Parts-of-Speech Tagging using Maximum Entropy Model

A dissertation submitted in partial fulfillment of the requirements for the M.Tech.(Computer Science) degree of Indian Statistical Institute

By

Swadesh Pratap singh Shakya

Roll No: CS0910

Under the supervision of

Associate Prof. Utpal Garain

INDIAN STATISTICAL INSTITUTE

203, Barrackpore Trunk Road
Kolkata-700108

Indian Statistical Institute

CERTIFICATE

This is to certify that the thesis entitled '*Parts-of-speech tagging using maximum entropy model*' is submitted in the partial fulfillment of the degree of M.Tech. in Computer Science at Indian Statistical Institute, Kolkata.

The work out by Swadesh under my supervision and guidance is adequate, in scope and quality as a dissertation for the required degree. It is further certified that no part of this thesis has been submitted to any other university or institute for the award of any degree or diploma.

Associate Prof. Utpal Garain
(Supervisor)

Countersigned
(External Examiner)
Date: of July 2010

Acknowledgement

I take this opportunity to thank **Associate Prof Utpal Garain**, CVPR Unit, ISI Kolkata for his valuable guidance, inspiration. His pleasant and encouraging words have always kept my spirits up. I am grateful to him for providing me to work under his supervision. Also, I am very thankful to him for giving me the idea behind the algorithm developed in thesis work.

Finally, I would like to thank all my colleagues, class mates, friends, and my family members for their support and motivation.

Swadesh

M.Tech.(CS)

Contents

Chapter 1

Introduction

Many different researchers, using a wide variety of techniques, have examined the task of Part-of-Speech (POS) tagging. The task itself consists of assigning basic grammatical word classes such as verb, noun and adjective to individual words, and is a fundamental step in many Natural Language Processing (NLP) tasks. The tags it assigns are used in other processing tasks such as chunking and parsing, as well as more complex tasks such as question answering and automatic summarization systems. Maximum Entropy modeling is one of the techniques that have been used to perform POS tagging, and gives state-of-the-art accuracy

We aim to find better ways to perform POS tagging on unknown words. We will use an existing Maximum Entropy POS tagger that already performs at state-of-the-art level, and implement additional new features in order to increase its accuracy. These features will be able to represent real values in any range greater than zero, rather than a binary 0 or 1 as has been the case for Maximum Entropy modeling system have used in the past.

The features themselves will encapsulate information found from the context around a word, as observed for unknown words. For example, if we find an unknown word in the test data, then it may still appear many times in a much larger unannotated corpus. By looking at the surrounding words in these contexts, we can formulate an idea of what POS tag should be assigned. This can be seen in the sentence below:

The **frub** house is up on the hill

Here, **frub** is the unknown word, and as a human we could conclude that it is an adjective or noun. This is because it sits between a determiner and a noun, which is a position often assumed by words with these two tags. Also, if we can find the word **frub** in other places, then we can get an even better, more reliable

idea of what its correct tag should be. This is what the large un-annotated corpus gives us: a number of examples of how and where unknown words are used.

We should also note that we do not need to know the correct POS tags for the and house. We can determine simply from the words themselves, that **frub** is occupying a position that is also taken up by words such as big or club, these being examples of adjectives and nouns respectively. Also, the fact that the word the precedes our unknown word tells us a lot by itself, as this is an extremely common word that exists with only tag. Our aim then, is to take this intuitive reasoning for determining the correct tag for an unknown word, and create features that aid the Maximum Entropy model in doing the same.

We will begin by describing the previous work that has taken place on the task of POS tagging, including the corpora that are used and the techniques that have been applied to the task. This will continue onto particular methods that have attempted to better classify unknown words, and then the statistical machine learner that we will be using: Maximum Entropy modeling. This will be followed by an extensive description of the experiments we performed, and the alterations to the Maximum Entropy features and calculations that were required to achieve the best performance. We then proceed to a thorough analysis and discussion of the results we attained, and finally, further applications, uses of, and improvements to the methods described.

Chapter 2

Background

2.1: Part-of-Speech Tagging:

In the following two sentences,

- Fruit flies like a banana.
- Time flies like an arrow.

The words *flies* and *like* are ambiguous. In the first sentence, *flies* is a noun and *like* is a verb, while in the second sentence, *flies* is a verb and *like* is a preposition. How can a computer program automatically and accurately predict the part-of-speech of ambiguous words *flies* and *like*?

2.2: Maximum Entropy

There are a number of machine learning techniques that can be applied to problems such as POS tagging, prepositional phrase attachment and parsing, as well as areas outside the NLP field. Maximum Entropy (MaxEnt) modeling is one of these techniques, which estimates a statistical model to give probabilistic results (Ratnaparkhi, 1996). It makes no assumptions about the independence of features, as is the case with other classifiers like Naive Bayes, and because its results are probabilistic, can easily be used in a larger framework for classification.

A Maximum Entropy model is built on a number of *constraints*, which are drawn as *features* from the training data. Once these constraints are met, the model assumes nothing further, giving a uniform distribution, and the model with maximum entropy, as suggested by the name. In this way, the model makes use of all the information available, but does not favor any further unfounded hypothesis, giving equal chance to all possibilities (Berger et al., 1996).

The observed expectation of features functions, as observed in the training, is calculated by:

$$E_{p'}f_j(h,t) = \sum_{i=i}^{N} p'(h_i, t_i) f_j(h_i, t_i) \quad (1)$$

Where $p'(h_i, t_i)$ denotes the observed probability of (h_i,t_i) and h_i is the history of word w_i, t_i is the tag of the tag of word w_i in training data.

Similarly, the model's expectation of features functions, is calculated by:

$$E_p f_j(h,t) = \sum_{h\in H, t\in T} p(h,t) f_j(h,t) \quad (2)$$

In practice, H is very large and the model's expectation $E_p f_j(h,t)$ cannot be computed directly, so the following approximation is used:

$$E_p f_j(h,t) \approx \sum_{i=1}^{N} p'(h_i) p(t_i|h_i) f_j(h_i, t_i) \quad (3)$$

Where $p'(h_i)$ is the observed probability of the history h_i in the training set.

In this equation, the probability model is calculated as the sum over all features, of the product of the frequency of a contextual predicate, the classification *h* given that contextual predicate, and the feature that determines whether this probability is taken into account. In the two previous equations, the feature function serves the purpose of including probabilities when the function is true. That is, if the contextual predicate occurs in the context we are looking at, and the classification (such as a particular POS tag) matches that of the current context, then *p'(h,t)* (which is a simple measure of frequency) is said to be *active*, and is used in calculating the probability for the feature.

We should also note, that the model should be an accurate reflection the training data. This is an important point, as it clearly makes sense that the statistical distributions in the training data, which are meant to be representative

of language in general, should be the same as those in the model. From this idea, we have:

$$E_p f_j(h,t) = E_{p'} f_j(h,t), \qquad 1 \leq j \leq k \tag{4}$$

And therefore,

$$\sum_{i=1}^{N} p'(h_i) p(t_i|h_i) f_j(h_i, t_i) = \sum_{i=i}^{N} p'(h_i, t_i) f_j(h_i, t_i)$$

$$P = \{p: E_p f_j(h,t) = E_{p'} f_j(h,t) = d_j(say), \qquad 1 \leq j \leq k\} \tag{5}$$

This gives us a mathematical approach towards finding a subset of models, from all possible probability models, where the constraints found in the training data match the probabilities in the estimated model. More simply, those models that satisfy the above equation for all features, will have a probability distribution identical to that of the training data.

Satisfying these constraints does not result in a unique solution, and so going back to the basic idea of a Maximum Entropy model, we should choose the solution that has the most uniform distribution. In order to calculate uniformity, we can use the mathematical measure of entropy, and in particular, conditional entropy, as described in the following equation

$$H(p) = -\sum_{h \in H, t \in T} p(h,t) \log p(h,t)$$

$$= -\sum_{h \in H, t \in T} p'(h) p(t|h) \log p(t|h) \tag{6}$$

The value of $H(p)$ will range from 0, where all probability is given to one item, to $\log|h|$, where $|h|$ is the number of possible classifications that can be made (which for POS tagging is the number of POS tags). The most uniform distribution is the one that maximizes the entropy, that is:

$$p^* = \underset{p \in P}{argmax}\ H(p) \qquad (7)$$

Where p^* is the Maximum Entropy model we are trying to find, and P is the set of all probability distributions that meet the constraints as specified in Equation 5.

2.3: Calculation for p*:

Maximize $E(p,\lambda)$:

$E(p) = H(p) + \sum_{j=0}^{k} \lambda_j (E_p f_j(h,t) - d_j)$

$E'(p) = 0$

$$\Rightarrow \frac{\delta}{\delta p(x)} \left(-\sum_x p(x) log p(x) + \sum_{j=0}^{k} \lambda_j \left(\left(\sum_x p(x) f_j(x) \right) - d_j \right) \right) = 0$$

$$\Rightarrow -(1 + \log p(x)) + \sum_{j=0}^{k} \lambda_j f_j(x) = 0$$

$$\Rightarrow \log p(x) = \sum_{j=0}^{k} \lambda_j f_j(x) - 1$$

$$\Rightarrow p(x) = \exp\left(\sum_{j=0}^{k} \lambda_j f_j(x) - 1\right) = \exp\left(\sum_{j=1}^{k} \lambda_j f_j(x) + \lambda_0 - 1\right)$$

$$\Rightarrow p(x) = \frac{\exp\left(\sum_{j=1}^{k} \lambda_j f_j(x)\right)}{Z} \quad \text{where } Z = \exp(1 - \lambda_0) \ \& \ x = (h, t)$$

To maximize E(p), E''(p) should be less than zero so

E''(p(x)) = $\frac{\delta}{\delta p(x)}(-(1 + \log p(x)) + \sum_{j=0}^{k} \lambda_j f_j(x))$

= $-\frac{1}{p(x)} < 0$

Maximum Entropy model:

$$p(h,t) = \frac{1}{Z}\exp\left(\sum_{j=1}^{k} \lambda_j f_j(h,t)\right)$$

Conditional Maximum Entropy Model:

$$p(t|h) = \frac{1}{Z(h)}\exp\left(\sum_{j=1}^{k} \lambda_j f_j(h,t)\right)$$

Where Z(h) = $\sum_{h,t}\exp\left(\sum_{j=1}^{k} \lambda_j f_j(h,t)\right)$

So E(p) would be maximum at p(x)

Put Z = $\frac{1}{\pi}$ and $\lambda_j = \log \alpha_j$

p(x) = $\pi \prod_{j=1}^{k} {\alpha_j}^{f_j(x)}$

So, Our Maximum Entropy Model is **p(x) = $\boldsymbol{\pi} \prod_{j=1}^{k} {\boldsymbol{\alpha_j}}^{f_j(x)}$**

2.4: Parameter Estimation:

2.4.1: Generalized Iterative Scaling:

GIS is a very simple algorithm for estimating the parameters of a Maximum Entropy model. The algorithm is as follows, where $E_{p'}f_j$ is the observed expected value of f_j and $E_p f_j$ is the expected value according to model p:

Set $\lambda_j^{(0)}$ equal to arbitrary value, say:

$$\lambda_j^{(0)} = 0$$

Repeat until convergence:

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} + \frac{1}{C} log \frac{E_{p'} f_j}{E_{p^{(t)}} f_j}$$

Where (t) is the iteration index and the constant C is defined as follows:

$$C = \max \sum_{j=1}^{k} f_j(h, t) \qquad (8)$$

In practice C is maximized over the (h,t) pairs in the training data, although in theory C can be any constant greater than or equal to the figure in (8). However, since $\frac{1}{C}$ determines the rate of convergence of the algorithm, it is preferable to keep C as small as possible.

Proof:

This proof of GIS convergence without the correction feature is based on the IIS convergence proof by Berger (1997).

Start with some initial model with arbitrary parameters $\Delta = \{\lambda_1, \lambda_2, \ldots, \lambda_k\}$. Each iteration of the GIS algorithm finds a set of new parameters $\Delta' = \Delta + \delta = \{\lambda_1 + \delta_1, \lambda_2 + \delta_2, \ldots, \lambda_k + \delta_k\}$. which increases the log-likelihood of the model.

The change in log-likelihood is as follows:

$L_{p'}(\Delta') - L_{p'}(\Delta)$

$$= \sum_{h,t} p'(h,t) log p_{\Delta'}(t|h) - \sum_{h,t} p'(h,t) log p_{\Delta}(t|h)$$

$$= \sum_{h,t} p'(h,t) log \frac{1}{Z_{\Delta'}(h)} \exp\left(\sum_{j=1}^{k} (\lambda_j + \delta_j) f_j(h,t)\right)$$

$$- \sum_{h,t} p'(h,t) log \frac{1}{Z_{\Delta}(h)} \exp\left(\sum_{j=1}^{k} \lambda_j f_j(h,t)\right)$$

$$= \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t) - \sum_{h} p'(h) \log \frac{Z_{\Delta'}(h)}{Z_{\Delta}(h)}$$

As in Berger (1997), use the inequality $-\log\alpha \geq 1 - \alpha$ to establish a lower bound on the change in likelihood:

$L_{p'}(\Delta') - L_{p'}(\Delta) \geq$

$$\sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t) + \sum_h p'(h) \left(1 - \frac{Z_{\Delta'}(h)}{Z_\Delta(h)}\right)$$

$$= \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t) + 1 - \sum_h p'(h) \left(\frac{Z_{\Delta'}(h)}{Z_\Delta(h)}\right)$$

$$= 1 + \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t)$$

$$- \sum_h p'(h) \sum_t \frac{1}{Z_\Delta(h)} exp \sum_{j=1}^{k} (\lambda_j + \delta_j) f_j(h,t)$$

$$= 1 + \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t)$$

$$- \sum_h p'(h) \sum_t p_\Delta(t|h) exp \sum_{j=1}^{k} \delta_j f_j(h,t)$$

Call the right hand side of this last equation $A(\delta|\Delta)$. If we can find a δ for which $A(\delta|\Delta) > 0$, then $L_{p'}(\Delta + \delta)$ is an improvement over $L_{p'}(\Delta)$. The obvious approach is to maximize $A(\delta|\Delta)$ with respect to each δ_j, but this cannot be performed directly, since differentiating $A(\delta|\Delta)$ with respect to δ_j leads to an equation containing all elements of δ.

The trick is to rewrite $A(\delta|\Delta)$ as follows, with an extra term which will be used to satisfy Jensen's inequality:

$$A(\delta|\Delta) = 1 + \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t)$$
$$- \sum_h p'(h) \sum_t p_\Delta(t|h) exp \sum_{j=1}^{k+1} \frac{f_j(h,t)}{C} C\delta_j$$

Where C is previously defined in equation a, $f_{n+1}(h, t) = f_c(h, t)$ as in (9), and δ_{n+1} is defined to be zero. Note that the correction feature has been introduced but has been given a constant weight of zero.

The next part of the proof introduces another, less tight, lower bound on the change in likelihood, by using Jensen's inequality, which can be stated as follows:

Let f be a convex function on the interval I. If $x_1, x_2, \ldots x_n \in I$ and $t_1, t_2, \ldots t_n$ are non-negative real numbers such that $\sum_{i=1}^{n} t_i = 1$, then

$$f(\sum_{i=1}^{n} t_i x_i) \leq \sum_{i=1}^{n} t_i f(x_i)$$

Since $\sum_{j=1}^{k+1} \frac{f_j(h,t)}{C} = 1$ and the exponential function is convex, we can apply Jensen's inequality to give a new form of $A(\delta|\Delta)$:

$$A(\delta|\Delta) \geq 1 + \sum_{h,t} p'(h,t) \sum_{j=1}^{k} \delta_j f_j(h,t) - \sum_h p'(h) \sum_t p_\Delta(t|h) \sum_{j=1}^{k+1} \frac{f_j(h,t)}{C} \exp(C\delta_j)$$

Call this bound $B(\delta|\Delta)$. Della Pietra et al. (1997) give extra conditions on the continuity and derivative of the lower bound, in order to guarantee convergence. These conditions can be verified for $B(\delta|\Delta)$ in a similar way to Della Pietra et al. (1997).

Differentiating $B(\delta|\Delta)$ with respect to each weight update δ_j ($1 \leq \delta_j \leq k$) gives:

$$\frac{\partial B(\delta|\Delta)}{\partial \delta_j} = \sum_{h,t} p'(h,t) f_j(h,t) - \sum_h p'(h) \sum_t p_\Delta(t|h) f_j(h,t) \exp(C\delta_j)$$

The effect of introducing C is that solving $\frac{\partial B(\delta|\Delta)}{\partial \delta_j}$ = 0 can be done analytically (at the cost of a slower convergence rate), giving the following:

$$\delta_j = \frac{1}{C}\log \frac{\sum_{h,t} p'(h,t) f_j(h,t)}{\sum_h p'(h) \sum_t p_\Delta(t|h) f_j(h,t)}$$

$$= \frac{1}{C}\log \frac{E_{p'} f_j(h,t)}{E_{p(t)} f_j(h,t)}$$

Which leads to the update rule in (7).

Chapter 3

Development of a POS Tagger

3.1: History:

To make history of any word, we require that current word, previous two word, next two word, tag of previous two word.

$$h_i = \{w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}, t_{i-1}, t_{i-2}\}$$

This is the history of i^{th} word.

Word:	The	stories	about	well-heeled	communities	and	developers
Tag:	DT	NNS	IN	JJ	NNS	CC	NNS
Position:	1	2	3	4	5	6	7

Table1: Sample Data

History of 1^{st} word will be:

h_1 = {the, stories, about}

History of 2^{nd} word will be:

h_2 = {the, stories, about, well-heeled, DT}

History of 3rd word will be:

h_3 = {the, stories, about, well-heeled, communities, DT, NNS}

History of 4th word will be:

h_4 = {stories, about, well-heeled, communities, and, NNS, IN}

History of 5th word will be:

h_5 = {about, well-heeled, communities, and, developers, IN, JJ}

History of 6th word will be:

h_6 = {well-heeled, communities, and, developers, JJ, NNS}

History of 7th word will be:

h_7 = {communities, and, developers, NNS, CC}

3.2: Features for POS Tagging:

The joint probability of a history h and tag t is determined by those parameters whose corresponding features are active, i.e., those α_j such that $f_j(h,t) = 1$. A feature, given (h,t), may activate on any word or tag in the history h.

For example,

$$f_j(h_i, t_i) = \begin{cases} 1 & if\ suffix(w_i) = "ing"\ and\ t_i = VBG \\ 0 & otherwise \end{cases}$$

If the above feature exists in the feature set of the model, its corresponding model parameter will contribute towards the joint probability $p(h_i,t_i)$ when w_i ends with "ing" and when t_i =VBG. Thus a model parameter α_j effectively serves as a "weight" for a certain contextual predictor, in this case the suffix "ing", towards the probability of observing a certain tag, in this case a VBG.

The model generates the space of features by scanning each pair (h_i, t_i) in the training data with the feature "templates" given in Table 2. Given h_i as the current history, a feature always asks some yes/no question about h_i, and furthermore constrains t_i to be a certain tag.

For example, Table 1 contains sample from training data while Table 3 contains the features generated while scanning (h_3, t_3), in which the current word is about, and Table 4 contains features generated while scanning (h_4, t_4), in which the current word, well-heeled, frequency of well-heeled is 3, i.e. less than 5 in training data .

Condition	Features
Frequency of $w_i \geq 5$	w_i = Word and t_i= Tag
Frequency of $w_i < 5$	Word is prefix of w_i , size of Word < 5 and t_i= Tag
	Word is suffix of w_i , size of Word < 5 and t_i= Tag
	w_i contains number and t_i= Tag
	w_i contains Uppercase character and t_i= Tag
	w_i contains hyphen and t_i= Tag
For all w_i	t_{i-1}= Tag of w_{i-1} and t_i= Tag
	t_{i-1} t_{i-2}= Tag of $w_{i-1.}$ Tag of w_{i-2} and t_i= Tag
	w_{i-1} = previous word and t_i= Tag
	w_{i-2} = previous to previous word and t_i= Tag
	w_{i+1} = next word and t_i= Tag
	w_{i+2} = next to next word and t_i= Tag

Table2: Features on the current history h_i

w_i = about	& t_i = IN
w_{i-1} = stories	& t_i = IN
w_{i-2} = the	& t_i = IN
w_{i+1} = well-heeled	& t_i = IN
w_{i+2} = communities	& t_i = IN
t_{i-1} = NNS	& t_i = IN
t_{i-2} = DT	& t_i = IN

Table 3: Features Generated From h_3 from table 1

w_{i-1} = about	& t_i = JJ
w_{i-2} = stories	& t_i = JJ
w_{i+1} = communities	& t_i = JJ
w_{i+2} = and	& t_i = JJ
t_{i-1} = IN	& t_i = JJ
t_{i-2} = NNS	& t_i = JJ
prefix(w_i) = w	& t_i = JJ
prefix(w_i) = we	& t_i = JJ
prefix(w_i) = wel	& t_i = JJ
prefix(w_i) = well	& t_i = JJ
suffix(w_i) = d	& t_i = JJ
suffix(w_i) = ed	& t_i = JJ
suffix(w_i) = led	& t_i = JJ
suffix(w_i) = eled	& t_i = JJ
w_i contains hypen	& t_i = JJ

Table 4: Feature Generated From h_4 from table 1

Chapter 4

Experimental results

In this chapter, We will describe our experimental results. We downloaded *Stanford tool kit for maxent* from *http://nlp.stanford.edu/software/tagger.shtml* We trained the tool kit with Bengali tagged text file consisting of 4318 tagged sentences and tested on untagged Bengali test file consisting of 150 sentences. The Stanford system is giving 90.58 % accuracy. We have trained our system on the same training data and tested on the same test file and our system is giving 93.81 % accuracy which is improvement with the existing algorithm. Our system is giving better results for Bengali text file.

4.1: Error types for Our System:

Word	Correct tag	Our Model's tag
এল	WQ	VM
মাঝে	XC	NN
দিল	VAUX	VM
আছে	VAUX	VM
করে	RP	VM
খুব	QF	INTF
নিয়ে	VAUX	VM
স্নান-খাওয়ার	VM	_

হল	VM	VAUX
বেশ	QF	INTF
যে	DEM	CC
বেশ	QF	INTF
আর	RP	CC
করে	RP	VM
করে	RP	VM
দিল	VAUX	VM
গিয়ে	VAUX	VM
দিতে	VAUX	VM
গিয়ে	VAUX	VM
কেমন	JJ	DEM
ম্যাড়	RDP	NN
যেন	CC	RP
নিয়ে	PSP	VM
নিয়ে	PSP	VM
নিয়ে	PSP	VM
সে	NNP	PRP
প্রতিবাদ	NN	XC
হয়েছে	VM	VAUX

গিয়ে	VAUX	VM
এত	INTF	QF
যে	RP	CC
সে	DEM	PRP
ধীরে	RDP	RB
থেকে	VAUX	PSP
সামান্য	RB	JJ
ধরে	PSP	VM
থাকে	VAUX	VM
করে	RP	VM
কেমন	INTF	DEM
ধরল	VAUX	VM
এমন	DEM	JJ
কোনরকম	DEM	QF
আছে	VAUX	VM
ধীরে	RDP	RB
দিয়ে	VAUX	PSP
হল	VM	VAUX
কেন-র	NN	_
করে	RP	VM

এ	DEM	NN
করে	RP	VM
যে	DEM	CC
পেল	VM	VAUX
আর	QF	CC
হল	VM	VAUX
সেই	XC	DEM
যে	DEM	CC
থাকত	VAUX	VM
পাওয়ার	VAUX	VM
ধরে	PSP	VM
যে	DEM	CC
আপনাদের	PRP	NNP
ও	PRP	CC
হবে	VAUX	VM
এই	PRP	DEM
যাক	VM	VAUX
প্রতিবাদ	NN	XC
যে	RP	CC
যে	DEM	CC

প্রতিবাদ	NN	XC
যে	DEM	CC
প্রতিবাদ	NN	XC
আপনাদের	PRP	NNP
আর	RP	CC
করে	RP	VM
কিন্তু	RP	CC
আপনাদের	PRP	NNP
চেয়ে	VM	PSP
নেই	VAUX	VM
নিয়ে	PSP	VM
কি	QF	WQ
এল	VAUX	VM
এ	DEM	NN
আর	RP	CC
কিছু	PRP	QF
নিয়ে	PSP	VM
এল	VAUX	VM
যেতে	VAUX	VM
এসেছে	VM	VAUX

কোন	WQ	QF

4.2: Error types for Stanford System:

Word (frq)	Correct tag	Stanford Model's tag
I (9)	SYM	NN
" (27)	SYM	NN
এল (1)	WQ	VM
মাঝে (1)	XC	NN
" (24)	SYM	XC
জন্য (1)	PSP	XC
এর (1)	PRP	QF
আগে (1)	NST	NN
কখনো (1)	PRP	NN
এখন (1)	PRP	NN
" (2)	SYM	INJ
" (7)	SYM	NNP
করে (1)	RP	VM
ম্যাড় (1)	RDP	NN
সঙ্গে (1)	NST	CC

নিয়ে (1)	PSP	VM
সে (1)	NNP	PRP
সে (1)	DEM	PRP
এখনো (1)	PRP	NN
সামান্য(1)	RB	JJ
ছাড়া (1)	PSP	CC
ধরে (1)	PSP	VM
গেলে (1)	VAUX	VM
যায় (1)	VAUX	VM
গেছে (1)	VAUX	VM
এমন (1)	DEM	PRP
কোনরকম(1)	DEM	QF
এখন (1)	PRP	NN
তারপর(1)	RB	NN
কেন-র(1)	NN	WQ
" (1)	SYM	PSP
ছাড়াও (1)	PSP	CC
সেই (1)	XC	PRP
কাঠ (1)	JJ	NN
এসেছি (1)	VM	VAUX

ও	(1)	PRP	CC
সরাসরি	(1)	RB	JJ
যে	(1)	DEM	WQ
পারবেন	(1)	VAUX	VM
কিন্তু	(1)	RP	CC
নিয়ে	(1)	PSP	VM
/	(1)	SYM	XC
/	(1)	SYM	NN
কি	(1)	QF	WQ
সঙ্গে	(1)	NST	CC
না	(2)	NEG	RB
"	(1)	SYM	RB
এসেছে	(1)	VM	VAUX
কোন	(1)	WQ	QF

Chapter 5

Future work

In feature work, we intend to apply natural language learning to corpora that are linguistically deeper, as well as corpora that are not linguistically annotated.

The development of a machine learning based good accuracy POS tagger requires a large amount of training data. The future work also includes the development of a large amount of annotated data which can be further used for training the system. The present tagger can be used for the initial annotation and the errors can be manually checked which otherwise a very difficult task to annotate large amount of corpus.

We also plan to explore some other machine learning algorithms (e.g. Support Vector Algorithm and Neural Networks) to understand their relative performance of POS Tagging task under the current experimental setup. MAXENT based models do not work well when the amount of annotated data is less. This might be due to the effect of transition probability over emission probability in the sequence identification. Support Vector Algorithm, Neural Networks or Decision Tree based algorithms might overcome the above situation.

Chapter 6

Conclusion

The aim of this project was to use the information in an unannotated corpus - in particular, the contexts surrounding unknown words - in order to increase performance on POS tagging unknown words. Although a number of techniques have been applied to this problem in the past, none attempted to draw upon the information that could be found in a larger amount of raw text. The advantage of this method, is that it only requires an unannotated corpus, which is much easier to create than a corpus like the Penn Treebank, and so it is easily portable to another domain or language.

In particular, the use of real-valued features resulted in a much larger improvement than binary features would have given us. Maximum Entropy features in the past have always been limited in this respect, and seeing the results we attained, one cannot doubt the benefits that real-valued features can bring. The increased flexibility they give us, and their ability to capture relationships between values, make them extremely advantageous. One can see that the kind of information we were trying to represent was a good example case for their usage, but there are many other features that would also be intrinsically suited to them.

POS tagging itself is a task that has been studied in detail, and consequently, it is also a task in which any increase in performance is hard to achieve. Furthermore, even a small improvement is worthwhile, because when one tags a large amount of text, even a small increase in accuracy will reduce the number of errors significantly. Our work has shown, through the thorough and extensive series of experiments that have been performed, that the techniques we implemented did indeed result in a substantial increase in accuracy.

Chapter 7

References

[Berger et al., 1996] Adam Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39-71

[Brill.] Eric Brill. A Simple Rul-Based Part-of-Speech Tagger. University of Pennsylvania.

[Chuneh, Wang and Chien, 2006] Chuang-Hua Chues, Hasin-Min Wang and Jen-Tzung Chien. 2006. A Maximum Entropy Approach for Semantic Language Modeling, Vol. 11, No.1, March 2006, pp. 37-56.

[Darroch and Ratcliff, 1972] J. N. Darroch and D. Ratcliff. 1972. Generalized Iterative Scaling for Log-Linear Models. *The Annals of Mathematical Statistics, 43(5):1470-1480.*

[Dandapat, 2009] Sandipan Dandapat. 2009. Parts-of-Speech Tagging for Bengali. Ph.D. thesis, Indian Institute of technology, Kharegpur

[Dalal, Nagaraj, Sawant, Shelke] Aniket dalal, Kumar Nagraj, Uma Sawant, Sandeep Shelke. Hindi Parts-of-speech Tagging and Chaunking: A Maximum Entropy Approach, Indian Institute of Technology, Mumbai.

[Ratnaparkhi. 1996] Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. *In Proceedings of the EMNLP Conference*, pages 133–142, Philadelphia, PA.

[Ratnaparkhi. 1998.] Adwait Ratnaparkhi. 1998. Maximum Entropy Models for Natural Language Ambiguity Resolution. Ph.D. thesis, *University of Pennsylvania*.

[Ratnaparkhi. 1999.] Adwait Ratnaparkhi. 1999. Learning to parse natural language with maximum entropy models. Machine Learning, 34(1-3):151–175.

[Vadas. 2004.] David Vadas. 2004. POS Tagging Unknown Words using an Unannotated Corpus and Maximum Entropy. Ph.D. thesis, The University of Sydney