Journal of Theoretical and Applied Information Technology

www.jatit.org

WORD PREDICTOR USING NATURAL LANGUAGE GRAMMAR INDUCTION TECHNIQUE

K.Sundarkantham, S.Mercy Shalinie
Department of Computer Science and Engineering
Thiagarajar college of Engineering
Madurai-625015, TamilNadu , India
([1]kskcse@tce.edu , [2]shalinie@tce.edu)
Tel: +91- 452-2482240 Ext:507 Fax: +91-452-2483427

ABSTRACT

Language is a unique phenomenon that distinguishes man from other animals. It is our primary method of communication with each other, yet very little is understood about how language is acquired when we are infants. A greater understanding in this area would have the potential to improve man machine communication

The problem that is attempted to be solved in this paper is that of programming a computer to play the Shannon Game. To play the Shannon game, one must predict which words are most likely to follow a given segment of English Text. Word Prediction would be most useful for writers with physical disabilities and severe spelling problems. The aim of this paper is to improve on existing results by writing a program that is capable of automatically inferring a grammar from a Natural Language Corpus, and applying this to the Shannon Game.

To play the Shannon Game, a stochastic Grammar for an approximation to the target language must be inferred from a text sample, and as the quality of this grammar improves so too does the quality of the predictor that uses the inferred grammar. The proposed algorithm in the paper uses Support Vector Machine to perform the part of speech tagging which produces 97.6% correct predictions.

Key words:
Natural Language Grammatical Inference, K-Means Clustering, Support Vector Machines

1. INTRODUCTION

This paper explores possible solutions to a problem that human beings find trivial, and a good solution would allow computer programs to process information in a more natural manner. The language models currently used for word prediction, such as IBM trigram model employed may be improved by this attempt. Word Prediction would be most useful for writers with physical disabilities like learning disability in reading and writing, attention deficit disorder. The most common and generally useful assistive technology for basic writing skills is spell checking [Venkatagiri 1993, MacArthur et al 1996]. In comparison with spell checkers, word prediction has both potential advantages and limitations. Students whose misspellings are too severe for correction by a spelling checker may benefit from word prediction. Word prediction does not require a user to type the entire word; consequently, knowing the first one to three letters may be sufficient for an accurate prediction of many words.

There were some attempts of natural language grammatical inference using evolutionary algorithms [Margaret Ayciena et al 2003] with little success on real examples. This paper uses Support Vector Machines for Part of Speech Tagging (POS) of English words. In the recent literature it can be found that there are several approaches to POS tagging based on statistical and machine learning techniques including many others: Hidden Markov Models [Weischedel 1993,Brants 2000], Maximum Entropy Taggers

[Ratnaparkhi 1996], Transformation –based learning [Brill 1995] ,Decision Trees [Marquez et al 2000]. Support Vector Machine based tagger introduced in this work fulfills the requirements for being a practical tagger and offers a very good balance of the High Accuracy and Speed. It is worth noting that that the Support Vector Machines (SVM) paradigm has been already applied to tagging in a paper [Nakagawa et al 2001] with the focus on the guessing of unknown word categories. The tagger constructed in the above paper gave a clear evidence that the SVM approach is specially appropriate for the flexibility and robustness, the main drawback being a low efficiency (in that paper a running speed of around 20 words per second is reported). In the present work, this limitation is overcomed by working with RBF kernels in the primal setting of the SVM framework taking advantage of the extremely sparsity of example vectors.

The Shannon game was proposed by Shannon himself, and has been adopted as a technique of presenting the word prediction problem The participants in the Shannon game are presented with the string of k words from an English text that contains N words altogether. The aim of the Shannon Game is for each participant to guess which word is likely to follow their given string. Even better, each participant could suggest several words that could follow the string, along with the probability of each suggested word occurring. A score may be kept by counting the number of attempts that the participant requires to correctly guess the next word, or by measuring how surprised the participant is when he or she discovers which word actually occurred.

This is all very well for human participants, who are capable of estimating their surprise [Noam Chomsky 1975 and Jeffrey L.Elman 1981]. A measure suitable for comparing the performance of computer implementation is required. Computer implementations may give their prediction as a probability distribution over the alphabet, and this distribution is often used by performance Measure.

2. PERFORMANCE MEASURES

The surprise experienced by the predictor when it discovers which word actually followed the string of k words may be measured as the information supplied to the predictor by the text, this measure is given by Eqn.(1)

$$I = \log_2 \frac{1}{p(w|S)} \quad (1)$$

where I= Information to the predictor

w= Word

S=String of k words

p(w|S) = Probability that Word w follows the String S

this is equivalent to Eqn.(2)

$$I = -\log_2 p(w|S) \quad (2)$$

The information supplied by a string of words is equal to the sum of the information supplied by these words separately. For example, $-\log_2 p(xy|S\} = -\log_2 p(x|S) - \log_2 p(x|Sy)$. This means that the information supplied by an entire text can be measured by summing the information provided by each of its words, rather than having to calculate the probability of the text itself occurring. If the predictor was certain that word w would follow the string S, it would assign p(w|S) = 1. If the predictor turned out to be correct, then the information supplied to the predictor would be $-\log_2 1=0$. That is the predictor would not experience any surprise at all.

If the predictor decided that word w couldn't possibly follow the string S, it would assign p(w|S)=0. If word w is found to follow the string, then the information supplied to the predictor would be $-\log_2 0 = \infty$

In this case the predictor is said to be infinitely surprised. Eqn.(3) will measure the performance of a predictor that is presented with a text of N words by summing the information supplied to the predictor by each word in the text. W_j is the j th word in the text.

$$I_{total}=-\sum_{j=1}^{N} \log_2 p(W_j|W_1,W_2,\ldots W_{j-1}) \quad (3)$$

This measure is equal to the surprise received by the predictor when it discovers the contents of the entire text, and is therefore equivalent to Eqn.(4)

$$I_{total}= - \log_2 p(W_1,W_2,\ldots W_N) \quad (4)$$

The total information supplied to the predictor by the text provides a basis for comparing the performance of different prediction algorithms. The total information can be normalized to provide a measure that isn't dependent on the size of the text. This normalized measure gives the average

surprise received by the predictor per word, and is given in Eqn.(5)

$$I = -\frac{1}{N}\sum_{j=1}^{N} \log_2 p(W_j \mid W_1, W_2, .. W_{j-1}) \quad (5)$$

3.SUPPORT VECTOR MACHINES

A Support Vector Machine is a learning algorithm for pattern classification and regression [B. Scholkopf ,1997]. The basic training principle behind SVMs is finding the optimal linear hyperplane such that the expected classification error for unseen test samples is minimized. Support Vector Machines (SVMs) are supervised machine learning algorithm for binary classification on a feature vector space $x \in R^L$ as given in Eqn .(6a and 6b).

$$w \cdot x + b = 0 \quad (6a)$$

$$w \in R^L, b \in R. \quad (6b)$$

Since SVMs are binary classifiers, they have to be extended to multi-class classifiers to predict more than 2 POS tags. Among several methods of multi-class classification for SVMs (Weston et al 1999), one versus-rest approach is applied as shown in Figure-1. In training, k classifiers $f_i(x)$ $(1<=i<=k)$ are created to classify the class i from all other classes as given in Eqn.(6c and 6d).

$$f_i(x) >= +1 \quad x \text{ belongs to the class I} \quad (6c)$$

$$f_i(x) <= -1 \quad \text{otherwise} \quad (6d)$$

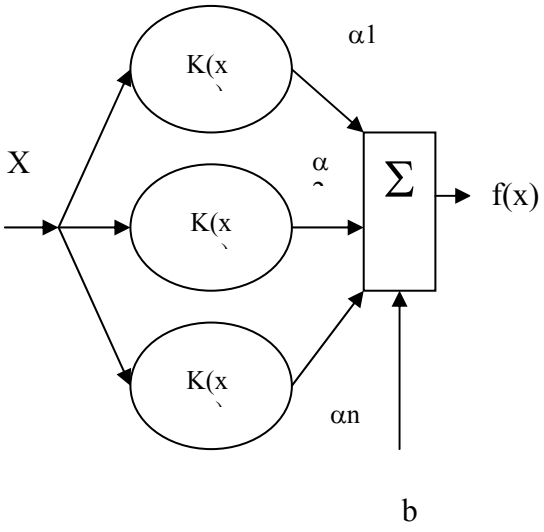

Figure-1 Support Vector Machine

For linearly non-separable cases, feature vectors are mapped into a higher dimensional space by a nonlinear function $\phi(x)$ and linearly separated there. In SVM's, since all data points appear as a form of inner product, we only need the inner product of two points in the higher dimensional space as mentioned in Eqn (7a). Those values are calculated in R^L without mapping to the higher dimensional space by the following function $k(x_i, x_j)$ called a Kernel Function

$$\phi(x_i) \cdot \phi(x_j) = k(x_i x, x_j) \quad (7a)$$

$$k(x_i, x_j) = \exp(-\gamma (x_i - x_j)^2), \quad \gamma > 0 \quad (7b)$$

In this paper the kernel used is the Radial Basis Kernel as given by the Eqn.(7b). The RBF kernel nonlinearly maps samples into a higher dimensional space, so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear. Furthermore, the linear kernel is a special case of RBF kernel and it needs less number of hyper parameters (Keerthi et al, 2003).

$$f(x) = \sum_{i=1}^{M} y_i \alpha_i \cdot k(x, x_i) + b \quad (8)$$

where k(.,.) is a kernel function as depicted in figure-1 and the sign of f(x) determines the membership X as shown in Eqn.(8). Constructing an optimal hyperplane is equivalent to finding all the nonzero α_i. Any vector X_i that corresponds to a nonzero α_i is a *support vector* (SV) of the optimal hyperplane.

4. PROPOSED ALGORITHM

The proposed algorithm for Prediction of the Unknown Words involves four phases namely POS Tagging, Statistical Analysis, Formation of Clusters, Upwriting to higher Level. Part of speech tagging is the problem of identifying parts of speech of words in a presented text. Since words are ambiguous in terms of their parts of speech, the correct part of speech is usually identified from the context the word appears in. The schema of the predictor is given in the Flow Diagram as in Figure-2.

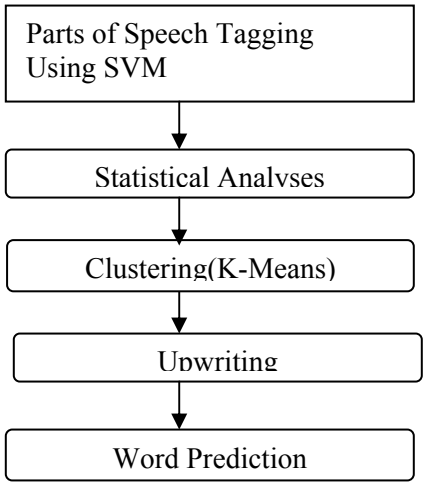

Figure -2 Prediction Process

4.1 Part of Speech Tagging using Support Vector Machines

Experiments for Word Predictor are performed using Penn Treebank Wall Street Journal (WSJ) corpus. Training data set was constructed by randomly selecting approximately 500 sentences. Test data set for word guessing consists of sentences that do not appear in the training data. Lexicon and the Parts of Speech Tagging are given as shown in Table 1. Support vector machines are used to perform the Part of Speech Tagging. SVM classifiers are created for each POS Tag using all words in the WSJ Corpus. Then POS tags of unknown words are predicted using the one versus rest classification scheme.

Table 1. *Parts Of Speech (Categorization Examples)*

AUX(G)	is, having
CC	And
CD	1, three
DT	The
EX	*there* is
IN	in, of, like, after, that
JJ	Green
JJR	Greener
JJS	Greenest
LS	1)
MD	Could, will
NN	Table
PRP$	my, his
PRP	I, he, it
POS	Friend*'s*
PDT	*Both* the boys
NNPS	Vikings
NNP	John
NNS	Tables
NN	Table
RB	However, usually, naturally, here, good
RBR	Better
RBS	Best
RP	Give *up*
TO	*to* go, *to* him
VB	Take
VBD	Took
VBG	Taking
VBN	Taken
WRB	where, when
WP$	Whose
WP	Who, what
WDT	Which
VBZ	Takes
VBP	Take

AUX(G)= auxiliary be, have, CC= coordinating conjunction, CD= cardinal number, NN= noun, singular or mass, MD= modal, LS= list marker, JJS= adjective, superlative, JJR= adjective, comparative, JJ= adjective, IN= preposition /subordinating conjunction, FW= foreign word, EX= existential there, DT= determiner, past participle, past tense, base form, INTJ= interjection, RP=particle, RBS=adverb, superlative, PRP$= possessive pronoun PRP= personal pronoun NNP=proper noun, singular, NNPS= proper noun, plural, PDT= predeterminer, POS= possessive ending, TO=to, VB= verb, VBD= verb, VBG= verb, gerund/present participle, VBN=verb, WRB= wh-adverb, WP$= possessive wh-pronoun, 3rd person sing. Present, WP=wh-pronoun, WDT=wh-determiner, VBZ= verb, VBP= verb, sing. present, non-3d

The Training Corpus is formed by performing Parts Of Speech Tagging to the WSJ Corpus by using Support Vector Machines. Figure-3 shows a Sample Hidden Markov Model that was used to generate a single sentence in the Training Corpus. The Prediction Algorithm will use the inferred grammar to make predictions about the Testing corpus [King Sun Fu et al,1986] and [S.M.Lucas,1993].

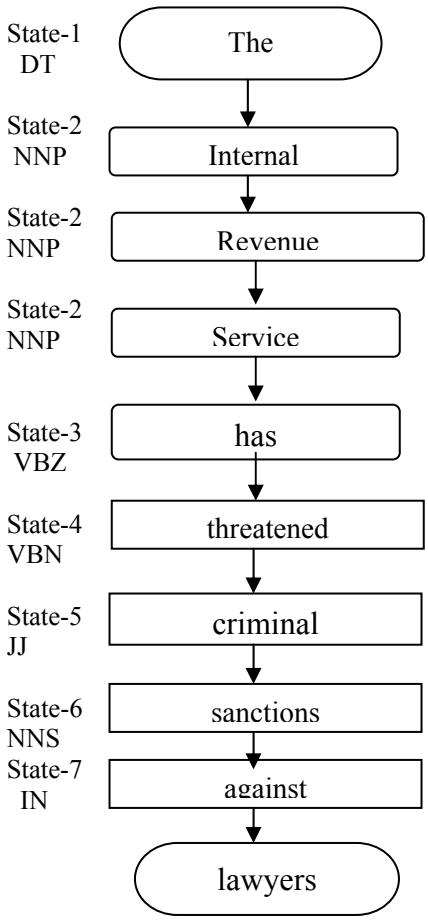

Figure-3 Sample HMM Model

4.2 Statistical Analysis

Strings of three sequential words taken from the training corpus are divided into groups of strings that share the same first two words. The Third word is used to determine a count vector for each group of Strings. This process gives the count vectors of various groups. The elements of all the vectors are sorted alphabetically, such that the leftmost element corresponds to the String with starting letter a in the Training corpus. The Count Vectors are normalized to produce the Frequency vectors .The Frequency vectors are approximations of the conditional probability distribution of sequent words given the context. This provides the second order Markov Model for the text. The trigram, bigram and unigram Statistics required by the Prediction algorithm are also collected in a similar fashion.

4.3 Clustering

K-clustering algorithm is used to find the groups of similar frequency vectors. The algorithm used for finding the cluster centroids is given below.

Algorithm

1. Begin by assigning the initial position of each centroid to frequency vectors chosen at random.
2. A frequency Vector f_i is selected, and its distance from each centroids is calculated using the Euclidean Metric Eqn .(9)

$$D_j = \sqrt{\sum_{i=1}^{A} (C_j[i] - f_i[i])^2} \quad (9)$$

where C_j is the Centroid of j^{th} cluster
f_i is the i^{th} frequency vector
D_j is the Distance
A is Number of Cluster Centroids

3. The closest Centroid is found and its position is updated if it is within some user defined distance from the frequency vector.
4. Assuming that the centroids has come closest to K data vectors in the past, its new position will be calculated as the mean of all vectors it has come closest to as shown in Eqn .(10)

$$C_j' = (K\, C_j + f_j) / K+1 \quad (10)$$

5. This process is iterated until the centroids remain at a fairly constant position, at which stage each data vector is assigned to the cluster corresponding to the closest centroid, with no assignment being made when this distance exceeds the defined cluster radius.

4.4 Upwriting

The text is upwritten by taking a string of two words from the text and upwriting the next word to the corresponding cluster. This process will be done on the entire testing corpus, which will give the upwritten text. The final grammar may be considered to be a compressed version of the training text, atleast an approximation to it [J.Hutchens,1994] and [J.L.Hutchens,1995].

The first word of the text cannot be upwritten since it is the beginning of the text and current context cannot be understood since it has not seen any words previously. If a new word occurs which

is not in the training corpus, in this situation also upwriting will not be possible.

4.5 Prediction

The Predictor makes use of the hierarchical grammar inferred from the upwriter [Allan Ramsay 1993] and [Jelinek.F,1985]. Starting at the highest level of the grammar, contextual information is used to predict a probability distribution for the next symbol in much the same way as the IBM fallback predictor does. The predictor will eventually downwrite to the lowest level, giving the prediction in the form of a probability distribution over the alphabet. The prediction vector $G_{i,j}$ as shown in Eqn.(11) is calculated by taking the weighted sum of all frequency vectors.

$$G_{i,j} = \sum_{m=1}^{A} \sum_{n=1}^{A} p(C_i [m] C_j [n]) f_{im,jn} \quad (11)$$

where A-Number of Cluster Centroids
$f_{im,jn}$ - Frequency Vector
C_i - Cluster Centroids

Justification is that it takes more information into account than the interpolated IBM Predictor. The upwrite process may allow words which occurred long before the current trigram have an effect on the prediction.

5.SAMPLE OUTPUT

(Wall Street Journal Corpus)

Zenith Data Systems Corp., a subsidiary of Zenith Electronics Corp., received a $534 million Navy contract for software and services of microcomputers over an 84-month period.

Parts OF Speech Tagging with SVM

[Zenith/NNP Data/NNP Systems / NNPS Corp./NNP] ,/, [a/DT subsidiary / NN] of / IN [Zenith / NNP Electronics/NNP Corp./NNP] ,/, received/VBD [a/DT $/$ 534 /CD million/CD Navy/NNP contract/NN] for/IN [software/NN] and / CC [services / NNS] of/IN [microcomputers /NNS] over/IN [an/DT 84-month/JJ period/NN] ./.

Grammar Generated from the Corpus After Upwriting

[Zenith/NNP Electronics/NNP Corp./NNP,1] ./.,1 Electronics/NNP Corp./NNP],1[subsidiary/NN] of/IN,1[,84-month/JJ period/NN],1534/CD million/CD Navy/NNP,1,/, [a/DT,1 [,1million/CD Navy/NNP contract/NN,1over/IN [an/DT,1data/NNS ,/, received/VBD [,1received/VBD [a/DT,1 for/IN [,1Systems/NNPS Corp./NNP],1[services/NNS],1of/IN [,1[software/NN],1 microcomputers/NNS] over/IN,1 [an/DT 84-month/JJ,1 [a/DT,1period/NN] an/DT 84-month/JJ period/NN,1] was/VBD given/VBN,1] over/IN [,1[targeting/VBG|NN equipment/NN,1 software/NN] and/CC,1 services/NNS] of/IN,1a/DT subsidiary/NN],1],1[a/DT $/$,0.8] ./. [,0.75] ,/, [,0.5 and/CC ,/, received/VBD,0.5 [Zenith/NNP Data/NNP,0.5 [Zenith/NNP Electronics/NNP,0.5 of/IN [microcomputers/NNS,0.5 and/CC [services/NNS,0.5 of/IN [Zenith/NNP,0.5 Corp./NNP] ,/,,0.4

6.PERFORMANCE COMPARISON RESULTS

The results shown in Table 2 are obtained when the predictor is executed on the testing corpus. The fact that the proposed predictor experienced less infinite surprises than the other existing predictors is indicative of the improvement of the language model. Similarly the fact that the fallback was not required in these cases or required less often generally is a similar measure of generality. The results indicate that the proposed Predictor copes better with sparse data as available with WSJ corpus, and is able to generalize its observations to cater for unseen trigrams.

Table 2 ***Performance Comparison for 50 Trials (Average)***

Algorithm	- Ī	**Fall back**
IBM Predictor	3.13	5
Q-Predictor	2.25	3
Clustered QPredictor	2.15	-
Proposed Algorithm	1.53	-

7. DISCUSSION

The Application of Support Vector Machine for Parts of Speech Tagging has resulted in the better results. We extend the existing works by using support vector machines to induce better

grammars. We would like to pursue further research on grammar inference in a manner that not only syntax but also semantics can be inferred from a given set of data.

8. REFERENCES

[1]. Venkatagiri, H. *Efficiency of Lexical Prediction as a Communication Acceleration Technique*. Augmentative and Alternative Communication, 9: 161-167, 1993.

[2]. Noam Chomsky. *Lectures in Government and Binding*, Foris Publications, 1981.

[3]. Jeffrey.L.Elman. *Distributed representations, simple recurrent networks and grammatical structures,* Machine learning , 7(2/3) :195-226, 1991

[4]. King Sun Fu and Taylor.L.Booth *Grammatical inference, Introduction and survey part 2,* IEEE Transaction on Pattern Analysis and Machine Intelligence 8(3), May 1986.

[5]. S.M.Lucas. *New directions in Grammatical Inference*, IEE colloquium on grammatical inference:theory applications and alternatives, 1:1-7, April 1993.

[6]. R.Weischedel, N.Meteer, R,Schwartz, L.Ramshaw and J.Palmucci. Coping with Ambiguity and Unknown words through Probabilistic Models. Computational Linguistics, 19(2): 359-382, 1993.

[7]. Allan Ramsay. *Inference in language processing*, IEE colloquium on grammatical inference: Theory Applications and Alternatives, 5:1-3, April 1993.

[8]. Jelinek, F. *The Development of an Experimental Discrete Dictation Recognizer*. Proc. IEEE, 73 (11): 1616-1624, 1985.

[9]. J. Hutchens, M. Alder, and Y. Attikiouzel. *Natural language grammatical inference*. Technical Report HT94-03, University of Western Australia, 1994.

[10]. J L Hutchens. *Grammatical Inference and The Upwrite Predictor*. Ph.D. report, July 1995.

[11]. MacArthur, S.Graham, J.A.Haynes, S.De La Paz, Spelling checkers and student with learning disabilities: Performance comparisons and impact on spelling. Journal of Special Education, 30: 35-57 ,1996

[12]. Margaret Ayciena et al. *An Evolutionary Approach to Natural Language Grammatical Inference*, The MIT Press, Cambridge, 2003

[13]. B. Scholkopf. *Support Vector Learning*. PhD thesis, Technical University of Berlin, 1997

[14]. Jesus Gimenez and Lluis Marquez. SVMTool: A general POS tagger generator based on Support Vector Machines Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04). Lisbon, Portugal, 2004.

[15]. T Brants . TnT – A statistical Part of Speech Tagger In Proceedings of the 6th Applied NLP Conference (ANLP-2000) :224-231, 2000

[16]. E Brill. Transformation –Based Error Driven Learning and Natural Language Processing: A case study in Part of Speech Tagging. Computational Linguistics, 21(4): 543-565, 1995

[17]. T. Nakgawa, T.Kudoh, Y. Matsumoto. Unknown Word Guessing and Part of Speech Tagging Using Support Vector Machines, In the Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium, 2001

[18]. A.Ratnaparkhi. A maximum Entropy model for Part of Speech Tagging. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP -1), 133-142, 1996.

[19]. J.Weston and C.Watkins. Support Vector Machines for Multi-class Pattern Recognition . In Proceedings of the Seventh European Symposium on Artificial Neural Networks(ES ANN-99), 1999.

[20]. Keerthi S.S and C.J.Lin. Asymptotic behaviors of Support Vector Machines with Gaussian Kernel, Neural Computation 15(7) :1667-1689, 2003.

[21]. C.E.Shannon. *Prediction and entropy of Printed English*, Bell Systems Technical Journal, 30: 50-64, 1951.

[22]. Marquez L., Padro L., Rodriguez H. A Machine Learning Approach to POS Tagging. Kluwer Academic Publishers, Boston, 1-34, 2000