Language Identification as Process Prediction Using WoMan

Stefano Ferilli[1,2(✉)], Floriana Esposito[1,2], Domenico Redavid[3], and Sergio Angelastro[1]

[1] Dipartimento di Informatica, Università di Bari, Bari, Italy
{stefano.ferilli,floriana.esposito,sergio.angelastro}@uniba.it
[2] Centro Interdipartimentale per la Logica e sue Applicazioni, Università di Bari, Bari, Italy
[3] Artificial Brain S.r.l., Bari, Italy
redavid@abrain.it

Abstract. Several high-level tasks in the management of Digital Libraries require the application of Natural Language Processing (NLP) techniques. In turn, most NLP solutions are based on linguistic resources that are costly to produce, and so motivate research for automated ways to build them. In particular, Language Identification is a crucial NLP task, that is preliminary to almost all the others, since different linguistic resources must be used for different languages. This paper investigates process mining and management approaches as a possible solution to the Language Identification problem. Specifically, it casts language identification as a process prediction task, and exploits the WoMan framework to carry it out. Experimental results are encouraging and suggest to further explore this approach.

Keywords: Natural Language Processing · Language identification · Process mining and management

1 Introduction

In order to perform several kinds of analysis, categorization and understanding of documents in a Digital Library, Natural Language Processing (NLP for short) techniques are needed. Research on NLP has developed a number of strategies, tools and resources that allow to perform many different tasks, from low-level ones, aimed at preprocessing the text, to high-level ones, aimed at extracting different kinds of information from it. These tasks may be connected in a sort of pipeline in order to extract many features and components from texts, through different levels of increasing complexity (morphological, lexical, and syntactic; more recently, also the semantic level, which is clearly the most challenging one, has been approached) [8]. Typical examples of steps in this pipeline are Language Identification (aimed at automatically discovering the language in which a document is written), Stopword Removal (that removes the terms that are widespread

C. Grana and L. Baraldi (Eds.): IRCDL 2017, CCIS 733, pp. 159–172, 2017.
DOI: 10.1007/978-3-319-68130-6_13

and frequent in any kind of text and hence are not informative about the specific text content), Normalization (that standardizes to a single form, stem or lemma, different inflected occurrences of the same term), PoS Tagging (that associates each term to its grammatical function), Parsing (that builds the syntactic structure of sentences), and Word Sense Disambiguation (that associates each term in the text to the underlying concept, to attack the synonymy and polysemy problems that affect natural language).

NLP tasks are typically based on linguistic resources, that must be available for them to work properly. Most such resources have been built for English, both because it has a simpler syntactical structure than other languages, and because it has established itself as the standard for scientific and administrative interaction all over the world. Much less has been done to deal with a few other important languages or with the vast majority of minor languages, dialects, jargons and slangs, that nevertheless represent a significant piece of our cultural heritage and a precious source of information. Unfortunately, experimental evidence shows us that approaches which perform well on a language are not ensured to behave in the same way on others. Building linguistic resources usually requires the intervention of human experts, and thus it is typically a time-consuming, costly and error-prone task. This motivates research to automatically learn such resources.

In this landscape, Language Identification plays a crucial role. Not only may it provide significant help to librarians, scholars or enthusiasts working with collections of multilingual documents. It is also fundamental for the whole NLP pipeline, because, depending on the language in which a text is written, different, language-specific, tools and resources must be exploited in the various steps of the pipeline. So, in order to answer Gordon's question (2005): "How well do existing language identification techniques support languages which form the bulk of the more than 7000 languages identified in the Ethnologue?", carrying on previous research in this direction [6,18], this work proposes the automatic learning of a tool for Language Identification. More specifically, the approach adopted for this purpose relies on Process Mining and Management techniques. Compared to existing approaches in the literature, it provides a more powerful representation of models, which might be leveraged in the future to improve the model's predictive performance, and allows the incremental learning and refinement of language models, which is important to improve the performance of existing language models and to add new languages at need, without having to learn from scratch using all at once huge datasets.

This paper is organized as follows. The next section recalls some background and related work on Language Identification and Process Mining. Section 3 describes the WoMan framework for process management. Then, Sect. 4 proposes the process-based approach to language identification and reports experimental results. Finally, Sect. 5 draws some conclusions and outlines future work issues.

2 Background and Related Work

As for many other NLP tasks, also Language Identification relies on the availability of suitable linguistic resources. A variety of features have been proposed

in literature as relevant to determine in which language a given text is written [19]. Sequences of letters, called n-grams, are a typical example [17], but also stopwords (as the words most frequently used in a language), or the suffixes used for inflection are significant [6].

Many works are based on the presence of particular character n-grams and often exploit n-gram probability distribution [1,3,11,15]. An n-gram is an n-character substring of a text, which can also refer to any co-occurring set of characters in a string. Typically, a string is sliced into a set of overlapping n-grams. The n-gram based approach introduced in [3] for text categorization, provides a popular, high-performance methodology for language identification. Attempts to build automatically resources for the task of language recognition are often based on statistics concerning n-grams occurrence. Some techniques are based on Recurrent Neural Networks (RNNs), as in [10], which have the drawback of requiring a large amount of data to be trained. Other solutions adopt Hidden Markov Models (HMMs), where visible states are considered to be a sequence of n-grams [2,12] and the probability of an observation is assumed to depend only on the previous $n-1$ observations [20].

Other works are based on words. E.g., [3] states that human languages invariably have some words which occur more frequently than others, which implies that there is always a set of words which dominates most of the other words of the language in terms of frequency of use. Many techniques, along the various steps of the NLP pipeline, consider only the list of terms appearing in the text (Bag-of-Words based processing), possibly associated to weights (often directly or indirectly related to their frequency). This approach has proven to be a good trade-off between efficiency and effectiveness in many applications.

As said, this work aims at checking whether a technique based on a Process Mining approach [9], where a text in natural language is seen as a process based on n-grams representation, may be effective for language identification. So, a quick recall of Process Mining basics may be helpful here. A *process* consists of actions performed by agents (humans or artifacts). A *workflow* is a formal specification of how these actions can be composed to result in valid processes. Allowed compositional schemes include sequential, parallel, conditional, or iterative execution. A process execution can be described in terms of *events* associated to the performed activities. A *case* is a particular execution of activities compliant to a given workflow. 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.

Interestingly, early research on Process Mining used HMMs as the underlying learning and representation approach, before being superseded by more specific solutions. Inspired by this connection with the Language Identification literature, here we would like to make the opposite journey, starting from Process Mining-specific approaches and checking whether they can be profitably exploited on a task where HMMs have proved effective.

While Process Mining and Management techniques have been typically motivated by and exploited in business and industrial domains, and their typical

tasks have been of mining and supervision of process enactment, more recently other application fields (such as Ambient Intelligence, and even Chess) have been approached with these techniques, and also the additional task of prediction has gained increasing attention [7]. In this perspective, given a formal model of the desired process behavior and an intermediate status of a process execution, the goal is predicting how the execution might proceed, or what kind of process is being enacted, among a set of candidates. Interesting solutions for the process prediction task adopt the WoMan framework for process management [5], that proved able to support the prediction task in other application domains [7]. Specifically, it is based on a representation formalism that can support the prediction task.

3 The WoMan Framework for Process-Related Prediction

The WoMan framework [4] lies at the intersection between *Declarative* Process Mining [16] and Inductive Logic Programming (ILP) [14]. It 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. The technical details of how WoMan works for process prediction are out of the scope of this paper. The interested reader is referred to [7] for a more technical description of this. 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'), W is the name of the reference workflow, P is the case identifier, A is the name of the activity, and O is the progressive number of occurrence of that activity in that case.

WoMan models describe the structure of workflows using two elements:

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

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 [4,5], this representation formalism is more powerful than Petri or Workflow Nets [21], 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.

While in supervision mode, WoMan can make several kinds of predictions. Specifically, **WoGue** (Workflow Guesser), given the events of an unknown workflow, returns a ranking (by confidence) of a set of candidate process models. Confidence here is not to be interpreted in the mathematical sense. It is determined based on a heuristic combination of several parameters associated with the possible alternate process statuses that are compliant with the current partial process execution.

4 Language Identification with WoMan

Using process prediction approaches for the Language Identification problem, predictions are more complex than in industrial processes, because there is a much larger set of possible task and much more variability and subjectivity in the users' behavior, and there is no 'correct' underlying model, just some kind of 'typicality' can be expected.

4.1 Language Identification as a Process

Following mainstream literature, we adopt the n-gram based approach to Language Identification. Compared to the stopword-based approach, it is more directly applicable, because there is no need to know the set of stopwords in advance (which would require a further linguistic resource to be available). Also, we consider only lowercase letters, shifting to lowercase all uppercase ones and ignoring spaces, punctuation and other symbols.

So, in our solution, each n-gram is a task in a process perspective. Using the basic latin alphabet, that underlies most natural languages (and can be used to transliterate most other languages using different alphabets), and ignoring punctuation and other symbols, this yields overall 26 unigrams, $26^2 = 676$ bigrams and $26^3 = 17576$ trigrams. In a process mining setting, 676 or 17576 tasks are a

really huge number, often outside the reach of state-of-the-art systems [5]. Transitions among tasks happen whenever a new character is read from the text, and consist in stepping from the current n-gram to the next one obtained by removing the first character of the previous n-gram and appending the new character to the result. So, for instance, the sentence '*Hello, world!*' would generate the following sequences of n-grams (where blanks, punctuation and other symbols have been replaced by a star *****):

1-grams: *, h, e, l, l, o, *, w, o, r, l, d, *
2-grams: **, *h, he, el, ll, lo, o*, *w, wo, or, rl, ld, d*, **
3-grams: ***, **h, *he, hel, ell, llo, lo*, o*w, *wo, wor, orl, rld, ld*, d**, ***

Clearly, our process models will not involve any concurrency, which simplifies the mining task and allows a fair comparison to HMMs. However, these models will involve loops (including nested and short ones), optional and duplicate tasks, which are among the main sources of complexity in process mining.

In our setting, a case is a sentence. Compared to using full texts as cases, this results in short cases, each of which involves just a very small fraction of all possible tasks (i.e., n-grams), making it more difficult for WoMan to find a training case that exactly matches a new case to be processed for predictions. However, this allows to get a larger set of training cases from a restricted set of long texts. This setting requires a way to chunk sentences in a text. Instead of relying on off-the-shelf chunkers, we adopt the baseline solution that splits sentences whenever a full stop is encountered. While possibly returning wrong sentences from time to time, this avoids the need for a further resource, which is one of the fundamental requirements for our work.

4.2 Datasets Description

We collected a dataset involving 7 languages: English, Italian, Spanish, French, Portuguese, German and Squinzanese. So, it includes both latin languages and german ones. Also, very close languages are included, to make the learning and prediction task more difficult. Squinzanese, in particular, is a dialect from Southern Italy, which mainly comes from Latin but was also affected by the Spanish and French dominations, and more recently by the spread of Italian as a national language. It also has some connections to German and Arabic in a few words.

11 texts per language were collected: 10 to be used for training, and 1 for testing. In order to have texts of medium size, and based on the available literature, tales were considered. We tried to have translations of the same texts in the different languages. This should ensure a more standardized base. Especially for closer languages, this made the recognition problem more difficult (because they use similar words and, thus, similar n-grams). For this purpose, we considered well-known tales taken from the Grimm brothers collection (www.grimmstories.com), whose translation was available for all languages except for Squinzanese. In fact, dialects have a mainly oral tradition and it is not easy to find written stuff for them. However, a book of tales was available also for Squinzanese [13], and

Table 1. Dataset statistics

Language	Training set		Learning runtime (sec)						Test set	
	#sent	#char (avg)	1-gram		2-gram		3-gram		#sent	#char (avg)
			Tot	Avg	Tot	Avg	Tot	Avg		
English	679	133.486	385.216	0.57	254.248	0.37	267.432	0.39	49	119.367
French	855	110.991	1996.724	2.34	2555.46	2.99	1874.72	2.19	61	104.049
German	665	141.883	372.992	0.56	373.6	0.56	996.272	1.5	45	130.022
Italian	801	146.571	344.708	0.43	403.46	0.5	457.064	0.57	100	157.91
Portuguese	823	113.205	346.648	0.42	321.972	0.39	334.732	0.41	56	98.911
Squinzanese	696	152.927	287.348	0.41	324.78	0.47	355.852	0.51	199	138.548
Spanish	742	119.923	186.48	0.25	213.244	0.29	263.948	0.36	46	124.5

we were lucky enough that it reported also the Italian translation of the tales. So, for Italian we used the translations of the tales selected for Squinzanese instead of those by the Grimm brothers, so that the former have representatives in at least two languages.

Table 1 reports some statistics on the experimental dataset. Specifically, for each language, it reports: number of sentences (i.e., cases in a process perspective) and average number of characters per sentence (each character determines a new task for all n-gram settings), both for the training set and for the test documents. As regards the training set, the number of sentences ranges from 679 for English to 855 for French, which is interesting because they were translations of the same texts, and the average number of characters per sentence is comparable. More variability is present in the test set, where Italian and Squinzanese include a almost twice as much sentences as the other languages, but still involve a comparable number of characters.

4.3 Model Training

Concerning the learning procedure, we fixed a random sequence S of the training texts, and then learned 30 models for each language, one for each setting (n, m), where $n \in \{1, 2, 3\}$ is the size of n-grams and $m \in \{1, \ldots, 10\}$ means that the first m texts in S were used for training. Table 2 shows some statistics for models (1, 4), (2, 9) and (3, 10), which provided the best average performance on the overall test set. Figures are comparable for all settings and languages, except Italian (somehow smaller) and Squinzanese (significantly smaller), possibly due to the different tales used for them. The number of tasks (i.e., n-grams) grows exponentially for increasing n, but its proportion compared to all possible n-grams is smaller and smaller. In any case, for $n = 2, 3$ this number is still huge, compared to the usual numbers handled by state-of-the-art process management systems in industrial environments. Nevertheless, WoMan was able to deal with this complex task. Tables 1 and 2 also report the runtime needed to learn the models (overall and/or average per sentence). Except for French and for German as regards 3-grams, average runtimes per sentence are comparable in all settings (n, m), regardless of the number of texts used, ensuring acceptable performance.

Table 2. Models statistic

Language	#cases			#tasks			#transitions			Avg runtime (sec)		
	(4, 1)	(9, 2)	(10, 3)	(4, 1)	(9, 2)	(10, 3)	(4, 1)	(9, 2)	(10, 3)	(4, 1)	(9, 2)	(10, 3)
English	370	639	679	29	404	2655	378	2619	8586	0.29	0.37	0.39
French	462	786	855	40	478	2935	436	2890	9421	0.24	0.34	2.19
German	350	623	665	34	518	3033	478	2968	9182	0.69	0.60	1.5
Italian	366	751	801	30	330	2339	319	2298	8602	0.36	0.49	0.57
Portuguese	399	757	696	43	552	2095	494	3007	8586	0.22	0.27	0.41
Squinzanese	208	589	823	29	345	3084	283	1995	9993	0.45	0.45	0.51
Spanish	372	694	742	38	505	3156	464	3095	10516	0.22	0.35	0.36

Figure 1 shows the plots of the learning behavior for the different n-gram settings, where the i-th point represents the number of changes applied to the model after processing the i-th sentence. Plots associated to the various language models are superimposed, in order to give the reader an overall idea of the learning trend. The fact that peaks become lower and sparser as long as the plot proceeds to the right confirms that the learned models actually converge. Of course, as long as n is increased, convergence comes later, as expected due to the increasing number of features to be handled.

4.4 Language Identification Performance

Concerning the testing procedure, for each set of 7 models/languages associated to the same (n, m) pair, we tested performances on the test set made up by all test texts (one for each language) as follows. On each event in the test set, WEST was called on each model to check compliance with the various languages and suitably update the process statuses, then WoGue was called using as candidate models the 7 models for the various languages. WoGue always returns a prediction, expressed as a ranking of the models/languages by decreasing confidence.

Each setting was evaluated according to four measures (see Tables 3 and 4). One is accuracy (Acc), computed as the average position of the correct prediction in the ranking, normalized to $[0, 1]$, where 1 represents the top of the ranking, and 0 its bottom. The other measures assess, on average, for what percentage of the case duration the prediction was:

correct (C): i.e., the correct process was alone at the top of the ranking;
uncertain (U): i.e., the correct process was at the top of the ranking together with others; or
wrong (W): i.e., the correct process was not at the top of the ranking.

Figure 2 shows the trend in accuracy and correctness (averaged on all languages) of the models learned for 1-, 2- and 3-grams as long as more texts are processed. While one might expect that the plots would always be monotonically increasing, we note that this somehow holds for 3-grams only, while for both 1-grams and 2-grams the plot reaches some kind of plateau after processing 4 texts. It is also very interesting to note that, after processing 2 texts or

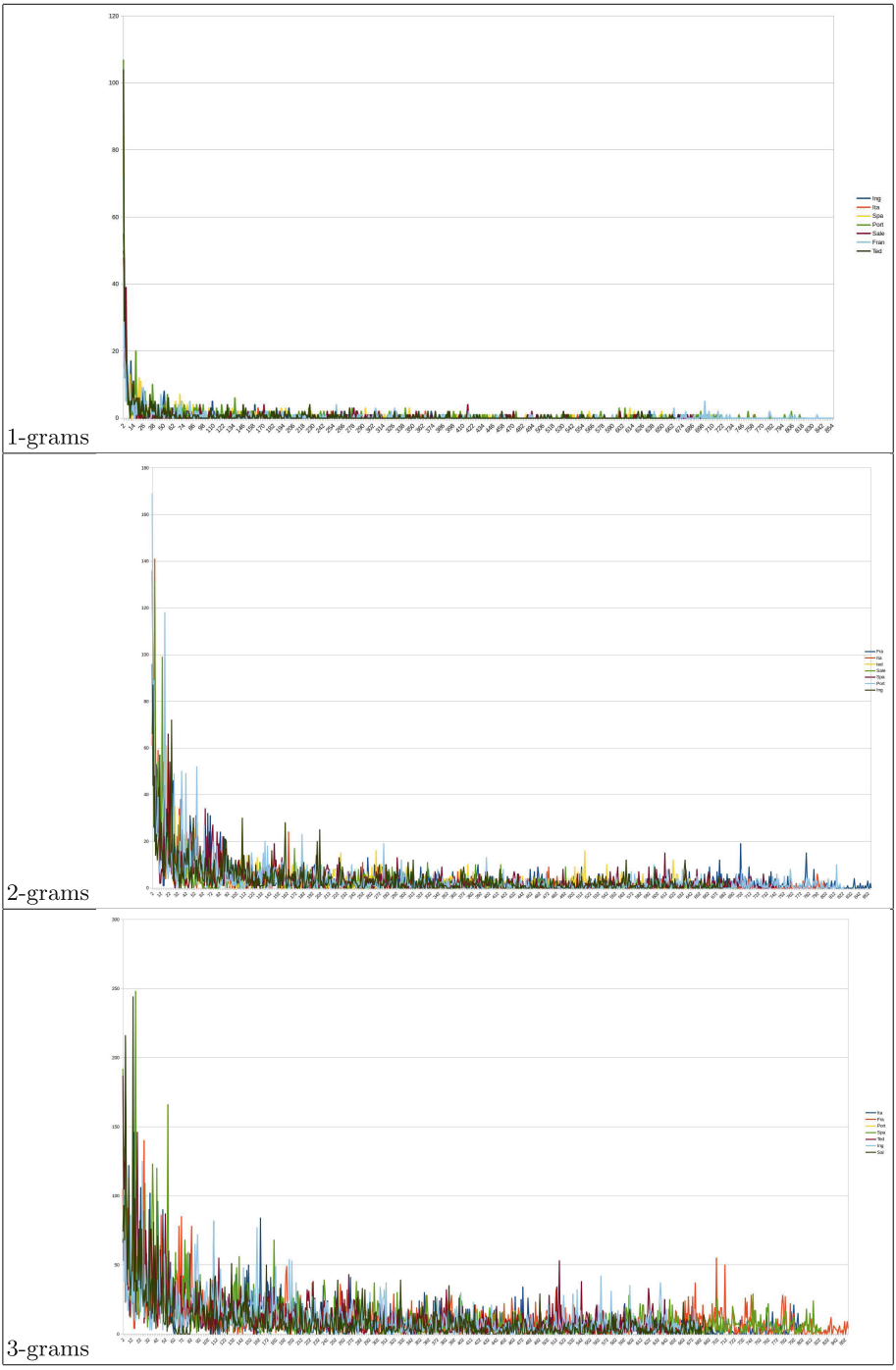

Fig. 1. Learning trend for different n-gram settings

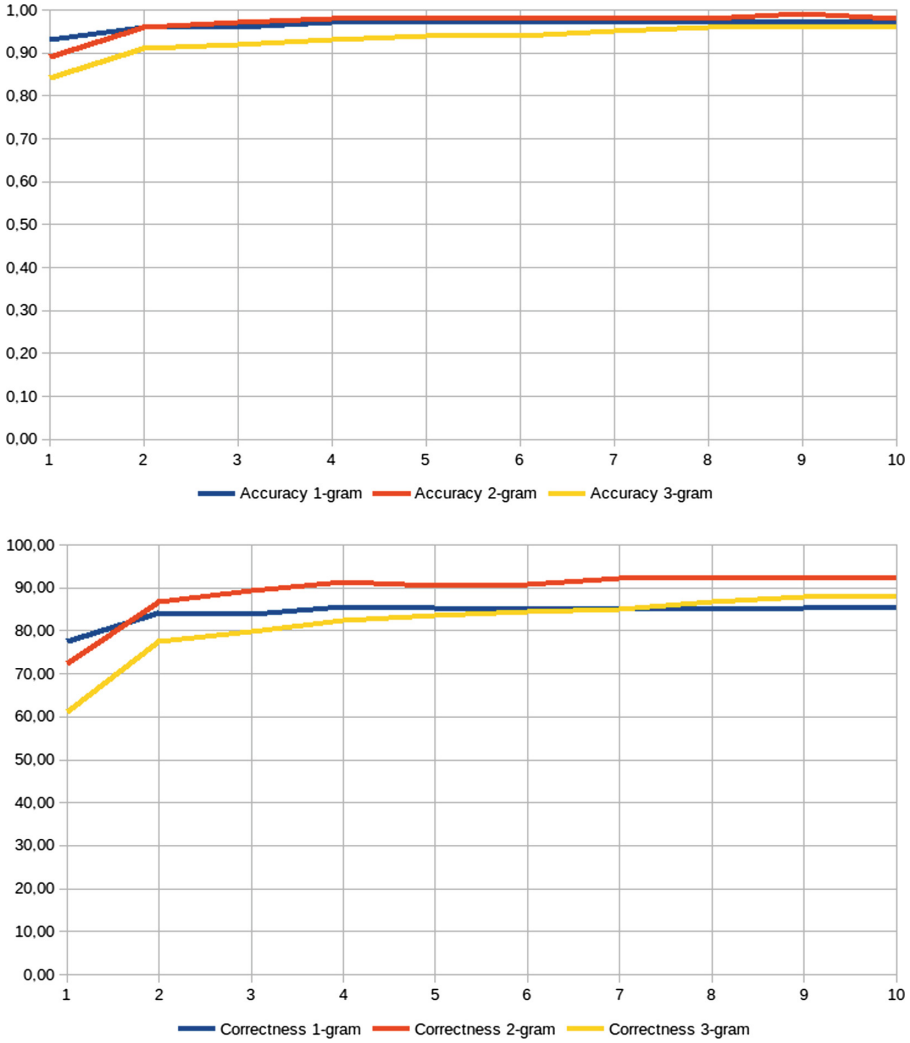

Fig. 2. Accuracy (top) and correctness (bottom) for predictions.

more, the best performance is obtained by 2-grams, albeit we would expect that 3-grams convey more information. Even more, while after processing just 1 text 1-grams perform best, 3-grams never win the competition. The 4-text threshold also brings both accuracy and correctness of 2-grams above 90%, and this is only slightly improved after processing all the other texts. Specifically concerning accuracy, some improvement (from 98% to 99%) is obtained only after processing 9 texts, but it is unreliable, since it is immediately followed (when processing 10 texts) by a drop in performance, that goes back to the value obtained for 8 texts. Given these results and considerations, we decided to focus on the 4-texts

Table 3. Predictions statistics: best for language

Language	1-gram				2-gram				3-gram			
	Acc		C		Acc		C		Acc		C	
	Text	Value	Text	Value	Text	Value	Text	Value	Text	Value	Text	Value
English	2	0.98	9	95.4	2	0.99	2	95	10	0.98	10	95.3
French	9	0.96	9	80.3	6	0.98	9	91.5	7	0.97	9	90.9
German	6	1	5	96.8	4	1	10	97.5	9	0.99	10	95.7
Italian	2	0.98	2	85.5	3	0.98	9	91.2	8	0.96	9	88.2
Portuguese	9	0.97	9	83.7	4	0.98	4	91.7	5	0.95	6	87.3
Squinzanese	1	0.96	7	83.6	9	0.98	9	88.8	9	0.93	10	75.6
Spanish	4	0.97	4	85.2	7	0.99	7	94.7	9	0.97	9	87.7

Table 4. Predictions statistics: average

n	N	*avg*	*avg*	Eng		Fre		Ger		Ita		Por		Squ		Spa	
		Acc	C	Acc	C	Acc	C	Acc	C	Acc	C	Acc	C	Acc	C	Acc	C
1	4	**0.97**	**85.58**	0.98	93.3	0.94	75.8	1	96.6	0.97	83	0.96	82.7	0.96	82.5	0.97	85.2
2	9	**0.99**	**92.33**	0.99	93.1	0.98	91.5	1	97.3	0.98	91.2	0.98	89.9	0.98	88.8	0.99	94.6
3	10	**0.96**	**87.91**	0.98	95.3	0.96	90.9	0.99	95.7	0.95	87.1	0.94	83.1	0.93	75.6	0.97	87.7

setting for 1-grams, 9-texts setting for 2-grams and 10-texts setting for 3-grams, which provided the best average performance on the overall set of languages.

Table 3 summarizes the performance on language identification, cast as a process prediction task, for a few settings selected as more relevant. U and W are not reported, since the former is always negligible (around 2.6%) and thus the latter is almost complementary to C. Looking at the detailed results, independently of the specific values, we see that in all settings German is best recognized, followed by English. This suggests that anglo-german languages are more characterized. Among latin languages, Spanish recognition is above average for 2-grams and mostly above average for 3-grams, then Italian has good performance on accuracy for 1-grams and Portuguese has some good performances on accuracy and/or correctness for 3-grams. The worst performance (Squinzanese) is still above 90% accuracy and 80% correctness for 2-grams.

Table 4 reports, for each n-gram, the number N of texts that provided the best average value for *Acc* and C, along with the performance on single languages. As regards Latin languages performances, they are close to average values; for each n-gram setting of French and English, their performances exceed the average values, while Squinzanese still has the worst performance.

To assign a language to a sentence, we used the last prediction in the sentence's n-grams. This quite basic decision rule allowed us to obtain 94% for (1, 4), 98% for (2, 9) and 92% for (3, 10) in predictive accuracy. Statistical significance of the results is supported by the size of our test set, that includes 556 sentences overall, compared to the overall population including 3083 sentences, so that the tolerance interval is ± 1.16 at 95% confidence, and ± 1.52 at 99% confidence.

The best performance, obtained using 2-grams, is comparable or even better than most other approaches in the current state-of-the art [11,12,19]. [11] works on 12 languages, reaching an average accuracy of about 91% overall and the following accuracy on languages we also considered: English 100%, French 99%, German 96%, Italian 80%, Portuguese 95%, Spanish 87%. [12] reports results on 4 languages only, 2 of which using different alphabets: Serbian, English, Chinese and Arabic. [19] works on 18 languages, reaching the following performance on languages we also considered: English 96% (97/101), French 83% (90/108), German 94% (94/100), Italian 98% (95/97), Latin 95% (100/105), Portuguese 92% (96/104), Spanish 95% (79/83). We are slightly worse than the Naive Bayes approach [1], which reached 100% but considering just 5 languages.

Additionally, our approach has several advantages over those solution: it can learn models using a few hundred sentences instead of thousands; and, most importantly, it is incremental both as regards training examples (i.e., it can refine and improve models as long as new training data are available, without the need to start from scratch each time) and as regards the target classes (i.e., we learn independent models for each language, and thus new languages can be added at any moment without affecting the existing models and without the need to learn them from scratch). Moreover, there is room for us to improve our performance in several directions: we can use the data gained during the whole path of a sentence to refine our decision rule; we may use WoMan's denoise feature to simplify the models by removing irrelevant components that possibly lower performance; we may use WoMan's model analysis features to identify the most characterizing or discriminating components in the models, and focus on these components hopefully improve performance.

5 Conclusions and Future Work

Several high-level tasks in the management of Digital Libraries require the application of Natural Language Processing (NLP) techniques. In turn, most NLP solutions are based on linguistic resources that are costly to produce, and so motivate research for automated ways to build them. In particular, Language Identification is a crucial NLP task, that is preliminary to almost all the others, since different linguistic resources must be used for different languages. On the other hand, for many languages that may be of interest in Digital Libraries, tools and resources for carrying out language identification are not available. This motivates the research for automatic approaches to do this.

While traditionally exploited for checking process enactment conformance, process models may be used to make predictions about a partial execution of a process. This paper proposes a way to see a text as a process, casts the Language Identification task as a process prediction problem, and investigates process mining and management approaches as a possible solution. Experimental results show very good performance. Also, some outcomes are not obvious. For instance, processing more texts does not always improve performance significantly. Also, 2-grams are the best, and 3 g are the worst, even if the latter

embed more information than the former. These results suggest several directions for future research. First, we will check if and how the performance changes when adding more languages, especially with ones who are completely different from each other. Also, we will check whether performance can be improved by suitably selecting more relevant components of the models and by refining our decision rule.

Acknowledgments. This work was partially funded by the Italian PON 2007–2013 project PON02_00563_3489339 'Puglia@Service'.

References

1. Ahmed, B., Cha, S.H., Tappert, C.: Language identification from text using n-gram based cumulative frequency addition. In: Proceedings of Student/Faculty Research Day, p. 12-1. CSIS, Pace University (2004)
2. Brown, P.F., deSouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Comput. Linguist. **18**(4), 467–479 (1992)
3. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Proceedings of 3rd Annual Symposium on Document Analysis and Information Retrieval (SDAIR 1994), pp. 161–175 (1994)
4. Ferilli, S.: WoMan: logic-based workflow learning and management. IEEE Trans. Syst. Man Cybern. Syst. **44**, 744–756 (2014)
5. Ferilli, S., Esposito, F.: A logic framework for incremental learning of process models. Fundamenta Informaticae **128**, 413–443 (2013)
6. Ferilli, S., Esposito, F., Grieco, D.: Automatic learning of linguistic resources for stopword removal and stemming from text. Procedia Comput. Sci. **38**(C), 116–123 (2014)
7. Ferilli, S.: The WoMan formalism for expressing process models. In: Perner, P. (ed.) ICDM 2016. LNCS, vol. 9728, pp. 363–378. Springer, Cham (2016). doi:10.1007/978-3-319-41561-1_27
8. Ferilli, S.: Natural language processing. In: Ferilli, S. (ed.) Automatic Digital Document Processing. Advances in Pattern Recognition, pp. 131–155. Springer, Cham (2015). doi:10.1007/978-0-85729-198-1_6
9. van der Aalst, W., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28108-2_19
10. Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., Wu, Y.: Exploring the limits of language modeling (2016)
11. Martins, B., Silva, M.: Language identification in web pages. In: Proceedings of the 2005 ACM symposium on Applied Computing, pp. 764–768. ACM (2005)
12. Mathew, T.: Text categorization using n-grams and hidden Markov models (2006). http://www.slideshare.net/thomas_a_mathew/text-categorization-using-ngrams-and-hiddenmarkovmodel
13. Messito, A.: Cuntame nnu cuntu! PhotoCity (2014)
14. Muggleton, S.: Inductive logic programming. New Gener. Comput. **8**(4), 295–318 (1991)
15. Nagarajan, T., Murthy, H.: Language identification using parallel syllable like unit recognition. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004), vol. 1, p. I-401. IEEE (2004)

16. Pesic, M., van der Aalst, W.M.P.: A declarative approach for flexible business processes management. In: Eder, J., Dustdar, S. (eds.) BPM 2006. LNCS, vol. 4103, pp. 169–180. Springer, Heidelberg (2006). doi:10.1007/11837862_18
17. Pierce, J.: Symbols, Signals and Noise: The Nature and Process of Communication. Harper, New York (1961)
18. Rotella, F., Leuzzi, F., Ferilli, S.: Learning and exploiting concept networks with connektion. Appl. Intell. **42**(1), 87–111 (2015)
19. Sibun, P., Reynar, J.C.: Language identification: examining the issues (1996)
20. Vatanen, T., Väyrynen, J.J., Virpioja, S.: Language identification of short text segments with n-gram models. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010). European Language Resources Association (ELRA), Valletta, May 2010
21. 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)