Naïve Bayes for Function Tagging in Myanmar Language

Win Win Thant
University of Computer Studies Yangon, Myanmar
winwinthant@gmail.com

Abstract

This paper proposes an approach to annotate function tags for unparsed text. We address the question of whether data-driven function tag assignment method can be applied to Myanmar language. We assign function tags directly basing on lexical information, which is easily scalable for languages that lack sufficient parsing resources or have inherent linguistic challenges for parsing. We investigate a supervised sequence learning method to automatically recognize function tags. In order to demonstrate the effectiveness and versatility of our method, we investigate function tag assignment for unparsed text by applying Naïve Bayesian theory. Our approach to functional analysis is to classify, so far as possible, all the processes and states which languages must describe, and to identify the functional elements which are needed for each one to construct a meaningful sentence.

1. Introduction

The natural language processing community is in the strong position of having many available approaches to solving some of its most fundamental problems [5]. The corpus-based statistical parsing community as many fast and accurate automated parsing system. A word can appear in a sentence for two reasons: because it serves a syntactic function, or because it provides semantic content. Words that play different roles are treated differently in human language processing: function and content words produce different patterns of brain activity, and have different developmental trends.

Syntactic information is an important processing step to many language processing applications such as Anaphora Resolution, Machine Translation, and Question Answering. Syntactic parsing in its most general definition may be viewed as discovering the underlying syntactic structure of a sentence. The specificities include the types of elements and relations that are retrieved by the parsing process and the way in which they are represented.

Myanmar is verb final language. It is also a variable word order language. The free word order feature of Myanmar makes parsing a challenging task. Syntactic analysis is a part of the Myanmar to English machine translation project. If high quality translation is to be achieved, language understanding is a necessity. One problem in Myanmar language processing is the lack of grammatical regularity in the language. This leads to very complex Myanmar grammar in order to obtain satisfactory results, which in term increases the complexity in the parsing process, it is desired that simple grammar is to be used. However, this will cause ambiguities in the parse result. Parsers operate at word-level with the assumption that input sentences are pre-segmented, tagged and chunked. We define function tags and sentence structure of Myanmar language.

Syntactic tags are useful for any application trying to follow the thread of the text –they fine the ‘who does what’ of each clause, which can be useful to gain information about the situation or to learn more about the behavior of words in the sentence [3]. For instance, any algorithm that needs to know the subject of a sentence would benefit from actually having that subject, rather than relying on an easy but stupid algorithm like “first noun phrase” or “verb phrase's left sibling”.

A small corpus annotated manually serves as training data because the large scale Myanmar Corpus is unavailable at present. The relations of the part-of-speech tags are obtained from training data. Since the large-scale annotated corpora, such as Penn Treebank, have been built in English, statistical knowledge extracted from them has been shown to be more and more crucial for natural language disambiguation [4]. As a distinctive language, Myanmar has many characteristics different from English. The use of statistical information efficiently in Myanmar language is still a virgin land waiting to explore.

We chose Naïve Bayesian for its simplicity and user-friendliness, respectively. Naive-Bayesian classifier make strong assumptions about how the data is generated, and use a probabilistic model that reflects these assumptions [9]. They use a collection of labeled training examples to estimate the parameters of the generative model. Classification of new examples is performed with Bayes’ rule by selecting the class that is most likely to have generated the example. The Naive Bayesian classifier assumes that all attributes of the examples are independent of each other given the context of the class. This is the so-called “Naïve Bayes assumption”. This assumption is wrong in many real world tasks, yet Naive Bayes classifiers often perform very well. This paradox is explained by the fact that classification estimation is only a function of the sign (in binary cases) of the function estimation; the function approximation can still be poor while classification accuracy remains high (Fried-man 1997). Because of the independence assumption, the parameters for each attribute can be learned separately, and this greatly simplifies learning, especially when the number of attributes is large (McCallum & Nigam 1998).

The rest of the paper is organized as in the followings. Next, in the Related Work section, we analyze previous efforts related to the task of function tags assignment. Section 3 includes the proposed function tags of Myanmar language. Section 4 describes about the model. Naïve Bayesian classifier is presented in section 5. Section 6 explains about features. Training data is described in section 7. Finally the conclusion of the paper is presented.

2. Related Work

Previous work to address the task of function tags assignment is presented in (Blaheta & Johnson 2000). They use a statistical algorithm based on a set of features grouped in trees, rather than chains. The advantage is that features can better contribute to overall performance for cases when several features are sparse. When such features are conditioned in a chain model the sparseness of a feature can have a dilution effect of an ulterior (conditioned) one.

Don Blaheta presented a system that utilizes a maximum-entropy inspired algorithmic framework along with a number of commonly used features (label, syntactic head, etc) to predict function tags with relatively high accuracy. He then presented two other algorithmic frameworks and a number of new features to be used with them. He proposed to use these expanded systems to improve performance on the function tagging task, and having done so, analyze the results to determine which features were most helpful in the task as a whole and in its various subtasks [2].

Previous to that, Collins (1997) approached the problem of distinguishing adjuncts from complements. The motivation in that case was to improve parser performance by guessing complement status during the parse, the statistics were a bit cleaner [7]. That paper defines constituents as complements or adjuncts based on a combination of label and function tag information. This Boolean condition is then used to train the improved parser. The system uses a generative model to evaluate parse quality, and the complement information is used as follows. After generating the head-containing child of a constituent (conditioned on the constituent and its head word), left and right subset frames are chosen conditional on that head-containing child (and the previously-used conditioning info). A subset frame is simply a bag of labels that are subcategorized by, i.e. complement to, the parent. Given the subset frame, then, and all the previous conditioning information, the actual labels of the other children are generated. Collins does not report his results on the complement tagging [8].

Also, there were previous attempts to enrich the output of syntactic parsers with additional information available in Penn Treebank such as dependency information (Johnson 2002; Jijkoun & De Rijke 2004).We present in this paper a common framework to address the problem of function tagging and report how Naïve Bayes performs within this framework. The framework is defined by a common underlying model and a common set of preprocessing steps. The model and the preprocessing steps are described later

3. Myanmar Function Tags

Function tags, such as subject, object, time, location, etc. are conceptually appealing by encoding an event in the format of “who did what to whom, where, when”, which provides useful semantic information of the sentences. When dealing with the task of function tag assignment, one basic question that must be addressed is what features can be extracted in practice for distinguishing different function tag types. Our proposal is to classify function types directly from lexical features like words and their POS tags and the surface sentence information like the word position. The task of function tagging, the problem addressed in this paper, is to add function tags to words in a sentence. In the proposed system, we identity the function tags based on preposition. There are 18 tags.

Function tag for verb chunk

1. Active ACTIVE

    သူ **ပြေးသည်။**

    He **runs.**

Function tags for other chunks

1. Subject SUBJ

    သူ သွားသည်။

    He goes.

2. Direct Object OBJ

သူသည်**ကော်ဖီကို**သောက်သည်။
He drinks **coffee**.

3. Indirect Object I-OBJ
သူသည် **မလှအား** စာအုပ်ကို ပေးသည်။
He *gives* ***Ma Hla*** the book.

4. Place PLA
သူ ကျောင်း **သို့** သွားသည်။
He goes **to** school.

5. Time TIM
သူသည် နံနက်(၆)နာရီ **တွင်** အိပ်ရာမှ ထသည်။
He gets up from bed **at** 6 o’clock in the morning.

6. Extract EXT
ကျောင်းသားများ**အနက်**မောင်ဘသည် အတော်ဆုံး ဖြစ်သည်။
Mg Ba is the cleverest boy **among** the students.

7. Refer REF
ကျွန်တော့်အမေ**အတွက်** ကိတ်မုန့် ဝယ်လာသည်။
I buy a cake **for** my mother.

8. Simile SIM
သူမသည် မင်းသမီး**လို** ဝတ်စား၏။
She wears the dress **as** an actress.

9. Compare COMP
သူသည် သူ့ဦးလေး**နှင့်**အတူ နေသည်။
He lives **with** his uncle.

10. Own OWN
သူသည် **သူ၏အမေကို** ချစ်သည်။
He loves **his mother**.

11. Predicative Complement AD-A
သူမ **လှသည်။**
She *is* **beautiful**.

12. Subject Complement PCOMPL-S
သူမသည် **ဆရာမဖြစ်သည်။**
She *is* **a teacher**.

13. Object Complement PCOMPL-O
မောင်လှသည် ရွှေကို **လက်စွပ်** လုပ်သည်။
Mg Hla makes the gold **a ring**.

14. Use USE
သူသည်ခွေးကို **တုတ်ဖြင့်** ရိုက်သည်။
He hits the dog **with a stick**.

15. Cause CAU
လယ်ကွင်းများသည် **မုန်တိုင်းကြောင့်** ပျက်စီးသည်။
The fields are destroyed **because of the storm**.

16. Aim AIM
သူသည် **သူ့အမေအတွက်** ကိတ်မုန့် ဝယ်သည်။
He buys the cake **for his mother**.

17. Conjunction CC
လှလှ **နှင့်** မြမြ သည် သူငယ်ချင်းများ ဖြစ်ကြသည်။
Hla Hla **and** Mya Mya are friends.

The task of function tagging is to add extra labels, called function tags, to the chunk. Let us pick as an illustrative example:

“U Hla is a teacher”
ဦးလှသည်ဆရာဖြစ်သည်။

NC[ဦးလှ]#PPC[သည်]#NC[ဆရာ]#VC[ဖြစ်]#SFC [သည်]။

Each word in the sentence has a corresponding chunk. For instance, the word ဦးလှ has NC as its chunk (NC indicates a noun chunk). All the other words will be labeled with a syntactic tag that marks the chunk corresponding to the word, such as NC (noun chunk), VC (verb chunk), PPC (preposition chunk) and SFC (sentence final chunk).

Technically, the task of function tags assignment is to generate a sentence that has function tags attached to certain words. Examples are:

SUBJ[ဦးလှသည်]#PCOMPL-S[ဆရာ]# ACTIVE[ဖြစ်သည်]။

SUBJ[လူထုက]#PCOMPL-O[ကောင်စီဝင်အဖြစ်]# OBJ[သူ့☐ကို] #ACTIVE[ရွေးသည်] ။

OBJ[စာအုပ်ကို]#PLA[ဆရာ့ဆီ]# ACTIVE[ယူလာပါ] ။

OBJ[ခွေးကို]#USE[တုတ်ဖြင့်]#SUBJ[သူ]#ACTIVE [ရိုက်သည်]။

4. Model

We model the problem of assigning function tags as a classification problem. Classifiers are programs that assign a class from a predefined set of classes to an instance based on the values of attributes used to describe the instance. We define a set of linguistically motivated features based on which we characterize the instances. We automatically generate instances from our tagged corpus and then use them to derive Naive Bayesian classifier as solutions to the function tags assignment problem [9].

The set of classes we used in our model corresponds to the set of functional tags we proposed. For instance, a chunk can have a label such as NC-SUBJ-OBJ to indicate a noun chunk (NC) that has attached to it two function tags, SUBJ (subject) and OBJ (object). Those tags were necessary to distinguish words or phrases that belong to one syntactic category and are used for some other function or when it plays a role that is not easily identified without special annotation.

5. Naïve Bayesian Classifier

Before one can build naive Bayesian based classifier, one needs to collect training data [9]. The training data is a set of problem instances. Each instance consists of values for each of the defined features of the underlying model and the corresponding class, i.e. function tag in our case. A small corpus annotated manually serves as training data because the large scale Myanmar Corpus is unavailable at present. The relations of the part-of-speech tags are obtained from training data. The development of a naive Bayesian classifier involves learning how much each parser should be trusted for the decisions it makes. In probability estimation for Naive Bayesian classifiers, namely that the attribute values are conditionally independent when the target value is given. Naive Bayesian classifiers are well-matched to function tagging problem.

The Naïve Bayesian classifier is a term in Bayesian statistics dealing with a simple probabilistic classifier based on applying Bayes' theorem with strong (naïve) independence assumptions. It assumes independence among input features. Therefore, given an input vector, its target class can be found by choosing the one with the highest posterior probability.

In simple terms, a Naïve Bayesian classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other features. This method is important for many reasons. It is very easy to construct, not needing any complicated iterative parameter estimation schemes. This means it may be readily applied to huge data sets. It is easy to interpret, so users unskilled in classifier technology can understand why it is making the classification makes [6].

They often face the data sparseness problem and do not generalize well. Bayesian reasoning is applied to decision making and inferential statistics that deals with probability inference. It is using the knowledge of prior events to predict future events. Parameter estimation for Naive Bayesian models uses the method of maximum likelihood.

The probability model for a classifier is a conditional model.

$$P(C/X) = P(X/C).P(C) / P(X)$$

P(X) is constant for all classes

P(C)=relative frequency of class C samples

C such that P(C/X) is maximum=such that P(X/C).P(C) is maximum

$$P(x_1 \ldots x_k /C) = P(x_1/C) * \ldots * P(x_k /C)$$

6. Features

We have found it useful to define our statistical model in terms of features. A feature, in this context, is lexical item. Features can be fairly simple. When using a number of known features to guess an unknown one, the usual procedure is to calculate the value of each feature, and then essentially look up the empirically most probable value for the feature to be guessed based on those known values [1]. We have the function tags and the miscellaneous tags. These are characterized by much more semantic information, and the relationships between lexical items are very important, making sparse data a real problem.

7. Training Data

In virtually all empirical NLP work, the training set is going to encompass the vast majority of the data. As such, it is usually impractical for a human (or even a whole lab of humans) to sit down and revise the training [2]. Purely on grounds of practicality, though, it would be difficult to effect significant correction on a training set of any significant size. Practicality aside, correcting the training set is a bad idea anyway. After expending an enormous effort to perfect one training set, the net result is just one correct training set. While it might make certain things easier and probably will improve the results of most algorithms, those improved results will not be valid for those same algorithms trained on other, non-perfect data; the vast majority of corpora will still be noisy. If a user of an algorithm, e.g. an application developer, chooses to perfect a training set to improve the results, that would be helpful, but it is important that researchers report results that are likely to be applicable more generally, to more than one training set. Furthermore, robustness to errors in the training, via smoothing or some other mechanism, also make an algorithm robust to sparse data thus eliminating all errors in the training ought not to have as much of an effect on a strong algorithm.

8. Conclusion

In this paper, we proposed 18 function tags for Myanmar language and used Naïve Bayesian technique for the task of assigning function tags. Function tags have in the past not been very well studied or exploited. Because of the lack of prior research on this task, we are unable to compare our results to those of other researchers; but the results do seem promising. One of the weaknesses of the lexical features is sparse data.

9. References

[1] Blaheta, D., and Johnson, M. 2000. Assigning function tags to parsed text. In Proceedings of the 1st Annual Meeting of the North American Chapter of the Association forComputational Linguistics, 234–240.

[2]Blaheta, D. 2003. Function tagging. Ph.D. Dissertation, Brown University. Advisor-Eugene Charniak.

[3] Eugene Charniak. 1999. A maximum-entropy inspired parser. Technical Report CS-99-12, Brown University, August.

[4] Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In Proceedings of the Fourteenth National Conference on Artificial Intelligence, pages 598-603, Menlo Park.

[5] John C. Henderson and Eric Brill. Exploiting Diversity in Natural Language Processing:Combining Parsers

[6] Leon Versteegen(1999) “The Simple Bayesian Classifier as a Classification Algorithm”

[7] Michael Collins. 1997. Three generative, lexicalized models for statistical parsing. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, pages 16-23.

[8] Michael Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In Proceedings of ACL-96, pp. 184–191.

[9] Mihai Lintean and Vasile Rus . Naive Bayes And Decision Trees For Function Tagging