Framework for Data Mining of Big Data Using Probabilistic Grammars

Aljoharah Algwaiz*, Reda Ammar†, Sanguthevar Rajasekaran‡
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT, USA
Emails: aljoharah.algwaiz@uconn.edu, reda@engr.ucon.edu, rajasek@engr.uconn.edu

***Abstract*—Most commonly used and researched data mining approaches use neural and statistical methods to extract information for large data sources. This paper proposes a framework for a novel approach in data mining using Probabilistic Context Free Grammar (PCFG). The framework is proposed for two distributed data structures; dependent and independent. *Escherichia Coli* promoter DNA sequences dataset are chosen as a case study in this paper.**

I. Introduction

Data Mining is defined as searching large volumes of data for significant information [10]. The big data growth explosion created the need for an efficient data management and extraction techniques. Most data mining research topics in the field focused on neural network through developing a learning system based recognizer to mine data. Instead, this paper conducts data mining in a different approach through applying Probabilistic Context-Free Grammar (PCFG) on distributed data sources. Formal grammars provide structural and statistical knowledge of data. Grammars are used to compress big data and simplify it into an easy to understand data. Identifying the data source is the first step to select a data mining technique. Here, data source will consist of DNA Escherichia coli promoter sequences (also called E. Coli promoter). As a case study, the paper will tackle two types of multi-source data mining. The first data type is independent data sources. Independent data sources are data sources that store all data of a single instance in one data source. Data search and extraction in this distributed data structure is usually done orthogonally. The second type of multi-source data mining is dependent data sources. It is when parts of an instance can have different data in various data storages but each is attribute independent. Data search and extraction in dependent distributed data structures are more complex and are usually done in a horizontal manner. The remainder of this paper will further explain general framework and approach. The next sections in this paper will be as follows; section II is work related work to this paper. Section III, covers grammars. A definition of data mining and different structures related to this paper is covered in section IV. Section V shows the different steps to infer a grammar to be used in data mining. Then, the paper shows how to use the inferred grammar to data mine in section VI. Section VII, provides a case study of this framework by using an Escherichia Coli promoter DNA sequences. Finally, in section VIII will conclude this paper and explain future work.

II. Related Work

With the recent vast amount of data availability, data mining big data became an interesting topic for various researchers. Most proposed data mining techniques are based on neural and statistical approaches. Some researchers proposed grammatical inference approaches to mine data. A paper published by Borges and Levene propose a hypertext probabilistic grammar with the use of kth-order Markov models. It learns the user web navigation history and generates probabilities to predict the next page the user is going to. The probabilities are generated where higher probability corresponds to the users preferred trail [7]. Their algorithm mines the trail and uses a model where the last N pages visited will effect which page will be visited next. This approach is based on collecting historical data before generating knowledge. The approach proposed in this paper studies the structure of data instead of historical data. Therefore, no need for previous knowledge or data collection time to infer grammar.

III. Grammars

Grammatical inference (also called grammatical learning or induction) is the study of finding the set of rules when given a finite sample of infinite elements [4]. An inferred grammar can be seen as a set of rules that is generated to describe the overall language. Inferred grammar represents, not only the statistical properties, but also the structural relations of data [1] [2]. Grammars can also assist in pattern recognition or regenerate an infinite set of sentences that belongs to a language [2]. Some of advantages of using formal grammar to data mine are:

- Data description. Data mining using grammars can help identify recurring patterns and trends in the given data. An example can been seen in a research conducted by Borges and Levene where they used grammatical inference to data mine users web-navigation patterns. After

978-1-4673-9431-4/15 $31.00 © 2015 IEEE
DOI 10.1109/ECONF.2015.50

analyzing using probabilistic context-free grammar, they were able to identify the users navigation patterns and predict the next page the user will most likely visit.[7]

- Data classification. Given big data, classifying the large data source into sub-groups with common features, where each sub-group has its own grammar, will help the investigator simplify and further analyze a subgroup of interest.
- Association. Given a string, the grammar can evaluate and analyze if it belongs to the data set or reject it otherwise. After inferring a grammar for a data, if the grammar was properly inferred, the production rules should be able to reproduce all the data. For instance, given a number of grammars for different functional genes (e.g. human promoter, operator, and terminator), the algorithm should be able to identify if this sequence can be reproduced by any of the grammars. If the promoter grammar regenerates the same sequence, then it is said that the given sample is a promoter DNA sequence. If not, other grammars are tested or the sequence rejected.

A language can be defined as a finite or infinite set of strings where each string is a combination of symbols from the alphabet. Languages of significance do not consist of arbitrary set of strings. Languages usually contain strings that follow a certain form, or what might be called a certain grammatical structure. Grammars, in general, are used to describe natural language syntax through simple replacement rules and transformations. However, they are not limited to natural languages. Grammatical inference has been research in various fields including machine learning, computational learning theory, pattern recognition, computational linguistics, and many others[6].Grammars can represent the syntax of languages or the structural relations of patterns of data. It is a syntactic source that can characterize the data and can be used to regenerate all finite and infinite sentences in a language. It can also characterize the patterns belonging to a specific call or set [2]. There are various types of grammatical probabilistic and deterministic methods surveyed by Booth and Fu [2]. Different types of data require different types of grammars.

Grammatical inference is a technique that generates a model that characterizes a set of strings. A formal grammar has four tuples.

G= <T, N, R, S>where
T is a finite set of terminal symbols
N is a finite set of non-terminal symbols
R is a finite set of production rules (also called rewrite rules)
S is the starting symbol $\in$ N (N and T are disjoint)

If X is the string generated by the grammar G, then $S \overset{*}{\Rightarrow} X$. starting from the starting symbol S, applying the production rules R on S will form the string X. A set of strings will be generated from the productions rules which is defined as the language L(G) where
$L(G) = \{ X \in T^* \mid S \overset{*}{\Rightarrow} X\}$.
Different set of operation are used in grammatical regular expression such as union ($\cup$), Keleen star (*), and concatenation (designated by juxtaposition).

1) Union is when there is two separate sets {a} and {b}. Then, {a}$\cup${b}= {a,b}.
2) Concatenation of two languages means appending each string with the string from the second language together to form a single string such as {a}{b}={ab}
3) When Keleen star is used, it means the set contains zero or more of concatenated symbol. For instance, L(a*)= $\{\lambda$, a, aa, aaa, aaaa, .}

Different types of grammars are used for different types of data types. Grammars differentiate based on their rewrite rules. The collection of all strings that can be generated from the grammar is called the language L(G). Finite state grammars and context free grammars are the two most used grammar classes used in computer science [1].

A. Finite-State Grammar

Chomsky defines a finite-state grammar as the process of a finite-state machine producing a set of symbols while moving from one state to the other, which is also called a sentence [8]. The set of sentences produced by a finite-state grammar is called a finite-state language. Finite-state grammar has the most restrictions and defines the most limited class of languages. A grammar G is defined as a finite-state grammar if and only if the grammars production rules R are in the format A $\rightarrow$ b where A $\in$ N and either b =c D or b = c where c $\in$ T 'terminal symbol' and D$\in$N 'non-terminal symbol'

B. Context-Free Grammar

In 1950, Noam Chomsky attempted to provide a specific definition of natural language structure [8]. His research has lead to the development of number of production rule restrictions and a promising class of grammar called context-free grammar. A grammar is considered context-free when the production rules R are as follows $A \overset{*}{\rightarrow} x$ where A is a single non-terminal symbol (A $\in$ N) and x is either a terminal or a non-terminal symbol (x $\in$ N $\cup$ T). Finite-state grammar is a type of a context-free grammar.

C. Probabilistic Context-Free Grammar

A probabilistic context-free grammar (PCFG) is a five tuple G= <T, N, R, P, S>where P is the probability associated with each productive rule such that the sum of rules for with the same left-side is equal to 1. To calculate the probability of a string x that uses the rule r where the probability of r if P_r and is called m_r times is as follows [1]: p(x)= $\prod_r (P_r)^{m_r}$

$0 \leq pr \leq 1$

Describing a data source with probabilistic grammar is called grammatical inference [2]. Grammatical inference provides a grammar that is identical to the data source with probability measures. Probabilistic Grammar has five tuples [7]. G= <T, N, R, P, S>, where:

T	finite set of terminal symbols
N	finite set of non-terminal symbols
R	finite set of rewrite rules
P	finite set of probabilities that are assigned by 1-1 mapping to R
S	start symbol

To infer a grammar, a number of sub-problems need to be solved [1]. First, collect and analysis a sample of the data. Second, choose a suitable number of grammars for the data. Third, outline how the suggested grammar is produced for the collected data set. Finally, verify if suggested grammar is a suitable representation for data (Figure 1). Each grammatical inference step is further described in later section detail.

IV. Data Mining

In this research, DNA sequences will be used as a case study data. However, the approach is not limited to DNAs. Finding an efficient approach to data mine that balances between time and accuracy is vital in order to utilize and benefit from stored data. Most data mining research topics in the field focused on developing a learning system based on a recognizer to mine data through neural network, clustering, and statistics. This paper conducts data mining using a different approach by using formal grammars.

Data mining is done on four main stages:

- Data exploration: In this step, a sample data is collected that can sufficiently represent the data source. This step is essential in analyzing and understanding the nature and structure of the data to generate an efficient grammar.
- Building model: The main step where all the analysis and knowledge is gathered. In this paper, this step will involve choosing a grammar, or number of grammars, that will work on the data based on the collected sample. Then choosing the most suitable one that can represent the data. Formal linguistics approaches has been used to describe DNA sequences biology since the structure of DNA has been solved in 1953[12]. Due to the nature of DNA sequences and complexity, most DNA structures are described and parsed using approaches more powerful than Context-Free Grammars such as DCG [12]. In this paper, we investigate data mining large amounts of DNA promoter sequences using PCFG and an algorithm introduced by Rajasekaran and Nicolae in 2014 as an error correcting cover grammar algorithm[15]. Probabilistic grammar has unique features that made it an interesting research choice. Probabilistic grammar can be used represent both statistical properties and structural relations of data [1].

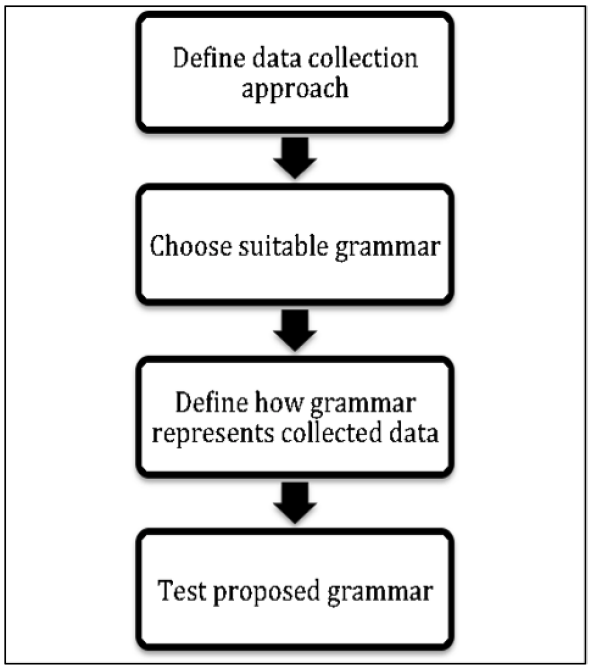

Fig. 1. Grammatical Inference Process

- Validation: Testing the build model on the actual data then testing it on another data set with similar structures and features. Which, in this paper, will involve applying the grammar then measuring results precision. Various methods of measurement are used with a preset acceptance value. One popular method is the Chi-square test.
- Deployment: When the grammar is accepted it can be deployed and used to mine data. Adjustments on the approach can be done after deployment to improve results can be done as deemed necessary.

Massive data are rarely saved in a single data source. Data is often distributed on multi-sources. There are several structures that can be used. Here, two structures are going to be studied; independent data sources and dependent data sources.

V. Grammatical Inference

Grammatical Inference is done in several steps [1]. A sample is first collected that represents the source is collected. Then, the grammatical inference process is done with assistance of the investigator. A basic grammatical inference model can be seen in figure 2.Then the goodness of the inferred grammar is test to be accepted or repeat the process.

A. Analyze Data

The nature of the data is studied to generate the grammar using a sample of that dataset. Terminal symbols are extracted. They can be found in data sentences collected from samples and as outputs of production rules. Non-terminal

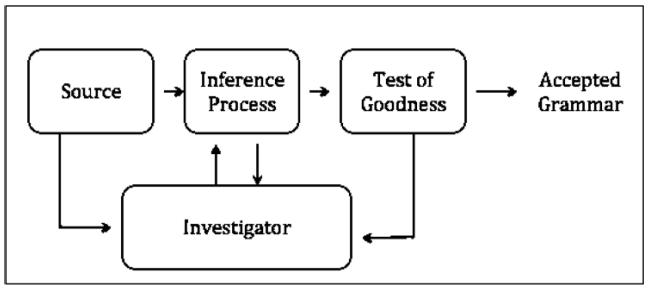

Fig. 2. Basic Model for Interactive Grammatical Inference[1]

symbols are also generated, which are the symbols that can be replaced and found as input or output of a production rule.Non-terminals can be seen as syntactic variables of the sentences. Grammatical rules production will depend on the type of grammar selected.

B. Selecting Suitable Grammar

The choice of grammar is based on the data that it is going to be applied on. Looking at the sample data, choose a suitable grammar that can reproduce all the data set. The grammar can be one of many types that differ in power based on the complexity of the data structure. Some of the most common types are Finite State Automa, Context-Free grammars, Probabilistic Context-Free grammars, Tree-Adjoining Grammar, and Context-Sensitive grammars.

C. Test and Validation

The last step is applying the test of goodness measure on the constructed grammar. If accepted, the results will be validated on the larger data set. A Chi-Square test will be used to confirm the fitness of the chosen grammar. Chi-square test is designed to confirm if the results probabilistic process is consistent or significantly different from what is assumed.

$$X^2(C,E) = \frac{(C_i - E_i)^2}{E_i}$$

where C is the number of observations found and E is the number of observations expected.

VI. Using Grammars for Data Mining

The grammar will be tested to data mine two different structures of data; independent and dependent.

A. Independent

Independent data sources are the sources were a single data entry that contains all input attributes in each entry. Data is searched and extracted orthogonally in independent data sources. However, this approach is more suitable for data with a limited number of attributes (Figure 3). Using PCFG to data mine independent data sources can help:

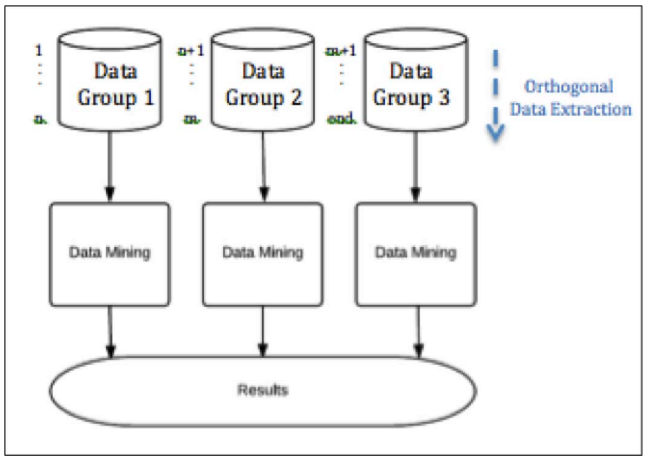

Fig. 3. Independent Data Distribution

- Identify and extract entries that belong to certain grammar in heterogeneous data sets. In other words, given several grammars, categorizing a large data set to various sub-sets based on sentries belonging to which grammar.
- PCFG can be used to identify the data that has the most similarity to the grammar structure. For instance, we were given a grammar for a human DNA sequence and horses DNA sequences data set, rats DNA sequences data set, and fish DNA sequences data set. These data sets can be data mined to determine the data set that has the highest probability based on the human DNA sequence grammar. This means this species has the most DNA similarity to human DNA.

For instance, given a large data set of heterogeneous DNA sequences of different types, the algorithm can scan through each data set and run the algorithm and extract sequences that belong to the grammar.

B. Dependent

Dependent distributed data is when data is gathered based on the attributes. For instance, when looking for a patients data in a hospital, there can be a data source dedicated to medical history, another for billing, and one for personal information. These data sources are linked to connect patients information. This approach is more complicated than the independent, since it requires searching each data source. Data extraction in this case is horizontal. (Figure 4). First a grammar is inferred to each data set separately. Then, the resulting grammars are concatenated to generate a single grammar. This resulting grammar will represent the overall structure of the distributed data sets.

VII. Case Study

As a case study for the framework, E. Coli DNA promoter sequences were chosen as data set. Promoter sequences are

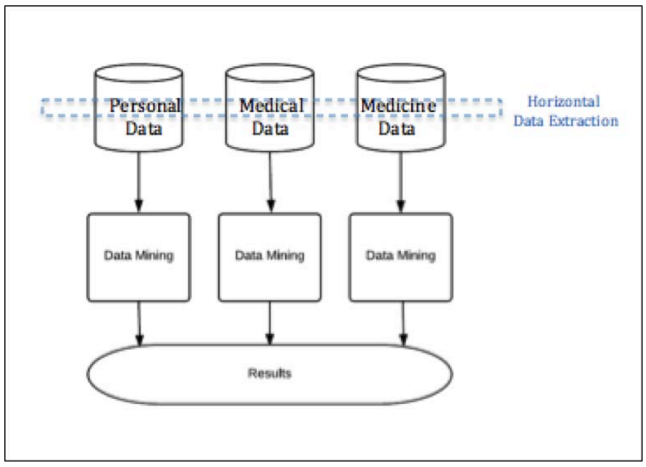

Fig. 4. Dependent Data Distribution

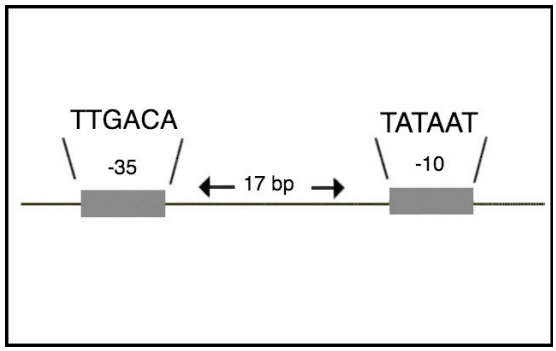

Fig. 5. A typical E. Coli Promoter Sequence

the part of DNA that effects the frequency and location if transcription initiation through interaction with RNA polymerase [14]. They are one the most researched and studied regions in DNA in bioinformatics. A sample of 106 E Coli DNA sequences is collected from the Machine Learning Repository UCI [12]. The data set contains 53 positive and 53 negative sets. DNAs Nucleic acid primary structure consists of four nitrogenous bases (Adenine, Guanine, Cytosine, and Thymine). The DNA sequence comprises of a large sequence of the four nitrogenous. Therefore, the grammars alphabet (T) will consist of four symbols {a, g, c, t} representing each nitrogens first letter respectively. Each DNA is a string consisting a sequence of alphabets.There are certain features that promoter DNA have that make them distinct from other DNA regions[12]. Certain repetitive pattern elements are identified that are commonly found in the DNA promoter region, which are the 'ttgaca' and 'tata' regions. These elements are going to be part of non-terminal symbols for our grammar. DNA E. Coli promoter sequences are identified by the -35 and -15 base pairs (bp) regions from the transcript site [14]. The -35 is called the TTGACA region and its standard form is ttgaca. The -15 bp region is called the TATA region and its standard form is tataat. These two regions are separated with a 15-19 long base pair sequence (Figure 5). However, these two region are frequently found as mutations of the standard sequence. For instance, taaaat and gacaat are both part of the TATA region [14]. The sequences preceding the TTGACA region and following the TATA region are called gap. A gap is a sequence that is not important or its significance is not known yet [19]. The sequence between the TTGACA and TATA regions is typically 17 bp space. However, it can range between 15-19 bp [16]. CFG is written in Chomsky Normal Form (CNF). A grammar is said to be in CNF when the grammar production rules are in only two forms; A $\rightarrow$ B C, or A $\rightarrow$ a.

Algorithm 1: Context-Free Grammar Error Correcting Algorithm[15]

Data: G = (N, T, P, S), a grammar;
I= $a_1 a_2 ... a_n$, input string;
Result: minimum distance ℓ and I and any string in L(G);
begin
  $GenerateG'$;
  for $A \in N'$ **and** $i \leftarrow 1$ **to** n **do**
    $X_A[i] := \{\}$;
  end
  for $i \leftarrow 1$ **to** n **and** $j \leftarrow (i+1)$ **to** *(n+1)* **do**
    $M_{i,j} := \{\}$;
  end
  for $i \leftarrow 1$ **to** n **do**
    for *(A* $\xrightarrow{\ell}$ a_i*)* $\in$ *P'* **do**
      insert(A, ℓ) into $M_{i,i+1}$;
      insert(i, i +1, ℓ) into $X_A[i]$;
    end
  end
  for $s \leftarrow 2$ **to** n **do**
    for $A \xrightarrow{\ell_2} BC \in P'$ **do**
      for $(i, k, \ell_1) \in$ *LIST(B)* **do**
        for $(C, \ell_2) \in M_{k,i+s}$ **do**
          $\ell := \ell_1 + \ell_2 + \ell_3$;
          insert (A, ℓ) into $M_{i,i+s}$;
          insert $(i, i+s, \ell)$ into X_A[i];
        end
      end
    end
  end
  find ℓ which (1, n + 1, ℓ) $\in X_S[1]$;
  return ℓ;
end

These regions that identify E. Coli promoter sequence, have multiple variations. This mismatch will require a beyond a simple CFG. Promoter sequences are rarely found in the

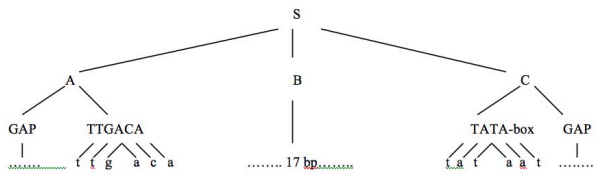

Fig. 6. Parsing for a Standard Promoter Structure

typical form (Figure 6). Searls used a Prolog-based definite clause grammar as an attempt to overcome CFG shortcomings to represent DNAs. This paper aims to utilize the CFG grammar by adding a error correcting cover grammar proposed by Rajasekaran and Nicolae that can parse in a cubic time (2014) [15]. The cover grammar counts the distance between a typical form of a sequence and the sequence given by counting differences (algorithm 1). The approach is extended and modified to sum probabilities instead of counting differences[20]. Probabilities on grammar rules are based on Lisser and Marglit research on 300 E. Coli mRNA promoter sequences [16][20].

After applying the grammar inferred to identify and extract DNA promoter sequences on the 106 sequences, the technique was able to correctly identify the positive sequences with 4 false positives out of the 53 negative and 53 positive data sequences given.

VIII. Conclusion

Data mining research are mostly focused on neural and statistical approaches. This paper presents a framework that can be used to data mine distributed data using formal grammar. Using formal grammar to data mine provides simplicity and description of targeted data. A sample of the data is gathered to be analyzed. Based on the nature of the data grammar is inferred. The grammar will be the base of the model for data mining. Two types of distributed data sources are studied; independent and dependent. This approach can be used for various applications;

- Separate large data into smaller sub-sets by inferring a grammar for each sub-category.
- In heterogeneous data set, it can identify data entries that belong to the grammar.
- Identify the degree of similarity between two data sets.

A case study for the technique was conducted on 106 E. Coli DNA promoter sequences. It was able to count and identify the positive sequences with 4 false positives.

References

[1] R. Ammar, *Interactive Grammatical Inference*, M.S. Thesis, University of Connecticut, 1981.

[2] K. Fu and T. Booth, *Grammatical Inference: Introduction and Survey - Part I*, IEEE Trans. Syst., Man, Cybern., vol. -5, no. 1, pp. 95-111, 1975.

[3] J. Horning, *A Study of Grammatical Inference*, Ph.D., Stanford: Stanford University Computer Science Department, 1969.

[4] W. Daelemans, *Colin de la Higuera: Grammatical inference: learning automata and grammars*, Machine Translation, vol. 24, no. 3-4, pp. 291-293, 2010.

[5] A. DUlizia, F. Ferri and P. Grifoni, *A survey of grammatical inference methods for natural language learning*, Artificial Intelligence Review, vol. 36, no. 1, pp. 1-27, 2011.

[6] Y. Sakakibara, *Recent advances of grammatical inference*, Theoretical Computer Science, vol. 185, no. 1, pp. 15-45, 1997.

[7] J. Borges and M. Levene, *Data Mining of User Navigation Patterns*, Web Usage Analysis and User Profiling, pp. 92-112, 2000.

[8] N. Chomsky, *Syntactic structures*. The Hague: Mouton, 1957.

[9] D. Searls, *The computational linguistics of biological sequences*, Artificial intelligence and molecular biology, vol. 2, pp. 47-120, 1993.

[10] S. Weiss and N. Indurkhya, *Predictive Data Mining: A Practical Guide*, 1998.

[11] Archive.ics.uci.edu, *UCI Machine Learning Repository: Molecular Biology (Promoter Gene Sequences) Data Set*, 1990. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Molecular+Biology+

[12] S. Datta and S. Mukhopadhyay, *A Composite Method Based on Formal Grammar and DNA Structural Features in Detecting Human Polymerase II Promoter Region*, PLoS ONE, vol. 8, no. 2, p. e54843, 2013.

[13] W. Raible, 'The Grammar of Genes: How the Genetic Code Resembles the Linguistic Code by ngel Lpez-Garca', Language, vol. 84, no. 4, pp. 885-888, 2008.

[14] C. Harley and R. Reynolds, *Analysis of E. coli promoter sequences*, Nucleic Acids Research, vol. 15, no. 5, pp. 2343-2361, 1987.

[15] S. Rajasekaran and M. Nicolae, *An error correcting parser for context free grammars that takes less than cubic time*, CoRR, vol. 14063405, 2014.

[16] S. Lisser and H. Margalit, *Compilation of E.coli mRNA promoter sequences*, Nucl Acids Res, vol. 21, no. 7, pp. 1507-1516, 1993.

[17] J. Mitchell, *Identification and analysis of 'extended -10' promoters in Escherichia coli*, Nucleic Acids Research, vol. 31, no. 16, pp. 4689-4695, 2003.

[18] S. Datta and S. Mukhopadhyay, *A Composite Method Based on Formal Grammar and DNA Structural Features in Detecting Human Polymerase II Promoter Region*, PLoS ONE, vol. 8, no. 2, p. e54843, 2013.

[19] S. Leung, C. Mellish and D. Robertson, *Basic Gene Grammars and DNA-ChartParser for language processing of Escherichia coli promoter DNA sequences*, Bioinformatics, vol. 17, no. 3, pp. 226-236, 2001.

[20] A. Algwaiz, S. Rajasekaran, A. Ammar, *Predicting E. Coli Promoters Using Formal Grammars*, submitted to The 15th IEEE International Symp. on Signal Processing and Information Technology, 2015.