Towards a Process Mining Approach to Grammar Induction for Digital Libraries
Syntax Checking and Style Analysis

Stefano Ferilli(✉) and Sergio Angelastro

Dipartimento di Informatica, Università di Bari, Bari, Italy
{stefano.ferilli,sergio.angelastro}@uniba.it

Abstract. Since most content in Digital Libraries and Archives is text, there is an interest in the application of Natural Language Processing (NLP) to extract valuable information from it in order to support various kinds of user activities. Most NLP techniques exploit linguistic resources that are language-specific, costly and error prone to produce manually, which motivates research for automatic ways to build them.

This paper extends the BLA-BLA tool for learning linguistic resources, adding a Grammar Induction feature based on the advanced process mining and management system WoMan. Experimental results are encouraging, envisaging interesting applications to Digital Libraries and motivating further research aimed at extracting an explicit grammar from the learned models.

Keywords: Natural Language Processing · Grammar Induction · Process Mining and Management

1 Introduction

One of the most relevant peculiarities and opportunities provided by Digital Libraries (DLs for short) and Archives, with respect to their physical counterparts, is the possibility of automatically accessing and processing their content by computers for several purposes. Some examples are: indexing aimed at faster and better information retrieval; topic extraction aimed at document organization or content understanding and summarization; content analysis aimed at supporting scholars in their research; etc.

Since most of DL content is in the form of text, Natural Language Processing (NLP) techniques play an important role. Such techniques may tackle problems at several levels of complexity. From the lowest to the highest we have:

Language Identification identifies the language in which a text is written;
Stopword Removal removes uninformative terms from a text;
Normalization reduces to standardized form inflected forms of words;
Part-of-Speech Tagging associates terms to their grammatical function;
Parsing returns the syntactic structure of sentences;

Understanding captures some kind of semantic information from the text.

While for some tasks (e.g., indexing and retrieval) the lexical level is sufficient, more advanced processing requires higher-level tasks to be carried out.

In turn, NLP techniques are often based on the use of linguistic resources: e.g., Language Identification often exploits n-gram distribution, Stopword Removal exploits lists of frequent terms, Normalization exploits lists of suffixes, Part-of-Speech (PoS for short) Tagging exploits suffixes and/or grammatical rules, Parsing uses grammars, Word Sense Disambiguation uses conceptual taxonomies or ontologies. The quality of such resources may dramatically affect the quality, or even determine the feasibility, of the NLP steps. However, developing these resources is a critical, costly, time-consuming and error prone task, because it is typically carried out manually by linguistic experts. To make things worse, each language requires its own set of resources. Most works in the literature are concerned with English [1,6,7,17,22], probably due to its having a structure which is easier than other languages and to its importance as the standard information interchange language worldwide. Little exists for a few other important languages [20], and almost nothing for the vast majority of minor languages. As a result, automatic processing techniques cannot be applied to documents in these languages, leading to the risk that entire cultures might be lost.

This situation motivated the development of (semi-)automatic techniques to learn the resources and other useful linguistic information from a (representative) set of texts in a given language. Our effort in this direction resulted in *BLA-BLA* (Broad-spectrum Language Analysis-Based Learning Application), a tool aimed at covering a wide spectrum of NLP tasks, including several techniques that allow to learn in a fully automatic way linguistic resources for language identification [12], stopword removal and term normalization [11,13] and concept extraction [18,21]. The learned resources may be used by NLP systems, and/or be taken as a basis for linguistic studies and/or further manual refinements. Most of the techniques in BLA-BLA are incremental, meaning that whenever more texts become available for the language, it is easy to run again the technique and obtain updated resources. This is a very important feature that is generally unavailable in other approaches in the literature.

This paper investigates the possibility of extending BLA-BLA with Grammar Induction and Checking features. In particular, we propose the use of an advanced Process Mining approach [16] called WoMan. As the first step in such an investigation, here we aim at assessing whether grammar models learned by WoMan are effective in recognizing the syntactic correctness of sentences in a given language. Experiments show that they are. Possible applications to document collections (libraries or archives) include an assessment of their overall linguistic quality, or an analysis of linguistic style variability therein.

This paper is organized as follows. After discussing some background and related work in the next section, Sect. 3 introduces the WoMan framework for process mining and management and Sect. 4 casts the linguistic problem into a process mining task. Then, Sect. 5 evaluates the proposed approach before concluding the paper.

2 Background and Related Work

Grammar Induction is a language acquisition problem. According to Gold's formalization [15], given a target language L from a set $\mathbf{L}$ of possible languages, a learner $\mathbf{C}$ is shown a sequence $[s_i]$ of positive examples ($\forall i : s_i \in L$) and after each example s_n it must maintain a hypothesis $L(\mathbf{C},[s_0, \ldots, s_n]) \in \mathbf{L}$ for L. Any $s \in L$ will be sooner or later present in the sequence (no guarantees on the order or frequency of examples). The hypothesis is *eventually correct* if $\exists k$ s.t. $\forall j > k : L(\mathbf{C}, [s_0, \ldots, s_k]) = L$. So, positive-only and incremental learning approaches are inherent this formalization. In general, a Grammar Induction algorithm should be able to discover an underlying grammar from examples, to be used to parse new sentences and to assess their grammaticality.

Approaches proposed in the literature can be divided into *supervised* and *unsupervised*. The former process sentences annotated with their constituent tree structure (e.g., the treebanks corpus [19]), which requires significant expert effort. In the latter, only words are annotated (manually or automatically) with their PoS tag. Unsupervised approaches typically exploit *phrase structure* or *dependency grammar* representations, and are further classified in *structural search*, aimed at discovering a suitable grammar structure that compactly describes the data, and *parameter search*, aimed at finding a set of optimal parameters for a fixed-structure grammar, such that the result best explains the language.

As regards Structural search approaches, [23] proposed a Bayesian model merging framework to find the structure of a probabilistic grammar. Possible uses include the discovery of an Hidden Markov Model (HMM) topology or the set of context-free production for a stochastic context-free grammar. The approach performs incremental merging operations on model substructures attempting to maximize the Bayesian posterior probability of the overall model. An objective evaluation of the model is missing. [7] presents a Context Distribution Clustering (CDC) algorithm that induces clusters of tag sequences based on the context in which they appear. A criterion based on *Mutual Information* between the left and right context of a sequence, is exploited to filter out non-constituent clusters. The algorithm is incorporated in a Minimum Description Length framework, whereby it chooses the clusters so that the resulting constituents have the shortest description length. However it is computationally expensive since it requires large amounts of memory.

As regards parameter search approaches, [1] proposed an inside-outside algorithm to induce Probabilistic Context-Free Grammars (PCFGs)[1], that generalizes the forward-backward algorithm for regular grammars with HMMs. The fixed model consists of productions in Chomsky Normal Form (CNF) (fully binary branching derivations). Given a sequence of words $W = w_p \ldots w_q$ in an example, the aim is to re-estimate production probabilities computing the inside probability $\beta(p, q)$ of generating W from a non-terminal X, and the outside probability $\alpha(p, q)$ of generating X and the words outside W in the example

[1] A PCFG is a context-free grammar with probabilities attached to productions.

from the start symbol. [6] extends the approach in [1] by introducing grammar constraints, which guide the search process avoiding the grammatically incompatible generation of non-terminals (e.g., a determiner from an adjective or a verb from a pronoun), and presents a set of experiments in inducing probabilistic dependency grammars.

The evaluation of a learned grammar is based on the comparison of either grammars or trees. Given a gold standard of correct parses, performance can be evaluated as the percentage of correct parses that the algorithm produces.

3 Process Mining and the WoMan Framework

This work aims at checking whether Process Mining approaches may be effective for dealing with the syntactic level of natural language. So, let us quickly recall some basic concepts of Process Mining. A *process* consists of actions performed by agents (humans or artifacts). A *workflow* is a formal specification of how these actions can be composed (using sequential, parallel, conditional, or iterative schemes) to result in valid processes. A *case* is a particular execution of activities compliant to a given workflow. It can be described in terms of *events* associated to the performed activities. Case *traces* consist of lists of events associated to time points. A *task* is a generic piece of work, defined to be executed for many cases of the same type. An *activity* is the actual execution of a task. Process Mining tasks of interest to this work are *Process Discovery*, aimed at learning a process model from sample cases, and *Conformance Checking*, aimed at checking whether a (new) case is compliant to a given process model.

In particular, inspired by the successful application of approaches based on HMMs to Grammar Induction in the literature, and by previous indications, obtained in other domains, that the Process Mining system WoMan may outperform HMMs in some cases, we propose the adoption of WoMan for our purposes. Also, while Process Mining and Management techniques in the literature have been typically motivated by and exploited in business and industrial domains, Woman proved able to support a wide variety of application domains (including Ambient Intelligence, and even Chess) [14].

The WoMan framework [9] introduced some important novelties in the process mining and management landscape. Experiments proved that it is able to handle efficiently and effectively very complex processes, thanks to its powerful representation formalism and process handling operators. In the following, we briefly and intuitively recall its fundamental notions.

WoMan takes as input trace elements consisting of 6-tuples $\langle T, E, W, P, A, O \rangle$, where T is the event timestamp, E is the type of the event (one of 'begin_process', 'end_process', 'begin_activity', 'end_activity', or 'context_description'), W is the name of the reference workflow, P is the case identifier, A is the name of the activity (or a list of contextual information for $A = \mathit{context\_description}$), and O is the progressive number of occurrence of that activity in that case.

WoMan models are expressed using two elements:

tasks: the kinds of activities that are allowed in the process;
transitions: the allowed connections between activities.

plus pre-/post-conditions (that specify what must be true for executing a given task or transition) in the form of First-Order Logic rules based on contextual and control flow information, possibly involving several steps of execution.

The core of the model, carrying the information about the flow of activities during process execution, is the set of transitions. A transition $t : I \Rightarrow O$, where I and O are multisets of tasks, is enabled if all input tasks in I are active; it occurs when, after stopping (in any order) the concurrent execution of all tasks in I, the concurrent execution of all output tasks in O is started (again, in any order). Any task or transition t is associated to the multiset C_t of training cases in which it occurred (indeed, a task or transition may occur several times in the same case, if loops or duplicate tasks are present in the model). It allows us to compute the probability of occurrence of t in a model learned from n training cases as the relative frequency $|C_t|/n$. As shown in [9,10], this representation formalism is more powerful than Petri or Workflow Nets [24], that are the current standard in Process Mining. It can smoothly express complex models involving invisible or duplicate tasks, which are problematic for those formalisms.

WoMan's supervision module, **WEST** (Workflow Enactment Supervisor and Trainer), takes the case events as long as they are available, and returns information about their compliance with the currently available model for the process they refer to. The output for each event can be 'ok', 'error' (e.g., when closing activities that had never begun, or terminating the process while activities are still running), or a set of warnings denoting different kinds of deviations from the model (e.g., unexpected task or transition, preconditions not fulfilled, unexpected resource running a given activity, etc.).

The learning module, **WIND** (Workflow INDucer), allows one to learn or refine a process model according to a case. The refinement may affect the structure and/or the probabilities. Differently from all previous approaches in the literature, it is *fully incremental*: not only can it refine an existing model according to new cases whenever they become available, it can even start learning from an empty model and a single case, while others need a (large) number of cases to draw significant statistics before learning starts. To learn conditions in form of logic theories, WIND relies on the incremental learning system InTheLex [8]. Indeed, InTheLEx is endowed with a positive only-learning feature [2], which allows it to deal with the positive-only learning approach typical of both Process Mining and Grammar Induction.

4 Grammar Induction as a Process Discovery Task

Following mainstream literature, we adopt the unsupervised setting for Grammar Induction. So, sentences in the training corpus are annotated with the sequence of PoS tags associated to their constituent tokens (words, values, punctuation). In our approach, a grammar corresponds to a process model; a sentence in natural language (actually, the sequence of PoS tags associated to the words in the sentence) is a case; a task is a PoS tag, and an activity is an occurrence of the tag in a sentence. Under this perspective, process discovery corresponds to grammar induction, and syntactic checking to conformance checking. Just as in Grammar Induction, process discovery typically adopts a positive-only learning approach (i.e., only correct sentences/process execution are included in the training corpus).

As regards the set of PoS tags to be used, several options are available in the literature. In the following, we will consider CoNLL-U, a revised version of the CoNLL-X format [5]. It represents a text in natural language as a plain text file, where three types of lines are available: comment lines (starting with #), blank lines (to separate sentences), and word lines (containing token annotations). Each word line reports 10 fields: word index *ID*; the word form or symbol *FORM* with its lemma or stem *LEMMA*; both universal (*UPOS*) and language specific (*XPOS*) PoS tag; the list of morphological features in *FEATS*; head *HEAD* of the current word (a value of *ID*) with its type of universal dependency relation *DEPREL*, and the list *DEPS* of head-deprel pairs forming the dependency graph; last, any other annotation in *MISC*. For instance, Table 1 shows the lines for sentence "Evacuata la Tate Gallery.", having ID *isst_tanl-3*.

Table 1. Example of a sentence in CoNLL-U format

ID	FORM	LEMMA	UPOS	XPOS	FEATS	HEAD	DEPREL	DEPS	MISC
#sent_id = isst_tanl-3									
#text = "Evatuata la Tate Gallery."									
1	Evacuata	Evacuare	*VERB*	*V*	Gender=Fem| Number=Sing| Tense=Past| VerbForm=Part	3	acl	-	-
2	la	il	*DET*	*RD*	Definite=Def| Gender=Fem| Number=Sing| PronType=Art	3	det	-	-
3	Tate	Tate	*PROPN*	*SP*	-	0	root	-	
4	Gallery	Gallery	*PROPN*	*SP*	-	3	flat:name	-	-
5	.	.	*PUNCT*	*FS*	-	3	punct	-	-

So, the sequence of PoS tags that make up a sentence in the CONLL-U file is transformed into a case trace in WoMan, where:

- the process name expresses the language of the sentence;
- the sentence id (*#sent_id*) is the case identifier;
- each word line determines an activity with the *UPOS* PoS tag (or combination *UPOS-XPOS*, for a more detailed model) as the activity name, and generates a pair of *begin_of_activity* and *end_of_activity* events;
- the features (*FEATS*) can be used as the context of the activity, and reported as a list of FOL predicates in a *context_description* event;
- *begin_of_process* and *end_of_process* events enclose the case.

For instance, using *UPOS* only for the activities, the sentence in Table 1 would generate the following trace in WoMan format:

```
entry(1,begin_of_process,ita_grammar_single,'isst_tanl-3',start,0).
entry(2,context_description,ita_grammar_single,'isst_tanl-3',
   [gender_fem(timestamp),number_ sing(timestamp),
   tense_past(timestamp),verbform_part(timestamp)],1,none).
entry(2,begin_of_activity,ita_grammar_single,'isst_tanl-3',verb,1).
entry(3,end_of_activity,ita_grammar_single,'isst_tanl-3',verb,1).
entry(4,context_description,ita_grammar_single,'isst_tanl-3',
   [definite_ def(timestamp),gender_fem(timestamp),
   number_ sing(timestamp),prontype_ art(timestamp)],1,none).
entry(4,begin_of_activity,ita_grammar_single,'isst_tanl-3',det,1).
entry(5,end_of_activity,ita_grammar_single,'isst_tanl-3',det,1).
entry(6,begin_of_activity,ita_grammar_single,'isst_tanl-3',propn,1).
entry(7,end_of_activity,ita_grammar_single,'isst_tanl-3',propn,1).
entry(8,begin_of_activity,ita_grammar_single,'isst_tanl-3',propn,2).
entry(9,end_of_activity,ita_grammar_single,'isst_tanl-3',propn,2).
entry(10,begin_of_activity,ita_grammar_single,'isst_tanl-3',punct,1).
entry(11,end_of_activity,ita_grammar_single,'isst_tanl-3',punct,1).
entry(12,end_of_process,ita_grammar_single,'isst_tanl-3',stop,1).
```

As regards the learned process model, the set of tasks is the same as the set of activities encountered in the training sentences. Using the *UPOS* tagset, that accounts for most natural languages, the model will include at most 17 tasks, which is a fair number for state-of-the-art Process Mining systems [10]. Using the *UPOS-XPOS* option, even if only a portion of all possible combinations is actually encountered in practice, the number of tasks significantly increases, going beyond the capabilities of many Process Mining systems in the literature but still being within reach for WoMan. Transitions correspond to pairs of adjacent PoS tags allowed in a sentence (e.g., [*det*] ⇒ [*pnoun*] means that a definite article may be followed by a proper noun). Note that process models representing grammars will not include any concurrency (sentences are just plain sequences of words), which means that the full power of WoMan in handling concurrency is not used in this domain, but also makes the comparison to HMMs more fair. However, such models will involve loops (including nested and short ones), optional and duplicate tasks, which are among the main sources of complexity in process mining and some of the strengths of WoMan. In particular, duplicate tasks are relevant, because different occurrences of the same PoS tag in a sentence represent distinct components of the discourse and cannot be handled by the same element of the model.

5 Experiments

Based on the proposed mapping between grammars and processes, we ran experiments aimed at checking whether WoMan is able to learn grammar models from sample sentences and whether the learned models can be used effectively for assessing grammatical correctness of new sentences. All experiments were run on a laptop endowed with a 2.8 GHz Intel Core i7-7700HQ Quad-Core (6M Cache) processor and 16 GB RAM, running on Kubuntu Linux 17.10. Our experiments concerned the Italian language, both because less resources are available for it in the NLP literature (but its grammar is well-known and studied in the linguistic area of research), and because its syntax is quite complex compared to English, and thus it can stress more the proposed approach.

5.1 Datasets Description

In our experiments we used two standard, publicly available datasets used for two EvalITA shared tasks:

UD_Italian-ISDT from EvalITA-2014 [3], obtained by conversion from ISDT (Italian Stanford Dependency Treebank), includes Wikipedia, News and Newspapers articles, Legal texts and Various genres and sentences. Since these are more formal and controlled texts, we expect to learn more reliable grammars from them.

PoSTWITA-UD from EvalITA-2016 [4], an Italian tweets collection. Since tweets often use fancy or odd sentences, it is expected to be more tricky than the other one.

Both datasets are annotated in Universal Dependencies (that can be exploited for the training of NLP systems), and are provided in CoNLL-U format and randomly split in three subsets (training, development and test). Since our approach does not need to tune any parameter, we will ignore the development subset in our experiments.

Table 2. Datasets statistics

Dataset	Corpus + Event log							
	Training set				Test set			
	#sent	#token	#event	#act	#sent	#token	#event	#act
UD_Italian-ISDT	13121	257616	784078	294397	482	9680	29632	11153
PoSTWITA-UD	5638	99441	266854	103553	674	12668	31910	12109

Table 2 reports some statistics about the datasets, for each considered subset thereof. For the linguistic perspective, it reports the number of sentences *#sent* (i.e., cases in a process perspective) and the overall number of tokens[2] *#token*. For the process perspective, it reports the number of events *#event* and the number of activities *#act* (i.e., instances of PoS tags associated to tokens or other symbols) generated by the translation into WoMan traces.

Both have several thousand training sentences, but **PoSTWITA-UD** has less than half than **UD_Italian-ISDT**, which is relevant because the former is expected to use a more tricky grammar than the latter, and thus to be more

[2] A token is a string with an assigned and thus identified meaning. Punctuation symbols are not tokens.

complex to learn. However, the former has a larger test set than the latter. The number of tokens/activities/events (which are somehow interrelated) are in the order of hundred thousands, which means the process discovery problem is not trivial.

5.2 Model Training

As a first step, we ran WoMan's Process Discovery feature to build a model for the Italian grammar from the training set(s). The learned models also included logic theories for pre- and post-conditions of tasks, as learned by InTheLEx. Table 3 shows some statistics about the models. As regards WoMan, for each *type* of tasks adopted (UPOS or UPOS-XPOS), it reports the number of tasks (*#task*) and transitions (*#trans*) in the learned model, and the average time per sentence (*time*) needed to learn the model (in seconds). The number of tasks and transitions is consistent among the two datasets for the two task types. The Twitter dataset has slightly less tasks, denoting a simpler lexicon, but slightly more transitions, denoting a more complex grammar. Nevertheless, the average time to process each sentence is the same, and is really low, allowing the use of the system for real applications. As regards Inthelex, again for each *type* of tasks adopted (UPOS or UPOS-XPOS), Table 3 reports the number of rules (*#rules*), and the average time per example (*time*) needed to learn them, for pre- and post-conditions of tasks. It also reports, for reference, the number of examples, which is the same as the number of activities in Table 2 (because WoMan generates one example of pre-condition and one example of post-condition for each activity). Again, the complexity of the theories (number of rules) is consistent between the two datasets, and the time needed to learn them is very low.

Table 3. Model statistics

Dataset	WoMan				Inthelex				
	Type	#task	#trans	time	#ex	Pre-Conds		Post-Conds	
						#rules	time	#rules	time
UD_Italian-ISDT	UPOS	17	273	0,04	294397	24	0,02	19	0,02
	UPOS-XPOS	51	905	0,04		66	0,01	43	0,03
PoSTWITA-UD	UPOS	17	307	0,04	103553	21	0,01	17	0,02
	UPOS-XPOS	45	1082	0,04		56	0,01	49	0,03

5.3 Model Evaluation and Possible Uses in DLs

Model evaluation consisted in a grammatical checking of new sentences with respect to a learned grammar. In Process Mining terms, this corresponds to a *Conformance Checking* task of new event traces with respect to a process model.

The learned models were evaluated in two different ways. First, for each dataset and task setting (UPOS or UPOS-XPOS), we ran classical 10-fold cross validation on the training set. This allowed us to understand how good WoMan was to learn the grammar underlying a given collection. Then, we tested the grammar learned on the entire training set of **UD_Italian-ISDT** on the sentences in the test sets. Sentences in the test set of **PoSTWITA-UD** were tested on the grammar learned from **UD_Italian-ISDT**. We did so because the former is a set of tweets, that are expected to use odd sentence structures, while the latter is a set of more controlled sentences, which are expected to use a correct grammar. So, testing the former on the latter may provide an indication of how bad a grammar tweets use, and of how good the system is in rejecting sentences with wrong syntax.

Table 4 reports the experimental results, evaluated according to: *Accuracy*, computed as the average portion of sentences identified as correct (i.e., the conformance checking never raised 'unexpected task or transition' warnings); *Support*, defined as percentage of training cases having the same structure as the sentence being tested; and *Runtime* (in minutes) spent to run the test procedure.

10-fold cross-validation results show that WoMan is extremely effective in learning the grammar underlying a given collection. In a DL, this would allow the librarians to understand whether the grammar style adopted by new texts added to the collection are consistent with the previous content, or to check user comments before publishing them if a discussion section is provided for. These figures also suggest the possibility of using a further feature of WoMan, which is process prediction [14], to automatically classify the linguistic style of new incoming texts from a pre-defined set of syntactic models. Average support for **UD_Italian-ISDT** is 20%, i.e., each test sentence has the same syntactic structure as 1/5 of the training sentences. For **PoSTWITA-UD** this number is more than doubled, indicating much less variability in writing style, as expected. In a DL, this would allow the librarian to determine how rich is the grammatical structure of the collection.

In the evaluation based on test sets, figures for **UD_Italian-ISDT** basically confirm the results of the cross-validation, reaching an even better performance (100%) for the UPOS setting. As regards **PoSTWITA-UD**, accuracy drops significantly, as expected due to the very different and informal syntax used in tweets with respect to more formal texts. In particular, it is still high for the UPOS setting, albeit almost 10% less than for the 10-fold cross-validation, but it drops to less than a half the value for UPOS-XPOS, as expected due to the fact that the more tasks available, the more complex the model, the more cases are needed to fully learn it. Support is about 10%, indicating much more style variability in the test set. In a DL, this would allow the librarian to distinguish the literary level of a text, or to distinguish texts by their type of content (e.g., novels vs. articles).

Runtime is still low, considering that several hundred sentences are processed in each test phase, ensuring scalability of the framework. Interestingly, again, runtime is higher to process the separate test set than the test sets obtained in the cross-validation.

Table 4. Performance statistic

	Measures	ISDT training		PoSTWITA training	
		UPOS	UPOS+XPOS	UPOS	UPOS+XPOS
Train and test	Accuracy	100%	99%	91%	43%
	Support	19%	19%	13%	12%
	Test runtime (min)	2.57	2.24	3.5	4.79
10-fold cross validation	Accuracy	99%	99%	99%	97%
	Support	20%	20%	42%	43%
	Runtime per fold (min)	1.5	2.4	2.99	4.77

6 Conclusions and Future Work

Since most content in Digital Libraries is in the form of text, there is an interest towards the application of Natural Language Processing (NLP) techniques that can extract valuable information from it in order to support various kinds of user activities. Most NLP techniques exploit linguistic resources that are language-specific, costly and error prone to produce manually, which motivates research for automatic ways to build them. Carrying on previous work on the BLA-BLA tool for learning several kinds of linguistic resources, this paper focuses on Grammar Induction. In particular, it investigates the ability of advanced process mining and management techniques to automatically learn, from a set of texts in a given language, effective models to check grammatical correctness of sentences in that language. In particular, it works on WoMan, a declarative process mining system that proved able to learn effective models in several application domains.

Experimental results show that the approach is effective for grammar checking and allows interesting analysis of the collections in a DL. Other possible applications to DLs can be also immediately envisaged, such as classification of the linguistic style in the form of process prediction, and will be further investigated. Future work will aim at extracting an explicit grammar from the learned models. This should be feasible, since WoMan models already learn grammatical rules made up of just single PoS tags, and so more complex rules might be obtained by suitable combinations thereof.

References

1. Baker, J.K.: Trainable grammars for speech recognition. J. Acoust. Soc. Am. **65**(S1), S132–S132 (1979)
2. Bombini, G., Di Mauro, N., Esposito, F., Ferilli, S.: Incremental learning from positive examples. In: Atti del (2009)
3. Bosco, C., Dell'Orletta, F., Montemagni, S., Sanguinetti, M., Simi, M.: The Evalita 2014 dependency parsing task. In: EVALITA 2014 Evaluation of NLP and Speech Tools for Italian, pp. 1–8. Pisa University Press (2014)

4. Bosco, C., Fabio, T., Andrea, B., Mazzei, A.: Overview of the Evalita 2016 part of speech on Twitter for Italian task. In: CEUR Workshop Proceedings, vol. 1749, pp. 1–7 (2016)
5. Buchholz, S., Marsi, E.: CoNLL-X shared task on multilingual dependency parsing. In: Proceedings of the Tenth Conference on Computational Natural Language Learning, pp. 149–164. Association for Computational Linguistics (2006)
6. Carroll, G., Charniak, E.: Two experiments on learning probabilistic dependency grammars from corpora. Department of Computer Science, Univ. (1992)
7. Clark, A.: Unsupervised induction of stochastic context-free grammars using distributional clustering. In: Proceedings of the Workshop on Computational Natural Language Learning, vol. 7, p. 13. Association for Computational Linguistics (2001)
8. Esposito, F., Semeraro, G., Fanizzi, N., Ferilli, S.: Multistrategy theory revision: induction and abduction in INTHELEX. Mach. Learn. **38**(1–2), 133–156 (2000)
9. Ferilli, S.: WoMan: logic-based workflow learning and management. IEEE Trans. Syst. Man Cybern. Syst. **44**, 744–756 (2014)
10. Ferilli, S., Esposito, F.: A logic framework for incremental learning of process models. Fundam. Inform. **128**, 413–443 (2013)
11. Ferilli, S., Esposito, F., Grieco, D.: Automatic learning of linguistic resources for stopword removal and stemming from text. Proc. Comput. Sci. **38**, 116–123 (2014)
12. Ferilli, S., Esposito, F., Redavid, D.: Language identification as process prediction using WoMan. In: Proceedings of the 12th Italian Research Conference on Digital Library Management Systems (IRCDL-2016), p. 12 (2016)
13. Ferilli, S., Grieco, D., Esposito, F.: Automatic learning of linguistic resources for stopword removal and stemming from text. In: Agosti, M., Ferro, N. (eds.) Proceedings of the 10th Italian Research Conference on Digital Library Management Systems (IRCDL-2014), p. 12 (2014)
14. Ferilli, S., Esposito, F., Redavid, D., Angelastro, S.: Predicting process behavior in WoMan. In: Adorni, G., Cagnoni, S., Gori, M., Maratea, M. (eds.) AI*IA 2016. LNCS, vol. 10037, pp. 308–320. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49130-1_23
15. Gold, E.M., et al.: Language identification in the limit. Inf. Contr. **10**(5), 447–474 (1967)
16. IEEE Task Force on Process Mining: Process mining manifesto. In: BPM Workshops, LNBIP, vol. 99, pp. 169–194 (2012)
17. Lari, K., Young, S.J.: The estimation of stochastic context-free grammars using the inside-outside algorithm. Comput. Speech Lang. **4**(1), 35–56 (1990)
18. Leuzzi, F., Ferilli, S., Rotella, F.: ConNeKTion: a tool for handling conceptual graphs automatically extracted from text. In: Catarci, T., Ferro, N., Poggi, A. (eds.) IRCDL 2013. CCIS, vol. 385. Springer, Heidelberg (2013). Publications/ircdl2013.pdf
19. Marcus, M., et al.: The Penn Treebank: annotating predicate argument structure. In: Proceedings of the Workshop on Human Language Technology, HLT 1994, pp. 114–119. Association for Computational Linguistics, Stroudsburg (1994). https://doi.org/10.3115/1075812.1075835
20. Naseem, T., et al.: Using universal linguistic knowledge to guide grammar induction. In: Proceedings of the 2010 Conference on Empirical Methods in NLP, pp. 1234–1244. Association for Computational Linguistics (2010)
21. Rotella, F., Leuzzi, F., Ferilli, S.: Learning and exploiting concept networks with conNeKTion. Appl. Intell. **42**, 87–111 (2015)

22. Spitkovsky, V.I., Alshawi, H., Jurafsky, D.: Three dependency-and-boundary models for grammar induction. In: Proceedings of the 2012 Joint Conference on Empirical Methods in NLP and Computational Natural Language Learning, pp. 688–698. Association for Computational Linguistics (2012)
23. Stolcke, A., Omohundro, S.: Inducing probabilistic grammars by Bayesian model merging. In: Carrasco, R.C., Oncina, J. (eds.) ICGI 1994. LNCS, vol. 862, pp. 106–118. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58473-0_141
24. Weijters, A., van der Aalst, W.: Rediscovering workflow models from event-based data. In: Proceedings of the 11th Dutch-Belgian Conference of Machine Learning (Benelearn 2001), pp. 93–100 (2001)