Proc. 2nd Annual Conference of the International Technology Alliance, ACITA 2008, London, UK, September, 2008

A Grammar Inference Algorithm for Event Recognition in Sensor Networks

Sahin Cem Geyik and Boleslaw K. Szymanski
Department of Computer Science and Center for Pervasive Computing and Networking
Rensselaer Polytechnic Institute, Troy, New York 12180
{geyiks,szymansk}@cs.rpi.edu

***Abstract*—In this paper, we demonstrate that probabilistic context free grammars (PCFGs) can be used to recognize events from a sensor data stream. A fast PCFG inference algorithm based on Stolcke(1994) and Chen(1996) is presented which utilizes the set of observation strings as training data. A real-world scenario is presented and we also show that multi-modal sensor information can be utilized using Dempster-Shafer theory of evidence.**

I. Introduction and Previous Work

Today's sensor applications collect vast amount of measurement data, so it is increasingly important to provide their users with high level representation of those measurements. The focus of this paper is on using Probabilistic Context Free Grammars (PCFGs) for event recognition. A PCFG is a context free grammar in which each production rule is assigned a probability defined by the frequency with which it appears among all nonterminal occurrences on the right hand sides of all productions. Examples of applications of PCFG to sensor networks data are presented in [3], [4], [5]. We introduce here a grammar induction method inspired by the scheme developed by Stolcke [1]. The novelty of our approach lies in defining the new evaluation metric for the grammar together with the method for fast calculation of this metric.

Literature on learning PCFG from training data currently focuses on two directions. One is about learning the rule probabilities when the general model for the grammar is already decided [6]. Another direction, which we also follow, detects the model (terminals, nonterminals and rules) as well as the probability distribution on productions [1,2].

II. Grammar Inference

This section explains the method initially introduced by Stolcke [1]. Grammar construction consists of two steps: sample incorporation and application of operators.

1) Sample Incorporation: Sample incorporation is the initial step of constructing the grammar from training data. Each sentence in the training data consists of a string of terminal symbols. Each of the terminal symbols introduces a new nonterminal and each sentence appears as a production rule of the *START* nonterminal.

2) Operators: We are using two operators: merge and chunk [1] for building up the grammar step by step.

Merging takes two nonterminals and reduces them into a new nonterminal. Right hand side occurrences of these two nonterminals are replaced by the new nonterminal which inherits the rules of these nonterminals as well as their counts.

The second operator, *chunking*, creates a new nonterminal with a single rule: a string composed of nonterminals in the current grammar. It simply replaces all occurrences of this string with the new nonterminal. Count of the new nonterminal rule is the number of times this replacement takes place.

We attempt to maximize a posteriori probability of the grammar G given the training data O.

$$G = argmax_G P(G|O) = argmax_G P(G) * P(O|G) \quad (1)$$

$P(G)$ (grammar's a priori probability) is inversely related to the length of the grammar, $l(G)$, and $P(O|G)$ is calculated as product of probabilities of all sentences in training data.

$$P(G) = 2^{-l(G)}$$
$$P(O|G) = \prod_{i=1}^{|O|} p(o_i|G) \quad (2)$$

Each nonterminal in the grammar increases its length, $l(G)$, by two (one accounts for the nonterminal name and other for a separation symbol). Each rule in the nonterminal, increases $l(G)$ by $2 + (\ \#\ of\ nonterminals\ in\ the\ rule)$, where 2 accounts for the separation symbol and the count value.

A. Computation of the Evaluation Metric

1) Chunking: Calculating $P(G)$ is easy given the modified grammar. Also, $P(O|G)$ does not change by the chunking operation.

2) Merging: Calculating $P(G)$ is again easy given the modified grammar. However, $P(O|G)$ changes whenever merging takes place during modification of the grammar. Instead of going through the training data however, $P(O|G)$ can be calculated using counts in the grammar. Fig.1 shows an example.

B. Inference Algorithm

Our implementation searches for a pair of of nonterminals to merge that yields the highest a posteriori value advancement.

For chunking operation, we search for chunks of up to size *7*. This number is chosen so that we can find a chunk that diminishes the grammar, if there is one. Each time a

X1 → a b c (3) | a b d (4)

X2 → a b c (2) | a g b (3)   :: (Merge X1 & X2) ::   M → a bc (5) | a d e (4) | a g b (3)

$$P(O|G)_{new} = \underbrace{[(5/12)/(3/7)]^3}_{\text{"a b c" in X1}} * \underbrace{[(5/12)/(2/5)]^2}_{\text{"a b c" in X2}} * \underbrace{[(4/12)/(4/7)]^4}_{\text{"a d e" in X1}} * \underbrace{[(3/12)/(3/5)]^3}_{\text{"a g b" in X2}} * P(O|G)_{previous}$$

Fig. 1. Calculating $P(O|G)$ for Merging Operation

replacement of a chunk with k nonterminals occurs, grammar gets shorter by *k-1* symbols as it replaces k symbols with a single symbol, the name of the new chunking nonterminal. At the same time, introducing a chunk of size k creates a new nonterminal with one rule which elongates the grammar by *k+4*. Hence, for the replacement of a chunk of repeating symbols to decrease the length of the grammar, the chunk must be of length of at least *7*. Therefore, if no chunk of such size ever repeats in the grammar, then there is no advantage in applying chunking operation.

In most cases, merging operation decreases a posteriori probability according to definition of $P(O|G)$ given by Eq. (2). When, during merging, we replace a production with small number of alternatives by the one with larger number of alternatives (merge in Fig.1 shows an example of such replacement), the advantage of the shorter grammar length usually does not make up for the decrease in $P(O|G)$. In contrast, chunking operation always decreases the grammar length and increases its a posteriori probability. Therefore, we expect that advantageous merging operations will be less likely to encounter than advantageous chunking operations. Consequently, it is beneficial for a posteriori probability of the grammar to execute as many chunking operations as possible before applying any merge operations. Hence, the general form of our interference steps is

$$chunk^n \ merge \text{ where } n \geq 2.$$

III. Real-world Scenario

Fig.2 shows a tracking example in which we gather terminal symbols from the camera images. A grammar can be trained by using several different examples for different possible events, such as parking, leaving, unsuccessful search, etc.

Fig. 2. Parking Lot Car Tracking for Event Detection

A. Multi-modal Sensor Information

Data collected by multiple sensors can be processed utilizing Dempster-Shafer theory of evidence based on belief functions and plausible reasoning [7,8].

Suppose we can detect two events a and b. Then all different events that can be detected are members of the power set $2^{a \cup b}$. Dempster-Shafer Theory's mass function assigns a probability to each set in the power set 2^X of all events $X = \{x_1, x_2, \ldots, x_n\}$ in such a way that all probabilities sum up to the total mass, m =1. If we have two independent sets of events, X and Y, with different mass functions, m_1 and m_2 respectively, then a new mass function m_{12} over the power set of the set of all combined events $F = X \times Y$ is defined by the Dempster's combination that assigns to any nonempty subset $F_s \subset 2^F$ the following mass:

$$m_{12}(F_s) = \frac{\sum_{X_s \cap Y_s = F} m_1(X_s) * m_2(Y_s)}{1 - \sum_{X_s \cap Y_s = \emptyset} m_1(X_s) * m_2(Y_s)}. \quad (3)$$

IV. Future Work

We are currently working on experiment settings which will provide us with actual cases for measuring effectiveness of this learning method for sensor network applications. We are also working on further improvements to the PCFG inference algorithm.

Acknowledgment

This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] Stolcke, A., Omohundro, S., *Inducing Probabilistic Grammars by Bayesian Model Merging*, Proceedings of the Second International Colloquium on Grammatical Inference and Applications, Pages: 106-118, 1994.

[2] S. F. Chen (1996), *Building Probabilistic Models for Natural Language*, Doctoral dissertation, Dept. of Computer Science, Harvard University.

[3] Lymberopoulos, D., Ogale, A.S., Savvides, A., Aloimonos, Y., *A Sensory Grammar for Inferring Behaviors in Sensor Networks*, The Fifth International Conference on Information Processing in Sensor Networks, 2006. IPSN 2006.

[4] Ivanov, Y.A., Bobick, A.F., *Recognition of visual activities and interactions by stochastic parsing*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, Volume 22, Issue 8, Aug. 2000, Page(s):852-872.

[5] Darnell Moore and Irfan Essa, *Recognizing multitasked activities from video using stochastic context-free grammar*, In Proceedings of AAAI-02, 2002.

[6] K. Lari and S.J. Young. 1990. *The estimation of Stochastic Context-Free Grammars using the Inside-Outside algorithm*, Computer Speech and Language, 4(35-56).

[7] Dempster, A. P., *A generalization of Bayesian inference*, Journal of the Royal Statistical Society, Series B 30 205-247, 1968.

[8] Shafer, G., *Perspectives on the theory and practice of belief functions*, International Journal of Approximate Reasoning 3 1-40, 1990.