue this book to me*
REISSUE_INFO_REQUEST	1.92	Utterance is bound to provide answer related to reissue of book	*Are you willing to reissue this book?*
ISSUE_ASSERT	3.68	Speaker makes a claim while issuing the book	*You have to return this book in a month, or else you will be charged.*
ACCEPT_ACKNOWLEDGE	6.46	Utterance indicates speaker has understood and accepted the stated fact.	*Ok sir, I will return this book in a month.*
ASSERT	0.25	Speaker states a general fact with out any relation to issue, reissue, return etc	*Days have passed since I saw you.*
COMMIT	9.99	Utterance states that speaker is committing to perform the action in future.	*Ok sir, I will come tomorrow.*
GREETINGS_REPLY	12.26	Utterance is replying to a greet so as to maintain the conversation	*I am fine.*
ANSWER	2.37	Utterance is answering to the question	*The author of this book is Dan Jurafsky.*
GREETINGS	7.67	Utterance states that the conversation is started	*Good Morning sir.*
ACCEPT	6.16	Utterance shows speakers agreement to the proposal or claim	*Ok sir.*
INFO_REQUEST	6.76	Utterance that is a generic question without relation to any domain specific task	*How are you sir?*
RETURN_ASSERT	3.08	Speaker makes a claim while issuing the book	*I am deleting this book from your account.*
GREETINGS_EOC	12.26	Utterance states that the conversation is completed	*Thank you.*
RETURN_INFO_REQUEST	2.32	Utterance is bound to provide answer related to return of book	*Do you want to return this book?*
REISSUE_ASSERT	2.77	Speaker makes a claim while returning the book	*I am extending the book's due date.*
REISSUE	3.13	Utterance intent is to reissue the book	*Please, reissue this book to me.*
ACTION_DIR	9.64	Utterance intent is to make hearer, perform an action	*Give me the id card and the book.*

Table 2.2 Description of our Tagset

Chapter *3*

A Semi Supervised Dialog Act Tagging for Telugu

3.1 Problem Definition

Telugu is a free word order language. The existing n-gram cue based methods are mainly developed for English. When these are applied to DA tagged Telugu dialogs, the baseline accuracy is not reached, because n-gram methods are position dependent. In this chapter, a new method is proposed for DA tagging in Telugu using n-gram karakas with back-off and Memory Based Learning such as kNN. The novel method is compared with n-grams and a combination of unigram method.

3.2 Classification Algorithms Discussed

Of all the methods developed for DA tagging, the easiest way to simply search whether the test data is present in the given training data or not. This is done by taking each utterance in the test data and verifying whether it is present in the training data utterances or not. The problem with this method is the unlikely occurrence of the same utterance in both the training data and the test data. It will also occupy a lot of memory (for huge corpus) as each and every unique utterance with its corresponding tag is stored in the training data. As the huge corpus was difficult to handle, the focus was made on words. The problem of considering only words is that the words do not contain any local contextual information. Later, the methods were extended to n-grams. The n-gram methods consist of splitting the utterances into a sequence of n words. These n-grams are called cues. In cue based n-gram DA tagging techniques, the n-grams obtained from the training data act as cues. By matching n-grams of the training data with n-grams of test data, an appropriate tag to the test data was given. Also, the size of unique n-grams obtained from the training data is very less when compared to the size of unique utterances in the training data.

Here, we discuss about two baseline classification algorithms. They are:

1. DA tagging using n-grams,

2. Combinations of unigrams with Naive Bayesian plus k Nearest Neighbors (kNN)

3.2.1 N-gram Method

Of all the approaches in DA tagging, cue based n-gram tagging methods are proven to be the easiest and the most powerful DA tagging scheme. In n-gram methods, the tag of the test utterance is obtained by converting the test utterance into a set of contiguous sequence of n words. Thus, allowing the obtained sequence to be compared with the other n-gram training sequences using efficient algorithms. These DA tagging schemes are mostly developed for corpus related to English language because English has a fixed syntactic structure. For English, in both the training and the test data, the position of the words in an utterance will remain mostly the same. By comparing the n-grams obtained from training data with the n-grams obtained from the test data, the best tag for the test utterance is obtained. As this method gives high accuracy for English, the same method is applied to the Telugu ASKLIB corpus.

For choosing the best tag, Naive Bayesian interpretation is used,

$$\hat{T} = \underset{T}{\operatorname{argmax}} \frac{P(U,T)}{P(U)} \tag{3.1}$$

where $\hat{T}$ is the correct tag, from the tagset T for the utterance U.

For n-grams, the above equation is modified to

$$\hat{T} = \underset{T}{\operatorname{argmax}} \prod_{i=1}^{N} \frac{P(w_i,T)}{P(w_i)} \tag{3.2}$$

where w_i represents the n-gram sequence and N is equal to the list of n-grams obtained for the utterance U.

3.2.2 Combination of Unigrams and kNN

The n-gram method will have a low accuracy for free word order languages like Telugu. In free word order languages like Telugu, even though the speaker's intent might be the same, the position of the words (the syntactic structure) might change. Hence the n-gram method will not work. So word position independent methods must be considered, One such method is to extract n-grams by considering combinations of unigrams. For example: In the combinations of unigrams for n=2 we get bigrams. Here, each word will appear with all the other words in the given utterance and with itself. The problem with this approach is that the time complexity increases when n value increases. As low order n-grams will capture less context, for further processing Memory Based Learning (MBL) method such as k Nearest Neighbors (kNN) [9] is applied. In MBL the pattern of the training data will be tested against the pattern of the test

data with the word sequence independence as one of the criteria. This method gives the tag of the nearest training utterance to the test utterance without considering the position of words in an utterance in both the training data and the test data.

In combinations of unigrams and kNN method, for choosing the best tag, Naive Bayesian interpretaion in combination with kNN is used.

The above equation 3.2 will undergo a small modification by considering combinations of unigrams.

$$\hat{T} = \underset{T}{\operatorname{argmax}} \prod_{i=1}^{N_{all}} \frac{P(w_i, T)}{P(w_i)} \tag{3.3}$$

where N_{all} is equal to the list of combinations of unigrams

The kNN method is

$$DA(U_{test}) = DA(U_{train}) \\ if : |U_{test} - U_{train_i}| = min\ |U_{test} - U_{train_N}| \\ \text{for N = 1,2,...(no of unique train utterances)} \tag{3.4}$$

where U_{test}, U_{train} represents train and test utterances respectively

3.3 Our Approach

In Paninian framework [5], for free word order Indian languages like Telugu, it is proven that the karaka based dependency relations will remain the same even though the syntactic structure of the sentence changes. By using these karaka dependencies, the syntactico-semantic relationships between the words is captured in the *modifier-karaka-modified* format. On careful inspection, this format seems similar to the n-grams with karaka relationships between them. After extracting all the n-grams with karaka dependencies, language modeling technique i.e. Katz's back-off model (at n-gram level) and memory based learning technique i.e. kNN (at utterance level) will be applied. The combination of the above two will give the best tag to test utterances when compared to the above models.

Why Katz's back-off model? Why not just check for karaka dependencies with smoothing algorithms and tag it?

Linguistically speaking, for an utterance to be given a specific tag, each and every word in the utterance must contribute to the tag. So in the test utterance, after extracting all the words with karaka relations between them, there will be certain words whose dependencies are missed. It might be due to any of the following reasons given below.

- Some karaka dependencies are currently not annotated in the Telugu tree bank.

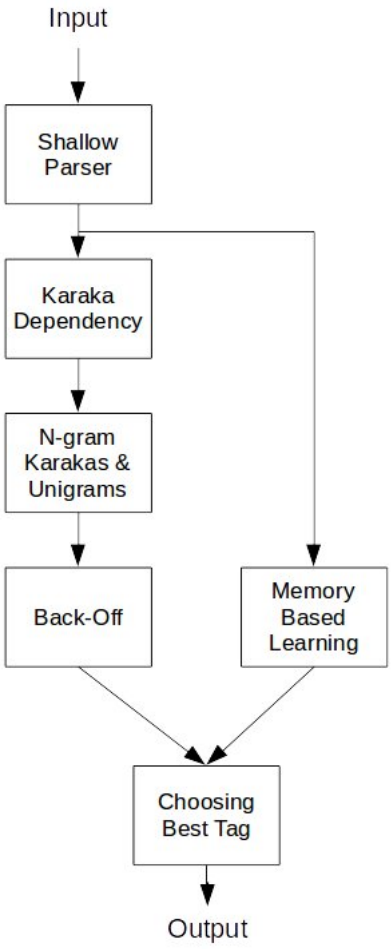

Figure 3.1 Figure showing the Karaka Dependency method with Back-off and Memory Based Learning for DA tagging.

- The n-grams have a different karaka dependency between them.
- Telugu is a morphologically rich language. In n-gram karakas, the root word and the karaka dependency might be the same but the suffix might change.
- The words themselves are not present in the training data.

Hence, for those n-gram with karaka relationships the karaka dependencies form the test data are dropped. It is verified that they are mostly unigrams. Hence back-off to unigrams is considered. One of the basic back-off techniques is Katz's back-off model, which is presently proven as an effective LM algorithm for the given training and test data.

3.3.1 Preprocessing

3.3.1.1 Shallow Parsing

Shallow parsing (also chunking, "light parsing") is an analysis of a sentence which identifies the constituents. It is a technique widely used in natural language processing. It is similar to the concept of lexical analysis for computer languages.

The training data obtained from ASKLIB corpus is run through the shallow parser tool (`http://ltrc.iiit.ac.in/analyzer/telugu/`). The shallow parser [21] does the following steps

1. Tokenizing: splitting the data based on white spaces, punctuation marks etc.

2. Morphing: analyzes the inflected word and provides information such as root word or stem and its constituent morphemes with which the original word was constructed. Building morph analyzers for highly inflectional languages (Indian Languages) is crucial for applications such as Machine Translation (MT) and dialog based natural language understanding systems. In our approach, the morph analyzer is use-full for back-off language modeling techniques.

3. Part of speech (POS) tagging: POS tagging is the process of labeling a part of speech or other lexical class marker to each and every word in a sentence. The part of speech tagger is used for constraint based karaka dependency approach.

4. Chunking: Chunking identifies simple noun phrases, verb groups, adjectival phrases, and adverbial phrases in a sentence. This involves identifying the boundary of chunks and the label. Chunking is used because, the karaka dependencies are constructed on chunks.

The input given to the shallow parser and its modified output is shown in example 1.a.

Example 1.a:
Shallow Parser Input is:
(Student) విద్యార్థి : నేను ఈ పుస్తకము తీసుకోవాలి.
Shallow Parser Output (in wx-notation) [1] is:
(Student) vixyArWi : [nenu#PRP] [I#DEM puswakaM#NN] [wIsukovAli#VM]

3.3.2 Extracting n-grams with Karaka Dependencies

1. Telugu tree bank (which consists of huge data containing karaka dependencies) and the manual dependency parsed output for the ASKLIB corpus is used as annotated data for extracting karaka dependencies.

[1] Words are in wx format (sanskrit.inria.fr/DATA/wx.html). All the examples given in the chapter are from Telugu language

2. We found out that the Telugu tree bank is not enough for extracting karaka dependencies as it developed on entirely other domain. So mostly karaka dependencies are extracted using manual dependency parsed output for ASKLIB corpus.

3. The karaka dependencies are extracted for the words present in each utterance (for both training and test data). An example of karaka dependencies is shown in Figure 3.2

4. The karaka dependencies between the words or word clusters will be converted to *modifier-karaka-modified* format as shown in table 3.1.

5. By observation, the format will be similar to n-grams with just karakas present between them. They are abbreviated as n-gram karakas.

Telugu utterance WX-format	VixyArWi : nenu I puswakaM wIsukuMtAnu
Gloss Translation	Student : I this book take+future Student : I will take this book
Karaka Dependencies extracted in modifier-karaka-modified format	[WisukuMtAnu]-k1-[nenu],[WisukuMtAnu]-k2-[I puswakaM]

Table 3.1 Example of n-gram karaka format extraction for an utterance

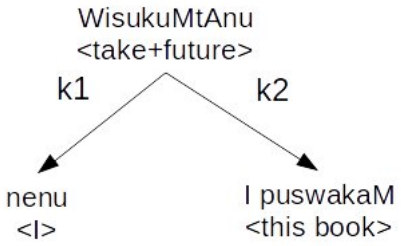

Figure 3.2 Showing karaka dependencies for an example utterance given in table 3.1

3.3.3 Language Modeling Technique and kNN

1. Language Modeling technique:

   (a) After extracting n-grams with karaka dependencies for all the training and test utterances, Katz's back-off model [27] is applied.

   (b) In Katz's back-off model, it is verified whether n-gram karakas are present or not in the training data, if present then tag probabilities are updated

   (c) If not, as Telugu is a morphologically rich language, we back-off to morphed n-gram karakas.

(d) If the morphed n-gram karakas are not present in the training data then, it is known that particular dependencies are not annotated in the training data

(e) Then the n-gram karakas will be decomposed to non karaka dependency words i.e. we back-off to unigrams and further to unigram morphs.

(f) When all the above steps fail, smoothing method is considered.

2. k Nearest Neighbor:

(a) To capture the utterance level information and to provide a strong ground for the respective tag, kNN is used.

(b) kNN technique applied is same as given in equation 3.4

Katz's back-off model for n-gram karakas with back-off to morphed n-gram karakas is given in equation 3.5.

Katz's back-off model for non dependencies i.e. unigrams with back-off to morphed unigrams is given in equation 3.6.

$p_{bo}(n\_gram\_karaka_i, T) =$

$$\begin{cases} discount_1 \ \frac{C(n\_gram\_karaka_i, T)}{C(n\_gram\_karaka_i)} \\ if \ C(n\_gram\_karaka_i, T) > 0 \\ \\ \alpha_1 \ \frac{C(morph\_n\_gram\_karaka_i, T)}{C(morph\_n\_gram\_karaka_i)} \\ if \ C(morph\_n\_gram\_karaka_i, T) > 0 \\ (otherwise) \end{cases} \tag{3.5}$$

$p_{bo}(unigram_i, T) =$

$$\begin{cases} discount_2 \ \frac{C(unigram_i, T)}{C(unigram_i)} \\ if \ C(unigram_i, T) > 0 \\ \\ \alpha_2 \ \frac{C(morph\_unigram_i, T)}{C(morph\_unigram_i)} \\ if \ C(morph\_unigram_i, T) > 0 \\ (otherwise) \end{cases} \tag{3.6}$$

where $discount_1, discount_2$ are discounts obtained by Good Turing estimation as C^*/C and α_1, α_2 are back-off weights

3.3.4 Experiments and Results

The testing is done on our ASKLIB corpus using all the three algorithms. The corpus consisting of 225 dialogs is divided into four parts, each time one part is used for the testing and the remaining are combined for the training.

Firstly, the experiment is run on an n-gram method with Bayesian interpretation where n-grams of length 1-3 are considered. The results are shown in table 3.2

n-grams	**Accuracy**
n = 1	58.17%
n = 2	54.71%
n = 3	47.79%

Table 3.2 Accuracy obtained for n-grams.

From the results in table 3.2, it is clear that the n-grams are not sufficient for DA tagging for free word order languages like Telugu. When the n-gram length increases, there is a decrease in accuracy. The explanation is that, as Telugu is a free word order language, even though the intent is the same, the position of words will not be the same in both the training data and the test data. Hence this method will not work for free word order languages like Telugu.

Next, n-grams with position independence are considered by taking combinations of unigrams from the training data and the test data combined with the kNN algorithm. The results are shown in table 3.3

Combinations of unigrams with kNN	**Accuracy**
n = 1 + kNN	66.67%
n = 2 + kNN	71.80%

Table 3.3 Accuracy obtained for combinations of unigrams.

From the results in table 3.3, it can be seen that, combinations of unigrams combined with kNN does show a good response in accuracy. The problem with this method is that, for an utterance of n words, each word is repeated n times which results in an increase in time complexity.

Now consider n-grams with karaka dependencies using back-off and kNN. This method is applied on ASKLIB corpus. The results are as shown in table 3.4

For free word order languages like Telugu, it is known that the karaka dependencies remain the same even though the word order changes. Hence there is no need to consider the methods

n-gram karakas using back-off and kNN	**Accuracy**
n-gram karakas + back-off + kNN	73.34%

Table 3.4 Accuracy obtained for n-gram karakas using back-off and kNN.

such as combinations of unigrams. From this, there will be no problem of time complexity. There will be an advantage of morphs during back-off. Also, utterance level contextual information is captured using kNN. Due to the above reasons, from the results in table 3.4 we can see that the accuracy has risen to 73.34%.

When karaka based dependency method is compared with the position specific n-grams methods and also combinations of unigrams, we can surely see the increase in accuracy, which proves that our method performs better.

Error Analysis

This model is unable to recognize the correct DA tag...

- For the utterances which depend on the context.
- For the utterances which depend on the speaker, as the co-reference resolution is not implemented.
- For the utterances that has the words that are not present in the training data, the karaka dependencies and back-off models won't help in determining the tag of the utterance.

Chapter 4

A Contextual Dialog Act Tagging for Telugu

4.1 Problem Definition

The previous chapter describes DA tagging of Telugu using karaka dependencies to obtain n-grams. Dependency parsed data is converted to n_gram_karaka format and Katz's back-off is applied as language modeling at an intra utterance level. After that, memory based learning algorithm viz. KNN is applied at the utterance level. The task oriented dialogs have conversational flow which lead to accomplishment of the required task. There might be a chance that previous data has its impact on the current utterance. In such a case the context of the previous utterances must be maintained to predict the tag of the current utterance. The context handling can be done with the help of forward backward algorithms. We have used HMM as one of the forward backward algorithms to capture that flow. Because HMM works on the principle of the Markov model, i.e. the present state of the sentence is predicted by observing its previous states.

4.2 Our Approach

We extend the previous approach as

1. Instead of considering naive language model (LM), we consider a combination of language models.

2. The context of previous utterances is considered for predicting the DA tag of the current utterance.

The language models considered are back-off [7] and interpolation [14].

We know that a task oriented dialog is a continuous flow of data. It can be broadly divided into 4 categories. First comes, conversation initiators. For example, *hello, hi, how are you?* etc. Followed by the request for start of the task, like, *I want to take this book.* Then comes the

accomplishment of the task, like, *Now, you can take this book*, finally follows the conversation terminator, like, *thank you*. Due to this dialog flow, we can say that each utterance in a conversation depends on its previous utterances. The context of the previous utterances must be maintained to predict the tag of the current utterance.

In the previous chapter, dependency parsed data is extracted manually. In this method, we automate the extraction of karaka dependencies by employing a constraint based approach.

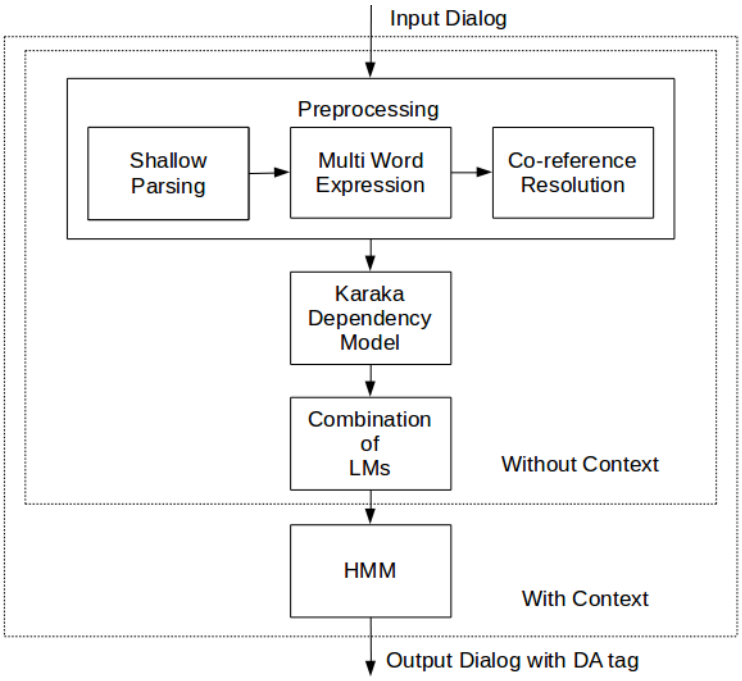

Figure 4.1 Figure showing karaka dependency based approach for DA tagging using combination of LMs and HMM.

4.2.1 Preprocessing

4.2.1.1 Shallow Parsing

Before extracting a list of modifier_karaka_modified chunks for an utterance, the preprocessing of the utterance is a must. This includes shallow parsing, handling multi word expressions and co-reference resolution.

Shallow parsing by [21] is done to establish intra chunk dependencies. Here, the Telugu utterances are given to the shallow parser (SP). The shallow parser has mainly 3 modules. They are morph module, POS module and chunker module. The morph module extracts the morph and suffix information. Then the POS module annotates each word in an utterance with its respective POS tags. Finally the chunking is done on which karaka dependencies are established.

The input given to the shallow parser and its modified output is shown in example 1.a.

Example 1.a:
Shallow Parser Input:
(Student) విద్యార్థి : నాకు ఈ పుస్తకము జారీ చేయండి.
Shallow Parser Output (in wx-notation):
(Student)vixyArWi : [nenu_ki#PRP] [I#DEM puswakaM#NN] [jArI#NN]
[ceVyyi_aMdi#VM]

4.2.1.2 Multi Word Expression Identification

After shallow parsing, by observing the output data, it is clear that there are certain words which are chunked into separate units by the shallow parser but a single meaning is obtained by joining them. These are known as multi word expressions. The problem of multi word expressions is solved by establishing constraints and grouping those multi word expressions as a single unit.

Example 2.a shows the addition of **<mwe>** to form a multi word expression.

Example 2.a:
(Input given) Shallow Parser Output (in wx-notation) is:
(Student)vixyArWi : [nenu_ki#PRP] [I#DEM puswakaM#NN] [jArI#NN]
[ceVyyi_aMdi#VM]
(Output obtained) multi word expressions are grouped:
(Student) vixyArWi : [nenu_ki#PRP] [I#DEM puswakaM#NN]
[jArI#NN]**<mwe>**[ceVyyi_aMdi#VM]

4.2.1.3 Co-reference Resolution

The third step is the co-reference resolution. In the library domain, we come across utterances such as “give me the book”. When it is said by a librarian, the utterance’s intent might be to ’return the book’, but when it is said by a student the utterance’s intent might be ’to issue the book’. If the personal pronoun “me” is resolved with the respective anaphora i.e. either the librarian or a student, this further leads to better intention recognition of the utterance. The co-reference resolution is purely a set or rules. By observing the data, rules are written to resolve the personal pronouns.

Examples of anaphora resolution is given in 3.a.

Example 3.a:
(Input given) multi word expressions are grouped:
(Student) vixyArWi : [nenu_ki#PRP] [I#DEM puswakaM#NN]
[jArI#NN]**<mwe>**[ceVyyi_aMdi#VM]

(Output obtained) co-reference resolution:
(Student) vixyArWi : [nenu_ki**@vixyArWi**#PRP] [I#DEM puswakaM#NN]
[jArI#NN]<cp>[ceVyyi_aMdi#VM]

In the scenario, "nenu" and "nenu_ki" is refering to the 1^{st} person i.e. "I" in English and "mimmalni" is referring to the 2^{nd} person i.e. "you" in English. When the constraint based co-reference resolution module is applied, the 1^{st} person and the 2^{nd} person are resolved w.r.t their anaphora.

4.2.2 N-grams with Karaka Dependencies Extraction

After preprocessing, we developed a constraint based karaka dependency module. By observing the ASKLIB corpus, rules are written by considering chunk, multi word expression information and parts of speech tags obtained from shallow parser. The constraint based karaka dependency module is totally dependent on the data. As the DA tag depends on the action of the utterance, where action is generally the verb in the sentence, we have written rules to extract karakas only for those sentences where the modifier is the verb. i.e. only k1-k7 and vmod karaka relationships are considered. These rules act as syntactic cues and help in predicting the modifier-modified relationships between the chunks in the preprocessed utterance. Here, the preprocessed utterance is given as input to the constraint based karaka dependency module and the output obtained will be a list of modifier_karaka_modified chunks for the corresponding utterance.

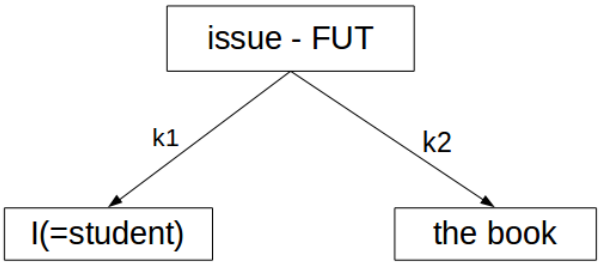

Figure 4.2 Figure showing karaka dependencies for an example utterance *"I will issue the book"*.

In figure 4.2, "k1" is the karta, the one who carries out the action and "k2" is the karma or the object of the verb.

Example 4.a shows the extraction of relating the clusters using karaka dependency relationships.

Example 4.a
Input given to extract karaka dependencies:
[nenu_ki@vixyArWi#PRP] [I#DEM puswakaM#NN] [jArI#NN]<cp>[ceVyyi_aMdi#VM]

Output obtained:
[jArI#NN]<cp>[ceVyyi_aMdi#VM]-**k1**-[nenu_ki@vixyArWi#PRP]
[jArI#NN]<cp>[ceVyyi_aMdi#VM]-**k2**-[I#DEM puswakaM#NN]

4.2.3 Language Modeling Combination

As n_gram_karakas can be treated likewise to n-grams, n-gram language modeling methods can be applied. The statistical n-gram LM technique used here is the language modeling combination of linear interpolation and back-off. We modified the simple linear interpolation method. Instead of obtaining the lambda values from the held out data, we replaced the lambda values using back-off weights. LM combination of interpolation and back-off is used because, it has been proven that the combination of LMs decreases the error rate when compared to the naive LM methods as the negative effects of interpolation and back-off cancel out. Another advantage of using back-off weights is that we can assign higher priority to the n_gram_karakas when compared to the non dependencies. The basic constraint of all the weights summing up to one is kept same. Along with LM combination, naive smoothing method is also applied.

Telugu language has a rich and productive morphology. The number of word forms might be infinite in such a case. The use of just n_gram_karakas is not enough while LM method is performed. Hence we go for the morphed form of n_gram_karakas, they are known as morphed n_gram_karakas. While identifying modifier-modified relationships, it is not necessary that all words in the utterance satisfy the modifier-modified criteria. In such a case, these words should not be left alone. Hence we reduce the n_gram_karakas to non karaka dependent chunks such as $chunk_1$ and $chunk_2$ as shown in equation 4.3. Similarly, morphed n_gram_karakas are further reduced to two morph chunks such as $morph\_chunk_1$ and $morph\_chunk_2$ as shown in equation 4.7.

Finally, in the interpolation method with back-off, the n_gram_karakas, the morphed n_gram_karakas, the non karaka dependent chunks and the morphed non karaka dependent chunks are interpolated by considering the back-off weights.

The equations that are used for calculating the probabilities over the DA tagset T using a combination of LMs are given below:

$$P_1 = \frac{C(n\_gram\_karaka, T)}{C(n\_gram\_karaka)} \tag{4.1}$$

$$P_2 = \frac{C(morph\_n\_gram\_karaka, T)}{C(morph\_n\_gram\_karaka)} \tag{4.2}$$

$$n\_gram\_karaka = chunk_1 + karaka + chunk_2 \tag{4.3}$$

$$P_{3_1} = \frac{C(chunk_1, T)}{C(chunk_1)} \tag{4.4}$$

$$P_{3_2} = \frac{C(chunk_2, T)}{C(chunk_2)} \tag{4.5}$$

$$P_3 = P_{3_1} + P_{3_2} \tag{4.6}$$

$$morph\_n\_gram\_karaka = morph\_chunk_1 + karaka + morph\_chunk_2 \tag{4.7}$$

$$P_{4_1} = \frac{C(morph\_chunk_1, T)}{C(morph\_chunk_1)} \tag{4.8}$$

$$P_{4_2} = \frac{C(morph\_chunk_2, T)}{C(morph\_chunk_2)} \tag{4.9}$$

$$P_4 = P_{4_1} + P_{4_2} \tag{4.10}$$

$$P_{LI} = \sum_{i=1}^{4} \alpha_i * P_i \tag{4.11}$$

$$where\ \alpha_{i+1} = 0.4 * \alpha_i\ and\ \sum_{i=1}^{4} \alpha_i = 1 \tag{4.12}$$

Where C refers to the count, T is the DA tagset used, P is the probability calculated for i^{th} tag T_i for the given chunk. Equation 4.11 gives the language model combination of interpolation with back-off. The condition given in equation 4.12 are taken from the stupid back-off by [7]. Stupid back-off is considered because of its simplicity and linearity in back-off weights which provides a good integration into the deleted linear interpolation method. The language model is applicable only at utterance level. For a better DA tagging, the context of the utterances must be considered. Hence, for context level we have used Hidden Markov Model (HMM).

4.2.4 Adding HMM for Context Handling

Now coming to the context preserving, we know that the task oriented dialogs have conversational flow which lead to accomplishment of the required task. An example of the flow of conversation is given in the figure 1.2.

This flow can be captured by using the forward backward algorithms. We have used HMM as one of the forward backward algorithms to capture that flow. As HMM works on the principle of the Markov model, i.e. the present state of the sentence is predicted by observing its previous states. We modified it as, *the present DA tag of the utterance is predicted by observing the previous 'k' DA tags in the dialog.* The Viterbi algorithm is a dynamic programming algorithm

DA tagging without context handling	**DA tag**
Student: I would like to issue this book Librarian: This book has already been issued to you. Would you like to reissue the book?	ISSUE REISSUE_INFO_REQUEST
DA tagging with context handling	**DA tag**
Student: I would like to issue this book Librarian: This book has already been issued to you. Would you like to reissue the book?	REISSUE REISSUE_INFO_REQUEST

Table 4.1 An example which proves that context handling is necessary.

and is the most common decoding algorithm used for HMMs, whether for part-of-speech tagging or for speech recognition. We used this for the DA tagging of the sentences.

Why context handling for ASKLIB corpus? In ASKLIB corpus there might be situations where the implicit meaning of the speaker is different from what the utterance literally conveys i.e. the words in an utterance might point to a particular DA tag based on the LM output but while considering the context they might point to another DA tag. An example is shown in table 4.1.

From the table 4.1 [1], after observing the 2 utterances, one can say that the student wants to reissue the book. On the other hand, if we use only the combination of LMs, the information conveyed by the librarian will be missed as LMs modeled here work at utterance level only. Hence without the context i.e. only at LM level, the DA tag given will be ISSUE. But with the help of context viz. with the librarian's utterance we can say that the appropriate DA tag is REISSUE.

$$\hat{Q} = argmax_Q \ P(Q)P(S|Q) \tag{4.13}$$

$$\hat{Q} = argmax_Q \prod_{i=1}^{n} P(Q_i|Q_{i-1}) \frac{P(Q_i|S_i)}{P(Q_i)} \tag{4.14}$$

Equation 4.13 which is further simplified to equation 4.14. Equation 4.14 calculates the best tag for the given utterance considering its previous context.

Viterbi algorithm consists of the emission matrix and the transition matrix. The emission matrix values are obtained from the LM combination method mentioned above and the transition matrix values are obtained from conversation flow of the training data. The Viterbi algorithm is run two times. Firstly, the dialog is given to the Viterbi algorithm and the output with DA tag and its probabilities are taken. During the second run, the dialog is reversed and given to Viterbi algorithm and the corresponding DA tag for utterance and its probabilities are

[1]The dialogs given here were originally written in Telugu but translated to English for reader's understanding.

taken. Then the two probabilities obtained for each utterance are compared and the best tag which has the greater probability is given to the utterance.

This approach helps in complete coverage of the utterance i.e. this approach not only considers previous tags but indirectly it takes next tags to predict the output of the current utterance.

4.2.5 Experiments and Results

We compared this approach with the various state of art tagging algorithms. The results are shown in table 4.2

The ASKLIB data is divided equally into 4 sets. The concept of leave-one-out validation is applied here. Every time one of the set is used for testing and remaining are combined for training.

Table 4.2 Accuracy obtained by comparing various classification algorithms.

Classification Algorithms	**Result**
Unigrams approach	58.17%
Bigrams approach	54.71%
Trigrams approach	47.79%
n_gram_karakas using Katz's backoff as LM plus kNN	73.34%
Current approach without context (i.e. without HMM)	75.89%
Current approach with context (i.e. with HMM)	78.67%

Firstly, from ASKLIB corpus n-grams of length 1 to 3 are extracted and Bayesian interpretation is used for DA tagging. From the results in table 4.2, it is clear that when the n-gram length increases, there is a decrease in accuracy. The reason is, as Telugu is a free word order language, even though the utterance's intent is the same, there might be change in the position of words in both the training data and the test data. Hence, extracting only fixed order n-grams for DA tagging will not work for free word order languages like Telugu.

Next half of the table 4.2 shows the method of karaka dependencies to extract n-grams. The current method is compared with various classification algorithms such as Katz's backoff and a combination of LMs.

From the results presented in table 4.2 it is clear that this method has shown an improvement in accuracy. The improvement in the accuracy is due to the addition of the following factors.

1. **Efficient processing of data by considering the linguistic features of Telugu language.**

Linguistic features such as POS, morph, chunk marking, anaphora marking and the use of multi word expressions helped in the optimized grouping of word clusters. This in-turn helped in the better establishment of the modifier-modified relationships.

2. **Coreference resolution for speaker recognition.**

   The Coreference resolution of speaker has differentiated the student and the librarian utterances which contributed effective DA tagging of the utterance.

3. **The use of combination of LMs instead of just Katz's back-off.**

   As the combination of the interpolation and back-off was used, high-order n-grams i.e. the n_gram_karakas which were sensitive, and had sparse counts were given high back-off weight. The reason was that n_gram_karakas have more detailed information for accurately predicting DA tag of the sentence.

   The low-order n-grams i.e. the non karaka dependent chunks (both inflected and morphed) having robust counts were given low back-off weight because the contribution of non karaka dependent chunks in predicting DA tag was low.

   Due to this LM combination the negative effects of both the naive interpolation and back-off on the data were reduced.

4. **Consideration of the context of the previous utterance while predicting the DA tag of the current utterance.**

   Context handling and backtracking by Viterbi algorithm helped in preserving the flow of the dialog, which in-turn helped in efficient DA tagging.

Note that the final accuracy is the multi-class accuracy that is obtained by calculating the accuracy of each and every DA tag, then taking the average of the normalized DA tagset accuracy as given in the equation 4.15

$$Accuracy = \sum_{i=1}^{T} \frac{Count\,(T_i)}{Count(T)} Accuracy(T_i) \tag{4.15}$$

Where T stands for the tagset, T_i is the i^{th} tag in the tagset, $Accuracy(T_i)$ gives the accuracy of the single tag T_i. $Accuracy(T_i)$ is calculated by comparing the tags obtained from the model w.r.t the manual tagged output.

Error Analysis

This model is unable to recognize the correct DA tag...

- If the test utterance has words that are unknown to the training data. This is because of the following reasons.
  - The karaka dependency extractor is purely data dependent. It cannot identify karaka relationships for the new words.
  - As there is no semantic model or word net integration is present, even the words of same meaning i.e. synonyms cannot be identified this results in poor tagging.
  - The language modeling technique gives very least scores for unknown words, which decreases the probability for good better tagging.
  - There might be situations where the utterance meaning varies only with one word but that word is not present in the training data, this situation might result to erroneous output too.

Chapter *5*

Applications

There are many applications of automatic dialogue acts detection. We mention here only the most important ones: Dialogue Systems, Machine Translation, Automatic Speech Recognition (ASR) and Animation of talking head.

Dialog Systems

In dialogue systems, the ability to assign user input with a functional tag which represents the communicative intentions behind each utterance — the utterance's dialogue act — is acknowledged to be a useful first step in dialogue processing. Such tagging can assist the semantic interpretation of user utterances, and can help an automated system in producing an appropriate response.

Machine Translation(MT)

Recent approaches to Statistical Machine Translation (SMT) have relied on improving translation quality with the use of phrase translation. However, in many dialog based machine translation scenarios, vital information beyond what is robustly captured by words and phrases is carried out by the communicative act. Incorporating dialog act tags in speech translation is motivated by the fact that it is important to capture and convey not only what is being communicated (the words) but how something is being communicated (the context). Augmenting current statistical translation frameworks with dialog acts can potentially improve translation quality and facilitate successful cross-lingual interactions in terms of improved information transfer.

Another problem in the source language of MT is that a surface utterance may represent several ambiguous meanings depending on the context. That means such utterances can be translated into many different ways depending on the context. Interpreting this kind of utterances often requires the analysis of intent in contexts. Here, dialogue acts can be useful to

choose the best solution when several translations are available. Recognizing dialogue acts may bring out relevant cues to choose between alternative translations, as the adequate syntactic structure may depend on the user intention.

Automatic Speech Recognition

Automatic speech recognition (ASR) can be defined as the independent, computer - driven transcription of spoken language into readable text. In a nutshell, ASR is a technology that allows a computer to identify the words that a person speaks into a microphone or telephone and converts it to a written text.

The ultimate goal of ASR research is to allow a computer to recognize all words that are intelligibly spoken by any person, independent of vocabulary size, noise, speaker characteristics or accent with 100% accuracy.

In ASR, the search space is all the possible sentences in the language and the solution we are looking for is the most likely sentence given some observations and constraints. Observations give us information about the utterance being recognized and constraints express our prior knowledge. Using dialog acts, we can classify the spoken utterances into types. In this approach an utterance is given a DA tag. This acts as the constraint in the speech recognition. Using training data (DA tagged utterance) and a language model we can statistically predict the best written text to a spoken utterance.

Animation of talking head

A talking head is a model of the human head that reproduces the speech of a speaker in real time. It may also render facial expressions that are relevant to the current state of the discourse. Exploiting DA recognition in this context might make the animation more natural, for example by raising the eyebrows when a question is asked. Another easier option is to show this complementary information with symbols and colors near the head.

Chapter *6*

Conclusions and Future Work

6.1 Conclusion

We present a DA tagging method which can tag the morphologically rich, free word order languages like Telugu. Before that, as there was no available dialog corpus for Telugu. We have developed ASKLIB corpus which focuses on interactions between a student and a librarian. Our proposed method uses karaka dependencies in the modifier_karaka_modified format. The use of modifier-modified karaka dependency model converts the free word order to a fixed grammatical format. By careful observation, this format is similar to the n-grams in English. Hence statistical n-gram language modeling is used.

The baseline n-gram algorithms such as unigram, bigrams, combination of unigram methods are applied to the Telugu corpus and its behavior is noted.

Then, our proposed model i.e. "A Semi Supervised Dialog Act Tagging for Telugu" considers the n-gram karakas with back-off language modeling technique at intra utterance level in combination with Memory Based Learning (MBL) such as k Nearest Neighbor (kNN) at utterance level. This method for DA tagging provided an accuracy of 73.34% when compared to the baseline methods. The results given in table 3.4 prove that this method performs better when compared to the other methodologies and is one of the best suited for the DA tagging in Telugu task oriented dialogs.

It is observed that the above method works only at utterance level and the linguistic features such as anaphora resolution and conjunct identification enhance the probability of assigning the correct DA tag. Hence in the next model, the aforementioned linguistic features are included. The LM method considered in the previous approach is a baseline method which can be improved by a combination of LM's i.e. A combination of interpolation LM and back-off LM. It is known that, dialog consists of flow of data. Hence before predicting the utterance's DA tag we have to consider its context. The problem of contextual handling is resolved with the help of HMM using Viterbi algorithm.

6.2 Future Work

The scope of future directions for this work is as follows,

- In this work, we only focused on n-gram dialog act tagging methods. Apart from these, there are several statistical methods developed tagging techniques such as graph based models etc. We will try to observe the behavior of other tagging techniques on our corpus.
- The combination of language models applied is a simple framework of linear interpolation and stupid back-off. If we substitute Katz's back-off instead of stupid back-off LM technique, we can increase the efficiency of the combination of language models.
- DA tagging can be explored in machine translation system. By recognizing the intent of the utterance, one can predict the actual target translation given its DA tag. An example showing the output of an MT system using DA tagging for intent recognition is given in table 6.1

Source Language (Telugu)	పిచుక మీద బ్రహ్మాస్త్రం.
English Translation (Literal)	Using Brahmastram on a sparrow.
English Translation with Intent Recognition	Using excessive force on a weak opponent.

Table 6.1 Use of Intention Recognition in Machine Translation.

Related Publications

- **Suman Dowlagar** & Radhika Mamidi. *A Semi Supervised Dialog Act Tagging for Telugu.* In Proceedings of 12th International Conference on Natural Language Processing(ICON), Kerala, India, December 11-14, 2015.
- **Suman Dowlagar** & Radhika Mamidi. *A Karaka Dependency based Dialog Act Tagging for Telugu using Combination of LMs and HMM.* In Computational Linguistics and Intelligent Text Processing: 16th International Conference, CICLing 2015, Konya, Turkey, April 3-9, 2016. Springer International Publishing, Cham, 2016.

Bibliography

[1] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. Towards a generic dialogue shell. *Natural Language Engineering*, 6(3):1–16, 2000.

[2] J. F. Allen, D. K. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. Toward conversational human-computer interaction. *AI magazine*, 22(4):27, 2001.

[3] J. L. Austin. *How to do things with words*. Oxford university press, 1975.

[4] R. Begum, S. Husain, A. Dhwaj, D. M. Sharma, L. Bai, and R. Sangal. Dependency annotation scheme for indian languages. In *IJCNLP*, pages 721–726. Citeseer, 2008.

[5] A. Bharati, V. Chaitanya, R. Sangal, and K. Ramakrishnamacharyulu. *Natural language processing: a Paninian perspective*. Prentice-Hall of India New Delhi, 1995.

[6] A. Bharati, R. Sangal, D. M. Sharma, and L. Bai. Anncorra: Annotating corpora guidelines for pos and chunk annotation for indian languages. *LTRC-TR31*, 2006.

[7] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. Large language models in machine translation. In *In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Citeseer, 2007.

[8] M. G. Core and J. Allen. Coding dialogs with the damsl annotation scheme. In *AAAI fall symposium on communicative action in humans and machines*, pages 28–35. Boston, MA, 1997.

[9] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[10] S. Dowlagar and R. Mamidi. A semi supervised dialog act tagging for telugu. ICON 2015 : 12th International Conference on Natural Language Processing, 2015.

[11] P. N. Garner, S. R. Browning, R. K. Moore, and M. J. Russell. A theory of word frequencies and its application to dialogue move recognition. In *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, volume 3, pages 1880–1883. IEEE, 1996.

[12] E. Ivanovic. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, pages 79–84. Association for Computational Linguistics, 2005.

[13] D. Jurafsky and J. H. Martin. *Speech & language processing*. Pearson Education India, 2000.

[14] D. Jurafsky and J. H. Martin. *Speech & language processing*. Pearson Education India, 2000.

[15] T. Klüwer, H. Uszkoreit, and F. Xu. Using syntactic and semantic based relations for dialogue act recognition. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 570–578. Association for Computational Linguistics, 2010.

[16] P. Král and C. Cerisara. Dialogue act recognition approaches. *Computing and Informatics*, 29(2):227–250, 2012.

[17] P. Král and C. Cerisara. Automatic dialogue act recognition with syntactic features. *Language Resources and Evaluation*, 48(3):419–441, 2014.

[18] P. Liu, Q. Hu, J. Dang, D. Jin, and J. Cao. Dialog act classification in chinese spoken language. In *Machine Learning and Cybernetics (ICMLC), 2013 International Conference on*, volume 2, pages 516–521. IEEE, 2013.

[19] M. M. Louwerse and S. A. Crossley. Dialog act classification using n-gram algorithms. In *FLAIRS Conference*, pages 758–763, 2006.

[20] K. P. Mohanan. Grammatical relations and clause structure in malayalam. *The mental representation of grammatical relations*, 504:589, 1982.

[21] A. PVS and G. Karthik. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. *Shallow Parsing for South Asian Languages*, 21, 2007.

[22] N. Reithinger and M. Klesen. Dialogue act classification using language models. In *EuroSpeech*. Citeseer, 1997.

[23] M. Rotaru. Dialog act tagging using memory-based learning. *Term project, University of Pittsburgh*, pages 255–276, 2002.

[24] J. R. Searle. A classification of illocutionary acts. *Language in society*, 5(01):1–23, 1976.

[25] A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.

[26] N. Webb and M. Ferguson. Automatic extraction of cue phrases for cross-corpus dialogue act classification. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1310–1317. Association for Computational Linguistics, 2010.

[27] Wikipedia. Katz's back-off model — wikipedia, the free encyclopedia, 2015. [Online; accessed 29-February-2016].

[28] Wikipedia. Dialog system — wikipedia, the free encyclopedia, 2016. [Online; accessed 19-April-2016].