Word-Sense Disambiguation of Sinhala Language with Unsupervised Learning

Chandrajith Ashuboda Marasinghe, Susantha Herath*, Ajantha Herath**
The University of Aizu, Aizu-wakamatsu, Japan
*St.Cloud State University, St. Cloud. MN 56301, USA
**University of Dubuque, IA 52001, USA
d8032103@u-aizu.ac.jp, sherath@stcloudstate.edu, aherath@dbq.edu

Abstract--Resolving ambiguity requires little conscious effort in human communications. To make decisions about the intended sense of a word we use our broad understanding of the language and the real-world knowledge. Disambiguation in translations is the selection of the intended sense from a known finite set of possible meanings of an ambiguous word. This choice is based upon a probabilistic model that tells which member of the set of possible meanings is the most appropriate in the context. In this paper we discuss computational methods used to resolve ambiguities in natural language understanding. Two statistical approaches, EM algorithm and Gibbs Sampling algorithm, were applied to solve the ambiguity problem in Sinhala language processing. In these algorithms, context is defined in a very limited way and consists of information that can easily be extracted from the sentence in which an ambiguous word occurs. Susantha-Corpus, containing more than 215,000 words from newspaper texts, is used in this experiment in an unsupervised learning environment.

1. INTRODUCTION

Resolving ambiguities is a routine process in our natural communications. In this process, we combine our broad understanding of the language with the real-world knowledge to make decisions about the intended sense of a word. For us, the context in which an ambiguous word occur provides a wealth of information beyond what is contained in the text. Modeling and representing this knowledge in a program is a complicated task. Accessing and making accurate inferences from the knowledge base is much more complex task. Duplicating the human process to resolve ambiguity[1] by machines is a hard problem to solve.

Disambiguation selects the intended sense of an ambiguous word from a finite set of known meanings. One of the basic problems encountered by any natural language processing system is lexical ambiguity, be it syntactic or semantic. The resolution of syntactic ambiguity of a word in language processing has largely been solved by part-of-speech taggers [1]. Speech taggers predict the syntactic category of words in text with high levels of accuracy. Resolving semantic ambiguity, known as word sense disambiguation, has proved to be more difficult than syntactic disambiguation.

Words often have more than one meaning, sometimes fairly similar and sometimes completely different. The meaning of a word in a particular usage can only be determined by examining its context. This is, in general, a trivial task for the human language processing system, for example consider the following three sentences, written in Sinhala, each with a different sense of the word watha :

1. *sudu watha andi katak dutumi.* [I saw a woman in a white dress.]
2. *ege watha madala sowin barawiya.* [Her face looked sad.]
3. *semage watha gotha eseema ohuge siritaki.* [Asking about others is his practice.]

The listener immediately recognizes that in the first sentence *watha* refers to the dress, and the second to a face, and the third to information. However, the task has proved to be very difficult for computers and some have believed that it would never be solved[3]. Bar-Hillel proclaimed that "sense ambiguity could not be resolved by electronic computer either current or imaginable".

Sinhala is the main language of Sri Lanka spoken over 15 million people. It is derived from old Indo-Aryan Sanskrit through middle Indo-Aryan Prakrit. Pali, the language of the Buddhist scriptures is the best representative of Prakrit. The old Indo-Aryan speeches were spoken in India during 2000-800 B.C. The Sinhala is a member of the Aryan family of languages, which is a member of a still larger family of languages known as Indo-European. There has been a very limited research on computational
linguistic of Sinhala and this would be the first research in word-sense disambiguation of Sinhala [2,7,8,9,10].

Corpus-based methods can be employed to make disambiguation decisions based on probabilistic models learned from naturally occurring text. In these approaches, context is defined in a very limited way and information can easily be extracted from the sentence; no deep understanding of the linguistic structure or real-world underpinnings of a text is required. This results in methods that take advantage of the abundance of text available online.

This paper presents computational methods that resolve ambiguity in natural language text, specially focussed on Sinhala Language. Section 2 of this paper introduces learning from text. Section three discusses unsupervised learning. Section four presents probabilistic models, EM algorithm and Gibbs Sampling. Section five describes implementation and experimental results.

2. LEARNING FROM TEXT

This research focuses on corpus-based approaches to learning probabilistic models that resolve ambiguity of words. These models indicate which sense of an ambiguous word is most probable given the context in which it occurs. In this framework disambiguation consists of classifying an ambiguous word into one of several predetermined senses.

Probabilistic models are learned via supervised and unsupervised. If disambiguated examples are available to serve as training data, supervised learning is most effective[4]. In supervised learning, each instance of the ambiguous word is manually annotated to indicate the most appropriate sense for that usage. If there is no training data available, learning is unsupervised where raw or untagged texts are used. An unsupervised algorithm divides all the usages of an ambiguous word into a specified number of groups based upon the context in which each instance of the word occurs.

A feature set must be developed for both supervised and unsupervised learning. To clarify these concepts, both the graphical representation and parameter estimates associated with several examples of decomposable models are presented in terms of a simple word sense disambiguation example. The task is to disambiguate various instances of *watha* in Sinhala. Feature vectors for *watha* is given in Table 1.

<table>
<thead>
<tr>
<th>Sense-tagged sentences- Sinhala word - watha</th>
<th>Feature vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>sudu watha[cloths] andi katak mama dutimi</td>
<td>1 0 0 0</td>
</tr>
<tr>
<td>ege watha[face] madala sowin barawiya</td>
<td>0 1 0 0</td>
</tr>
<tr>
<td>semage watha[information] gotha eceema ohuge siritaki</td>
<td>0 0 1 0</td>
</tr>
<tr>
<td>ohu adi watha[cloths] pawa ugaseta thabeya</td>
<td>1 0 0 0</td>
</tr>
<tr>
<td>preethimath puwatha esu ohuge watha[face] madala sada watha[face] men aalokamath wiya</td>
<td>0 1 0 1</td>
</tr>
</tbody>
</table>

Table 1. Feature Vectors

This defines the context of the ambiguous word and properties of both the ambiguous word and the sentence in which it occurs that are relevant to making a sense distinction. These properties are referred to as contextual features. Human intuition and linguistic insight are necessary at this stage. The development of a feature set is a subjective process; given the complexity of human language there are a huge number of possible contextual features and it is not possible to empirically examine even a fraction of them. New feature sets appropriate for unsupervised learning for Sinhala language text is developed in this study.

3. UNSUPERVISED LEARNING

A general limitation of supervised learning approaches to word sense disambiguation is that sense-tagged text is not available for most domains. While sense-tagged text is not as complicated to create as more elaborate representations of real-world knowledge, it is still a time-consuming activity and limits the portability of methods that require it.

The unsupervised learning models consist of a parametric form and parameter estimates. The parametric form shows the contextual features that affect the values of other contextual features and the sense of the ambiguous word. The parameter estimates tell how likely certain combinations of values for the contextual features are to occur with a particular sense of an ambiguous word. Unsupervised learning presents an alternative that eliminates dependence on sense-tagged text. In word sense disambiguation, this corresponds to grouping instances of an ambiguous word into some pre-specified number of sense groups, where each group corresponds to a distinct sense of the ambiguous
word. We developed knowledge-learn approaches that learn probabilistic models from raw untagged text in Sinhala.

Raw text only consists of the words and punctuation that normally appear in a document; there are no manually attached sense distinctions to ambiguous words nor is any other kind of information augmented to the raw text. Even without sense-tagged text it is still possible to learn a probabilistic model using an unsupervised approach. In this case parametric form must be specified by the user and then parameter estimates can be made from the text. This research uses the parametric form of the Naive Bayesian classifier when performing unsupervised learning of probabilistic models. The parametric form of any probabilistic model of disambiguation must include a feature representing the sense of the ambiguous word; however, raw text contains no values for this feature. The sense is treated as a latent or missing feature. Two different approaches to estimating parameters given missing data are evaluated; the EM algorithm and Gibbs Sampling.

4. PROBABILISTIC MODELS

The parameters of a probabilistic model can be estimated using a number of approaches. Maximum Likelihood Estimation Values for the parameters of a probabilistic model can be estimated using maximum likelihood estimates such that \( \theta_i = \frac{f_i}{N} \). In this framework, a parameter can only be estimated if the associated event is observed in a sample of data. A maximum likelihood estimate maximizes the probability of obtaining the data sample that was observed, \( D \), by maximizing the likelihood function, \( p(D|\Theta) \). The likelihood function for a multinomial distribution is defined as follows:

\[
p(D|\Theta) = \frac{N!}{\prod_{i=1}^{q} \theta_{i}^{n_i}} \prod_{i=1}^{q} \theta_{i}^{n_i}
\]

Implicit in the multinomial distribution is the assumption that all the features of an event are dependent. When this is the case the value of any single feature variable is directly affected by the values of all the other feature variables. A probabilistic model where all features are dependent is considered saturated. However, if the event space is very small it may be reasonable to assume that all feature variables are dependent on one another and that every possible event can be observed in a data sample. For example, if an event space is defined by two binary feature variables, \( F_1, F_2 \), then the saturated model has four parameters, each representing the probability of observing one of the four possible events.

Table 2 shows a scenario where a sample consists of \( N = 150 \) events. The frequency counts of these events are shown in column \( \text{freq}(F_1, F_2) \), and the resulting maximum likelihood estimates are calculated and displayed in column \( \text{MLE} \). It is more often the case in real world problems that the number of possible events is somewhat larger than four. The number of parameters needed to represent these events in a probabilistic model is determined by the number of dependencies among the feature variables. If the model is saturated then all of the features are dependent on one another and the number of parameters in the probabilistic model is equal to the number of possible events in the event space.

<table>
<thead>
<tr>
<th>F1</th>
<th>F2</th>
<th>freq (F1, F2)</th>
<th>MLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>21</td>
<td>0.14</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>38</td>
<td>0.25</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>60</td>
<td>0.40</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>31</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Table 2: Maximum Likelihood Estimates (MLE)

In supervised learning, given the parametric form of a decomposable model, maximum likelihood estimates of parameters are simple to compute. The sufficient statistics of these parameters are the frequency counts of marginal events that are defined by the marginal distributions of the model. These counts are obtained directly from the training data.

Here the assumption is made that the parametric form of the model is Naive Bayes. In this model, all features are conditionally independent given the value of the classification feature, i.e., the sense of the ambiguous word. This assumption is based on the success of the Naive Bayes model when applied to supervised word-sense disambiguation. In these discussions, the sense of an ambiguous word is represented by a feature variable, \( S \), whose value is missing. The observed contextual features are represented by \( Y = (F_1, F_2, \ldots, F_q) \). The complete data sample is then \( D = (Y; S) \) and the parameters of the model are represented by the vector \( \Theta \).

5. EM ALGORITHM

The EM algorithm is an iterative estimation procedure in which a problem with missing data is recast to make use of complete data estimation techniques. The EM algorithm formalizes a long-standing method of making estimates for the parameters of a model, \( \Theta \), when data is missing. A high-level description of the algorithm is as follows:

1. Randomly estimate initial values for the parameters \( \Theta \). Call this set of estimates \( \Theta^{old} \)
2. Replace the missing values of S by their expected values given the parameter estimates \( \Theta^{old} \).
3. Re-estimate parameters based on the filled-in values for the missing variable S. Call these parameter estimates \( \Theta^{new} \).
4. Have \( \Theta^{old} \) and \( \Theta^{new} \) converged? If not, rename \( \Theta^{new} \) as \( \Theta^{old} \) and go to step 2.

At the heart of the EM Algorithm lies the Q-function. This is the expected value of the log of the likelihood function for the complete data sample, \( D = (Y, S) \) where Y is the observed data and S is the missing sense value:

\[
Q(\Theta^{new} | \Theta^{old}) = \mathbb{E} \left[ \ln p(Y, S|\Theta^{new}) | \Theta^{old}, Y \right]
\]

Here, \( \Theta^{old} \) is the previous value of the maximum likelihood estimates of the parameters and \( \Theta^{new} \) is the improved estimate; \( p(Y, S|\Theta^{new}) \) is the likelihood of observing the complete data given the improved estimate of the model parameters.

When approximating the maximum of the likelihood function, the EM algorithm starts from a randomly generated initial estimate of the model parameters and then replaces \( \Theta^{old} \) by the \( \Theta^{new} \) which maximizes \( Q(\Theta^{new} | \Theta^{old}) \). This is a two step process, where the first step is known as the expectation step, i.e., the E-step, and the second is the maximization step, i.e., the M-step. The E-step finds the expected values of the sufficient statistics of the complete model using the current estimates of the model parameters. For decomposable models these sufficient statistics are the frequency counts of events defined by the marginal distributions of the model. The M-step makes maximum likelihood estimates of the model parameters using the sufficient statistics from the E-step. These steps iterate until the parameter estimates \( \Theta^{old} \) and \( \Theta^{new} \) converge.

The M-step is usually easy, assuming it is easy for the complete data problem. In the general case the E-step may be complex. However, for decomposable models the E-step simplifies to the calculation of the expected marginal event counts defined by a decomposable model, where the expectation is with respect to \( \Theta^{old} \). The M-step simplifies to the calculation of new parameter estimates from these counts. Further, these expected counts can be calculated by multiplying the sample size N by the probability of the complete data within each marginal distribution, given \( \Theta^{old} \) and the observed data within each marginal \( Y_m \). This simplifies to:

\[
\text{freq}_{new}^{new}(S_m, Y_m) = p(S_m | Y_m) \times \text{freq}(Y_m)
\]

where \( \text{freq}^{new} \) is the current estimate of the expected count and \( p(S_m | Y_m) \) is formulated using \( \Theta^{old} \).

6. GIBBS SAMPLING

Gibbs Sampling is a more general tool than the EM algorithm; it is not restricted only to handle missing data; it is a special case of Markov Chain Monte Carlo methods for approximate inference. These methods were first used for applications in statistical physics in the 1950’s; perhaps the most notable example being the Metropolis algorithm[6]. Gibbs Sampling was originally presented in the context of an image restoration problem but has since been applied to a wide range of applications.

In general, Gibbs Sampling provides a means of approximating complex probabilistic models. In unsupervised learning probabilistic models are complex because there is missing data, i.e., the sense of the ambiguous word is unknown. Gibbs Sampling approximates the distribution of the parameters of a model as if the missing data were observed. By contrast, the EM algorithm simply maximizes the estimated values for the parameters of a model, again by acting as if the missing data were observed.

Gibbs Sampling has a Bayesian orientation in that it naturally incorporates prior distributions, \( p(\Theta) \), into the sampling process. When a prior distribution is specified in conjunction with an observed data sample, Gibbs Sampling approximates the posterior probability function, \( p(\Theta|D) \), by taking a large number of samples from it. If a prior distribution is not utilized then Gibbs Sampling still takes a large number of samples, however, they are drawn from the likelihood function \( p(D|\Theta) \). In this research, non-informative prior distributions are employed and the sampling is from the posterior distribution function. A Gibbs Sampler creates Markov Chains of parameter estimates and values for missing data whose stationary distributions approximate the posterior distribution, \( p(\Theta|D) \), by simulating a random walk in the space of \( \Theta \). A Markov Chain is a series of random variables \( (X^0, X^1, \ldots) \) in which the influence of the values of \( (X^0, \ldots, X^q) \) on the distribution of \( X^{q+1} \) is mediated entirely by the value of \( X^q \).

Let the values of the observed contextual feature variables be represented by \( Y = (F_1, F_2, \ldots, F_n, F_a) \) and let \( S \) represent the unknown sense of an ambiguous word. Given that the parametric form of the model is known, random initial values are generated for the missing data \( S^0 = (S_1^0, S_2^0, \ldots, S_n^0) \) and the unknown parameter estimates of the assumed model \( \Theta^0 = (\theta_1^0, \theta_2^0, \ldots, \theta_q^0) \). \( S^0 \) is a vector containing a value for each instance of the missing sense data; \( \Theta^0 \) is a vector containing the parameters of the model, \( N \) is the number of observations in the data sample, and \( q \) is the number of parameters in the model.
7. IMPLEMENTATION AND EXPERIMENTAL RESULTS

In addition to having many possible meanings, words are also ambiguous syntactically in that they can serve as multiple possible parts-of-speech. However, we do not handle syntactic ambiguity in this study.

In our experiment, Susantha-Corpus containing 215,000 word from newspaper articles is used.

<table>
<thead>
<tr>
<th>Word</th>
<th>Sense Count in Corpus</th>
<th>EM</th>
<th>Gibbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>'siya'</td>
<td>Sense 1 - tamage</td>
<td>210</td>
<td>0.912</td>
</tr>
<tr>
<td></td>
<td>Sense 2 - siyagananni</td>
<td>14</td>
<td>0.076</td>
</tr>
<tr>
<td>'mata'</td>
<td>Sense 1 - uda</td>
<td>213</td>
<td>0.924</td>
</tr>
<tr>
<td></td>
<td>Sense 2 - adahasa</td>
<td>10</td>
<td>0.078</td>
</tr>
<tr>
<td>'ek'</td>
<td>Sense 1 - eka <img src="http://example.com" alt="1" /></td>
<td>183</td>
<td>0.874</td>
</tr>
<tr>
<td></td>
<td>Sense 2 - ekattuwa</td>
<td>18</td>
<td>0.087</td>
</tr>
<tr>
<td></td>
<td>Sense 3 - wenas</td>
<td>5</td>
<td>0.043</td>
</tr>
<tr>
<td>'diya'</td>
<td>Sense 1 - jalaya</td>
<td>78</td>
<td>0.654</td>
</tr>
<tr>
<td></td>
<td>Sense 2 - labadeema</td>
<td>41</td>
<td>0.358</td>
</tr>
<tr>
<td>'mul'</td>
<td>Sense 1 - prathma</td>
<td>52</td>
<td>0.683</td>
</tr>
<tr>
<td></td>
<td>Sense 2 - pradana</td>
<td>13</td>
<td>0.178</td>
</tr>
<tr>
<td></td>
<td>Sense 3 - hetuwa</td>
<td>8</td>
<td>0.107</td>
</tr>
<tr>
<td></td>
<td>Sense 4 - gasemul</td>
<td>3</td>
<td>0.098</td>
</tr>
</tbody>
</table>

Table 3 illustrates experimental results. These experiments assume that the parametric form is Naïve Bayes and use two different methods to estimate parameter values; the EM algorithm and Gibbs Sampling. Five words are disambiguated using 2 unsupervised methodologies of EM and Gibbs. An unsupervised algorithm is limited to creating sense groups. A sense group is simply a number of instances of an ambiguous word that are considered to belong to the same sense.

8. CONCLUSIONS

The development and improvement of unsupervised learning techniques is an important issue in natural language processing given the difficulty in obtaining training data for supervised learning. The lack of sense-tagged text poses a considerable bottleneck when porting supervised learning methods to new domains and unsupervised methods over a way to eliminate this need for sense-tagged text.

Several feature sets appropriate for unsupervised learning of word senses from raw text were developed. Feature sets designed for use with supervised approaches are not directly applicable in an unsupervised setting since they often contain features whose values are based on information only available in sense-tagged text. The local context features developed for unsupervised learning are co-occurrences that occur within a few positions of the ambiguous word. This study developed probabilistic models for word sense disambiguation without utilizing sense-tagged text. The EM algorithm and Gibbs Sampling were used to estimate the parameters of probabilistic models of disambiguation based strictly upon information available in raw untagged text in Sinhala.

REFERENCES