Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 28, S. 545-547, Kasım 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 28, pp. 545-547, November 2021

Research Article

The Paradigm for Solving the Derivation Problem in Infinite Models

Mehmet Kurucan [1*], Mete Özbaltan [2]

[1*] Ardahan University, Faculty of Engineering, Departmant of Computer Engineering, Ardahan, Turkey, (ORCID: 0000-0003-4359-3726), mehmetkurucan@ardahan.edu.tr

[2] Erzurum Technical University, Faculty of Engineering, Department of Computer Engineering, Erzurum, Turkey, (ORCID: 0000-0002-3215-6363), mete.ozbaltan@erzurum.edu.tr

(1st International Conference on Applied Engineering and Natural Sciences ICAENS 2021, November 1-3, 2021)

(**DOI:** 10.31590/ejosat.1008716)

ATIF/REFERENCE: Kurucan, M., Özbaltan, M. (2021). The Paradigm for Solving the Derivation Problem in Infinite Models. *European Journal of Science and Technology*, (28), 545-547.

Abstract

The problem of finding the most probable derivation for probabilistic context-free grammar is expensive. The Viterbi algorithm has been adopted to one-counter grammar that is a sub-class of stochastic context-free grammar to solve this issue. However, the absence of the implementation of the adapted algorithm attracts our attention. We experimentally validate this algorithm and present the implementation part of it to monitor the performance, in this research.

Keywords: Probabilistic Context-Free Grammars, Viterbi Algorithm, One-Counter Grammar, Hidden Markov Model, Formal Languages.

Sonsuz Modellerde Türetme Problemini Çözme Uygulaması

Öz

Olasılıksal bağlamdan bağımsız dilbilgisi için en olası türetmeyi bulma probleminin çözümü pahalıdır. Bu problemin çözümü için Viterbi algoritması, olasılıksal bağlamdan bağımsız dilbilgisinin bir alt sınıfı olan tek sayaçlı dilbilgisine uayarlanmıştır. Ancak adapte edilen algoritmanin uygulanmamış olması dikkatimizi çekmektedir. Bu araştırmada, bu algoritmayı deneysel olarak doğruladık ve uygulamada izlenilen performansı sunuyoruz.

Anahtar Kelimeler: Olasılıksal Bağlamdan Bağımsız Dilbilgisi, Viterbi Algoritması, Tek-Sayaçlı Dilbilgisi, Saklı Markov Model, Biçimsel Diller.

[*] Corresponding Author: mehmetkurucan@ardahan.edu.tr

1. Introduction

Grammar is a useful tool for studying languages. It can be used to generate a complex language via all valid strings [1]. Stochastic Context-free Grammar (SCFG) is a notable proper syntax that give different production rules to be utilized in the conventional language fields [3]. The rules in the syntax can be applied paying little mind to framework. That implies the language is produced by rehashed the standard applications and their sentences are not impacted by the unique circumstance.

The derivation problem in natural languages is an important issue. The derivation (or finding most probable derive) of the stochastic context-free language and its subclasses, which is generated by using SCFG, is computed in cubic time complexity. CYK algorithm is used to solve the derivation task of SCFG [4]. However, this algorithm is expensive for the applications of natural language processing.

The circumstance of this issue attempted to be solved in linear time by adopting a dynamic programming algorithm which is called Viterbi [2]. This algorithm is used for decoding problem of a Hidden Markov Model (HMM) [5]. The inference of the most likely derivation is settled in linear time of the size of the contribution for these languages by the Viterbi calculation [6].

This work [2] adopted this dynamic programming algorithm to lessen the time complexity of derivation problem of stochastic context-free languages (SCFL) from cubic to liner time level. They focused on one-counter language (OCL) which is an important subclass of SCFL to solve this issue. However, the study was only considered theoretically, the implementation was not included.

It is well-known fact that the language processing is an important issue; in this manner the application part is also very important. As it well known, more often than not, such theoretical works are not validated unless they are implemented. In this study, we examine this subject and show the power of the adopted algorithm by completing the missing implementation part of [2].

The rest of this work is organized as follow. Section II gives definition of the Viterbi and probabilistic one-counter model. In Section III, we represent the implementation code of the adopted Viterbi algorithm and results. In Section IV, we conclude the whole paper.

2. Definitions

In this section, we briefly give simple descriptions of the Viterbi algorithm and probabilistic one-counter systems with their languages that are produced through one-counter grammar to ensure the readers comprehend the paper without any problem

2.1. The Viterbi Algorithm

The instructional paper [7] presented three fundamental problems for HMMs. One of them is the interpreting issue that in regards to finding the best derivation which clarifies the given output sequence. The Viterbi calculation is utilized to solve this issue in HMMs. It is officially characterized as

$$v_t(q) = max_{S_1 \dots S_{t-1}} Pr(S_1 \dots S_{t-1}, o_1 \dots o_t, S_t = q) \quad (1)$$

In here, $v_t(q)$ denotes the maxımum likelihood value of hidden state q after passing through $S_1 \dots S_{t-1}$.

2.2. Probabilistic One-Counter Systems

The simple definition of one-counter model can be represented as it is a subclass of probabilistic push-down model [8]. Literally a pushdown system is providing an infinite input size due to it consists a stack. For each transition process a stack symbol either is pushed into the stack or popped from the stack. Thus, the system provides a powerful infinite state model.

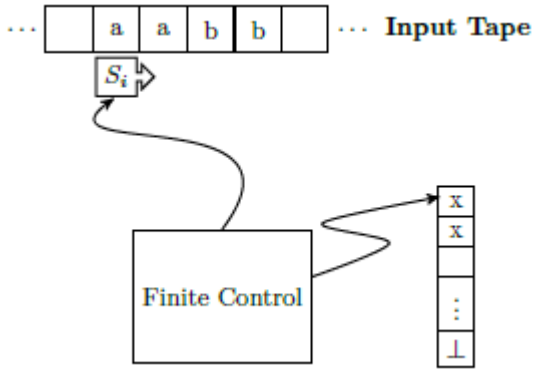

Fig. 1 The structure of one-counter model

The structure of one-counter model is very similar to pushdown model but it contains only one stack symbol. The number of the stack symbol represents the counter. As shown in Fig. 1, there are three components: input tape, a control unit (i.e.S_i) and a stack. The counter (i.e. the number of the stack symbol) can be incremented or decremented depending on the transitions (i.e. readıng one symbol from the input tape). There are also two transition functions δ^0 and δ^+) which are enabled depeding on the counter value eqauls to zero or not.

2.3. Methodology

According to [2], the Viterbi algorithm was adopted to probabilistic one-counter model for derivation issue in stochastic context-free languages. Let attempt to clarify the methodology part of this work without dive into too many details. Let S represents the finite control state, A denotes alphabet, $F \subset S$ is particular final state. According to [2], a transition form is

$$s, a, \delta \rightarrow r, k$$

where s and r are control states, a is alphabet, $\delta \in \{0,1\}$, and $k \in \{-1,0,1\}$. If $\delta = 0$ then δ^0 is enabled. The counter value can be either same or incremented after a particular transition. If $\delta = 1$ then the transition function δ^+ is enabled. The counter changes can be as unchanged, increased, or decremented after the transition. Each transition is related with a probability value. Thus, the sum of the probability of each transition must equal to one. At the initial step, the counter value is zero. The given input tape is accepted if and only the machine is in the final state with zero counter value after the last symbol of the input tape is perused.

The adopted version of the Viterbi algorithm takes an input tape as input and it gives the probability value of the most probable derivation, state and counter sequences as *output*.

3. Results

In this part, we analyse how adopted Viterbi is executed according to our perspective. First and foremost, we create the dataset which corresponds to a probabilistic one-counter framework. We produced diverse datasets relying upon the quantity of the control states and input symbols. Then, at that point, we made our own pseudocode as opposed to utilizing it from [2].

1) Initialization
Read *dataset*
Make *init*
for (*sequence* ∈ *sequenceList*)
2) Finding Viterbi Values
for ($t \in T$)
for ($s \in$ *stateList*)
for ($c \in c$+k)
for (*viterbiValue* ∈ δ[*t-1*,*s*,*c*]) ($t = 0$? 1 : cont.)
if *maxViterbiValue* < *viterbiValue*
maxViterbiValue = *viterbiValue*
δ[*t*,*s*,*c*] = *maxViterbiValue*
3) Finding State Sequence Traceback
for (*t=T;t* ∈ *T;t-1*) ($t = T$? s = *finalS* & c = *finalC* : cont.)
prob = $max_{prob}(\delta)$
s' = $max_{s}(\delta)$
c' = $max_{c}(\delta)$
stb[*t*] = [*s',c',prob*]

Fig. 2 Pseudocode of the adopted Viterbi algorithm

Fig. 2 shows the pseudocode of the adopted Viterbi algorithm, where: *dataset* includes all the given dataset eg., observation sequences and the number of states and symbols in the given sequences; and *init* creates the zero matrices used in the program. The first loop executes the Viterbi algorithm for each sequence among all sequences in a given dataset. Latter, three nested loops find the maximum Viterbi value for the trio *t*, *s*, *c*, where they denote the current time, state, and counter respectively, and then δ identically stores it. Then, the sequence traceback *stb* holds the maximum Viterbi value *prob* with the state *s'* and counter *c'* pair, with the direction from the final state towards the starting state.

Table 1. KL Diversion Comparison of adopted Viterbi algorithm and CYK algorithm

Input	Adopted Viterbi	Original Viterbi
D:20;L:20;S:2;O:2	0,2509	46,5618
D:50;L:50;S:2;O:2	0,5956	81,0225
D:100;L:100;S:2;O:2	1,4965	187,4625
D:20;L:20;S:3;O:3	0,1827	26,7187
D:50;L:50;S:3;O:3	0,4732	35,4326
D:100;L:100;S:3;O:3	0,2488	42,2428
D:20;L:20;S:4;O:4	0,2501	15,6061
D:50;L:50;S:4;O:4	0,3276	25,0568
D:100;L:100;S:4;O:4	0,1083	6,3593

Eventually, we thought about the power of adopted Viterbi against to unique Viterbi calculation. Yet, remind that the counter value is consistently zero when the original Viterbi proceeds. We utilized Kullback-Leibler (KL) divergence to address the outcomes.

Table 1 shows the KL diversion comparison of the adopted Viterbi and original Viterbi algorithms; where the header input in the table indicates that: D is the dimension; L is the length; S is the number of states; and O is the number of observations. As the outcomes in the table show, the KL values of the model nearest to the first model are more like zero. Along these lines, as per the outcomes, the adopted Viterbi calculation is utilized as a dynamic programming algorithm in the derivation problem of languages delivered by a one-counter grammar.

4. Conclusion

In this paper, we initially clarify an alternative calculation that is utilized to take care of the derivation issue in the stochastic context-free languages is done in [2]. That work focused on the issue only theoretically. Yet, the implementation part has a significant spot while working on language processing. We filled this gap by executing the adopted calculation and show and validate its force against the original. As per the accomplished outcomes, the embraced rendition of the Viterbi calculation can be utilized as a dynamic programming calculation to take care of the inference issue of stochastic context-free languages by decreasing the calculation cost.

References

[1] C. D. Manning and H. Schutze, *Foundation of Statistical Natural Languages* MIT Press, Cambridge, MA, USA, 1999.

[2] A. Sakharov and T. Sakharov, "The Viterbi algorithm for subsets of stochastic context-free languages", *Information Processing Letters*., vol. 135, pp. 68-72, Jul. 2018.

[3] J. Autebert, J. Berstel and L. Boasson, *Context-free Languages and Pushdown Automata in: Handbook of Formal Languages*, Springer,1997.

[4] J. C. Chappelier, and M. Rajman, "A generalized cyk algorithm for parsing stochastic cfg," in *Proc.TAPD'98*, 1998, p. 133.

[5] A. J. Viterbi, "A personal history of the Viterbi algorithm," *IEEE Signal Process*., vol. 4, pp. 120, 2006.

[6] B. Brejova, D. G. Brown and T. Vinar, "Advances in hidden Markov models for sequence annotation", *Bioinformatics Algorithm: Techniques and Application*, vol. 3, pp. 55-92, 2008.

[7] L. R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Morgan Kaufmann Publishers Inc*., 1990, p. 267.

[8] K Etessami, D. Wojtczak and M. Yannakakis, "Quasi-birth-death processes, tree-like qbds, probabilistic 1-counter automata, and pushdown system," in *QEST'08*, 2008, p. 243.