J Intell Inf Syst (2016) 46:1–19
DOI 10.1007/s10844-014-0349-9

A sentence structure-based approach to unsupervised author identification

Stefano Ferilli

Received: 27 May 2014 / Revised: 26 November 2014 / Accepted: 27 November 2014/
Published online: 19 December 2014

Abstract Assessing whether two documents were written by the same author is a crucial task, especially in the Internet age, with possible applications to philology and forensics. The problem has been tackled in the literature by exploiting frequency-based approaches, numeric techniques or writing style analysis. Focusing on this last perspective, this paper proposes a novel technique that takes into account the structure of sentences, assuming that it is strictly related to the author's writing style. Specifically, a (collection of) text(s) in natural language written by a given author is translated into a set of First-Order Logic descriptions, and a model of the author's writing habits is obtained as the result of clustering these descriptions. Then, if an overlapping exists between the models of a known author and of an unknown one, the conclusion can be drawn that they are the same person. Among the advantages of this approach, it does not need a training phase, and performs well also on short texts and/or small collections.

Keywords Author identification · Natural language processing · Clustering

1 Introduction

Nowadays, electronic publishing facilities and the spread of the Internet have made document production faster and easier than ever. Correspondingly, the number of plagiarism cases and of documents of unknown author has increased. Thus, author identification has become a primary issue in several contexts. Unfortunately, it has also become much more challenging than in the past, because in the last decades people's speaking and writing habits have changed significantly, as a consequence of dramatic changes in communication technology. Indeed, natural language is an extremely flexible tool, which can be adapted

S. Ferilli (✉)
Dipartimento di Informatica – Centro Interdipartimentale per la Logica e sue Applicazioni,
Università di Bari Bari, Italy
e-mail: stefano.ferilli@uniba.it

to convey feelings and concepts that continuously change due to social and technological evolution. In this landscape, the huge number of available documents and writers, and the variability of their writing styles, prevent the application of traditional (manual) approaches to author identification.

Authorship attribution is a well-defined task: “given a (text) document, identify the person who wrote it”. Nevertheless, it is amenable to many variations: given a document, determine a profile of the author; given a pair of documents, determine whether they were written by the same author; given a document, determine which parts of it were written by a specific person. Its motivations are also quite clear: it is useful to settle controversies about copyright ownership and in some cases, e.g. in areas such as law and journalism, knowing the author’s identity may save lives. Several solutions have been proposed in the literature to tackle the problem, exploiting frequency-based approaches, numeric techniques and writing style analysis. Many of these proposals use Machine Learning to obtain more flexibility (Argamon et al. 2003; Diederich et al. 2003; Lowe and Matthews 1995; Tweedie et al. 1996).

The most common approach for testing candidate algorithms is to cast the problem as a text classification task: given a (small) set of known authors, and sample documents for each of them, the aim is to assess if any of those authors wrote a ‘questioned’ document of unknown authorship. A special case, occurring in many real-world domains such as professional forensic linguistics, is: given a set of documents written by a single author, and a questioned document, determine whether the questioned document was written by that particular author or not. This paper focuses on the latter setting, with the additional requirement that also the number of ‘known’ documents can be very small (no more than 10, possibly just one). In the following, the known author (and the corresponding texts), used as a reference, will be called the *base*, while the unknown author (and the corresponding text), that must be classified, will be called the *target*.

Capturing the peculiarities of an author is not trivial. It requires a deep understanding of many aspects of his behavior, and involves modeling his literary style. This is in itself a hard problem, due to the intrinsic ambiguity and flexibility of natural language and to the huge amount of common sense and linguistic/conceptual background knowledge needed to switch from the purely syntactic level of a text to the underlying semantics. Moreover, also psychological and cultural aspects play a fundamental role in determining the text content. Traditional propositional approaches are not able to handle the whole complex network of (often hidden) relationships between the events and/or objects involved in a text.

Conversely, the relational setting provides more representational power. It allows to handle the complex syntactic structures underlying texts in natural language and to mine complex patterns from them. Leveraging these opportunities, the approach proposed in this paper extracts the typed syntactic dependencies among the components of each sentence in the text, and formally expresses them in First-Order Logic (FOL for short). In this way, structured patterns are obtained from plain text, on which automatic (relational) processing techniques can be applied to model the author’s style. Then, the resulting model can be used to classify new documents and decide whether it is likely that they were written by the same author or not.

This work is organized as follows. The next section introduces the text processing system used to ingest the text and transform it into a structured representation for the author modeling and identification algorithms described in Section 3. Then, Section 4 describes and compares related work, after which the proposed approach is evaluated in Section 5. The final section concludes the paper and outlines future work directions.

2 Overview and preliminaries

Text in natural language is a complex kind of data. Indeed, as regards the form, the syntax is not trivial, and, as regards the content, it conveys high-level conceptual information that is not always explicitly and objectively expressed. Both these aspects are strictly related to the culture, feelings and objectives of the authors, which are partly expressed by their writing style. This rich amount of information might be precious for supporting the author identification task. In order to exploit it as much as possible, one should be able to transform it into a more standard and machine-processable form. This requires the availability of suitable pre-processing techniques.

In a nutshell, the proposed approach works as follows. Given a (set of) text(s) from a given author, a structural description of each sentence is extracted. The assumption here is that this description can explicitly capture (part of) the complex patterns representing the author's style. These descriptions can be clustered, provided that a similarity measure for relational representations is available. Note that this prevents the possibility of using the relational descriptions at the level of whole document. In fact, we assume to have just one target document, and possibly just one base document, and clustering one document would not make sense. Since we do not know the length of a document, nor whether it is organized into paragraphs, sentences are the first useful granularity level. If agglomerative clustering is adopted, a stopping criterion for the grouping procedure is also needed, and it is desirable that such a criterion can be determined automatically. The resulting set of clusters can be seen as a model of the author's writing style, that describes possible ways in which he composes his sentences. So, applying the above procedure to both the base and the target text(s) yields two corresponding models, and an answer to the author identification problem can be obtained as a result of the comparison between these models. The underlying idea is that, if the writing habits expressed by the target model can be brought back to the base model, then one may conclude that the author is the same.

2.1 ConNeKTion

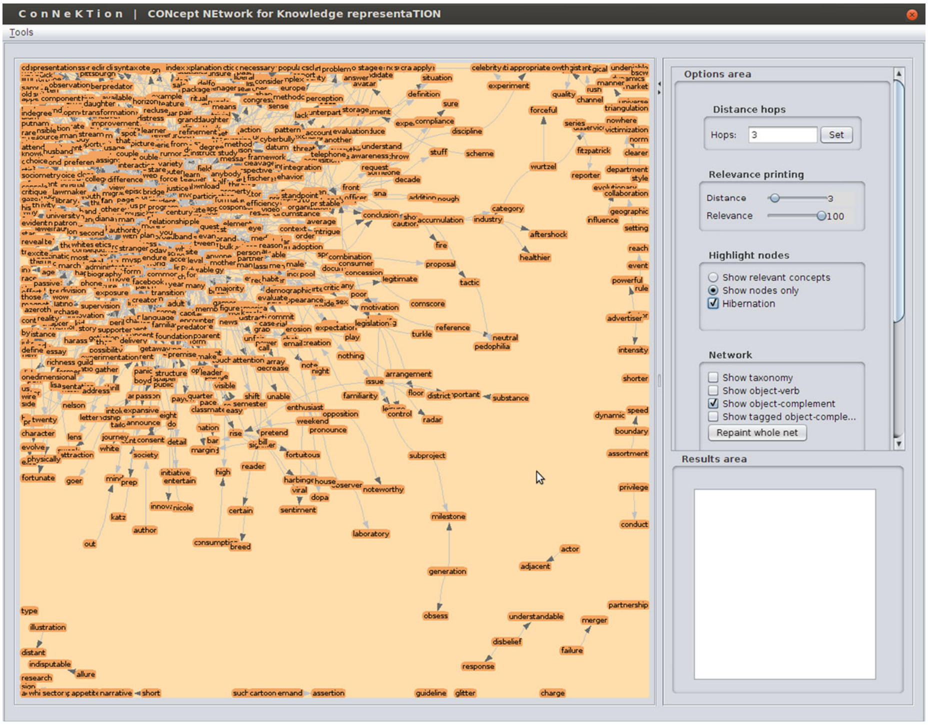
ConNeKTion (acronym for 'CONcept NEtwork for Knowledge representaTION') (Ferilli et al. 2011, Leuzzi et al. 2013) is a system that aims at partially simulating some human abilities in text understanding and concept formation. Its main feature is the ability to build a network that expresses the concepts underlying a text collection, along with their definitions and several kinds of relationships among them. However, it also provides several other functionalities, both as steps towards this main objective and as complements to it. For instance, it embeds techniques for assessing the relevance of the extracted concepts. It associates concepts to terms, and can build formal descriptions of these concepts. These descriptions can be flat (attribute-value) or relational. In addition to the explicit (taxonomic or non-taxonomic) relationships among concepts extracted from the text, it can enrich the network with further taxonomic relationships obtained by clustering and/or generalizing concepts. It can exploit the learned concept network to carry out different kinds of reasoning 'by association', that looks for possible indirect connections between two concepts.

It can apply multi-perspective approaches to the problem of identifying relevant keywords in the text, and may help the user in retrieving useful information. Some of these abilities are involved in the author identification procedure, as described in the next subsections. The main interface of ConNeKTion is shown in Fig. 1, where the concept network is displayed in the panel on the left, and the author identification functionality can be activated from the **Tools** menu.

2.2 Text pre-processing

A fundamental functionality of ConNeKTion, needed in our author identification procedure, is the extraction of a relational representation of the syntactic features of sentences. Several steps concur to build this representation, progressively clarifying different aspects of the input text:

Term Normalization Words in sentences are inflected, according to the rules of the language grammar. However, inflection is nearly irrelevant for identifying the concepts expressed by terms. So, it is useful to select as a reference a normalized version of each word. ConNeKTion uses lemmatization instead of stemming, which may allow to distinguish the grammatical role of the word and is more comfortable to read for humans.


ConNeKTion | CONcept NEtwork for Knowledge representaTION

Tools

Options area

Distance hops

Hops: 3 Set

Relevance printing

Distance 3

Relevance 100

Highlight nodes

○ Show relevant concepts
● Show nodes only
☑ Hibernation

Network

□ Show taxonomy
□ Show object-verb
☑ Show object-complement
□ Show tagged object-comple...

Repaint whole net

Results area

Fig. 1 The main interface of ConNeKTion

Collocation Extraction Collocations are linguistic expressions consisting of two or more words that denote a compound concept. Among various solutions proposed in the literature, ConNeKTion adopts an approach based on Pointwise Mutual Information (PMI) (Church and Hanks 1990), associated to the difference in likelihood between the compound term and the single component terms.

Anaphora Resolution Anaphora are references to concepts cited elsewhere in the same text, usually expressed by pronouns. So, to make a sentence autonomous, it is necessary to identify the referred concept and replace each pronoun by the explicit concept. ConNeKTion uses a modified version of the Resolution of Anaphora Procedure (RAP) based on Part-of-Speech tagging (Qiu et al. 2004), and specifically its JavaRAP implementation.

Parsing A significant improvement in text understanding can be obtained by stepping up from the purely lexical level to the syntactic level. The syntactic relationships between subjects, verbs and (direct or indirect) objects in a sentence can be represented in a tree that reproduces its phrase structure. ConNeKTion extracts the tree-like structure of each sentence in a text using the *Stanford Parser* (Klein and Manning 2003), a well-known multilingual tool.

Dependencies Extraction Based on the parse tree of a sentence, a set of grammatical (typed) dependencies among the sentence components can be identified. These dependencies can be expressed as binary relations between pairs of words, the former of which represents the governor of the grammatical relation, and the latter its dependent. ConNeKTion uses the *Stanford Dependencies* tool (De Marneffe et al. 2006) for this. This tool is currently available only for English, which actually limits the application of ConNeKTion to texts in this language only. However, the proposed strategy is general and applicable, in principle, to any language for which such dependencies can be extracted.

2.3 Representation formalism

After the structure of sentences has been extracted, a suitable representation formalism and language is needed to express them. ConNeKTion adopts Horn clause logic (Lloyd 1987). A clause is expressed in Prolog format as:

$$l_0 :- l_1, \ldots, l_n.$$

It represents an implication where l_0 is the conclusion (called the *head*) and $l_1, \ldots, l_n$ (called the *body* of the clause) denotes the conjunction of premises. In FOL, the l_i's are atoms, i.e. predicates applied to their arguments, expressing properties of, or relationships among, their arguments.

For our purposes, the predicate in the head is used to label the sentence that is described in the body. Each sentence is associated to a unique identifier $idSentence$. So, the translation of a sentence into a Horn clause takes the following form:

$$sentence(IdSentence) :- l_1, \ldots, l_n.$$

where the body expresses the relationships between the terms of the sentence, their grammatical roles and the phrases to which they belong. The atoms are built on the following predicates:

- *phrase(Tag,IdSentence,Pos)* represents a sentence constituent with Tag denoting the type of phrase (e.g., *NP* for Noun Phrase, *VP* for Verb Phrase, *S* for Sentence, etc.) and Pos the term position in the phrase;

- *term(IdSentence,Pos,Lemma,PosTag)* denotes a single term whose position in the sentence is Pos, specifying its lemma $Lemma$ and its PoS tag (e.g., N for Noun, V for Verb, P for Preposition, etc.) $PosTag$;
- *sd(IdSentence,Type,PosGov,PosDep)* expresses the existence of a grammatical relation of type $Type$ (e.g., *dobj* for direct object, *subj* for subject, etc.) between the governor word in position $PosGov$ and the dependent word in position $PosDep$.

For instance, the following sentence:

> The main purpose of this chapter is to explain in more detail some of the problems that arise in connection with what the lawyers call "prior art"–meaning, in the case at hand, systems that use the traditional direct-image approach to implementation"

would be translated in a Horn clause as shown in Table 1.

2.4 Similarity measure

The adopted strategy for assessing the similarity between Horn clauses, presented in Ferilli et al. (2009a), is based on the following formula:

$$\mathrm{sf}\left(i', i''\right) = \mathrm{sf}(n, l, m) = \alpha \frac{l+1}{l+n+2} + (1-\alpha)\frac{l+1}{l+m+2}$$

Table 1 An excerpt of the translation of a sentence in First-Order Logic

```
sentence(39) :– phrase(s,39,1), . . . , phrase(s,39,48),
    phrase(np,39,1), phrase(np,39,2), phrase(np,39,3), phrase(pp,39,4),
    phrase(pp,39,5), phrase(np,39,5), phrase(pp,39,6), phrase(np,39,6),
    phrase(vp,39,7), phrase(vp,39,8), phrase(vp,39,9), phrase(vp,39,10),
    phrase(pp,39,10), phrase(vp,39,11), phrase(pp,39,11), phrase(np,39,11),
    phrase(vp,39,12), phrase(pp,39,12), phrase(np,39,12), . . . ,
    phrase(sbar,39,17), . . . , phrase(sbar,39,47), phrase(vp,39,47), phrase(pp,39,47),
    phrase(np,39,47), . . . ,
    term(39,1,'the',dt), term(39,2,'main',jj), term(39,3,'purpose',nn),
    sd(39,prep(of),3,6), term(39,5,'this',dt), term(39,6,'chapter',nn),
    term(39,7,'be',vbz), term(39,8,'to',to), term(39,9,'explain',vb),
    sd(39,prep(in),9,12), term(39,11,'more',jjr), term(39,12,'detail',nn),
    term(39,13,'some',dt), sd(39,prep(of),13,16), term(39,15,'the',dt),
    term(39,16,'problem',nns), term(39,18,'arise',vbp), sd(39,prep(in),18,20),
    term(39,20,'connection',nn), term(39,22,'what',wp),
    term(39,23,'the',dt), term(39,24,'lawyer',nns), term(39,25,'call',vbp),
    term(39,27,'prior',jj), term(39,28,'art',nn), term(39,31,'mean',vbg),
    term(39,34,'the',dt), term(39,35,'case',nn), sd(39,prep(at),35,37),
    term(39,37,'hand',nn), sd(39,prep(with),18,39), term(39,39,'system',nns),
    term(39,41,'use',vbp), term(39,42,'the',dt), term(39,43,'traditional',jj),
    term(39,44,'directimage',jj), term(39,45,'approach',nn), sd(39,prep(to),41,47),
    term(39,47,'implementation',nn),
    sd(39,amod,12,11), sd(39,nsubj,18,16), sd(39,dobj,9,13), sd(39,dep,39,25),
    sd(39,xcomp,25,31), sd(39,det,24,23), sd(39,nsubj,25,24), sd(39,rcmod,16,18),
    sd(39,det,3,1), sd(39,rel,25,22), sd(39,dobj,41,45), sd(39,nsubj,41,39),
    sd(39,amod,3,2), sd(39,det,35,34), sd(39,det,6,5), sd(39,prep(in),39,35),
    sd(39,nsubj,7,3), sd(39,det,45,42), sd(39,amod,28,27), sd(39,xsubj,9,3),
    sd(39,rcmod,39,41), sd(39,dobj,25,28), sd(39,aux,9,8), sd(39,amod,45,43),
    sd(39,amod,45,44), sd(39,xcomp,7,9), sd(39,det,16,15).
```

where:

- i' and i'' are the two items under comparison;
- n represents the information carried by i' but not by i'';
- l is the common information between i' and i'';
- m is the information carried by i'' but not by i';
- α is a weight that determines the importance of i' with respect to i'' (0.5 means equal importance).

More precisely, the clause similarity assessment procedure carries out a layered evaluation that, starting from simpler components, proceeds towards higher-level ones repeatedly applying the above similarity formula. At each level, it exploits the information coming from lower levels and extends it with new features. At the basic level, terms (i.e., constants in our setting) are considered, that represent objects in the world. Their similarity is based on their properties (expressed by unary predicates) and roles (expressed by their position as arguments in n-ary predicates). The next level involves atoms built on n-ary predicates, that represent relationships among objects. The similarity of two atoms is based on their "star" (the multiset of predicates corresponding to atoms directly linked to them in the clause body) and on the average similarity of their arguments. Then, the similarity of sequences of atoms is based on the length of their compatible initial subsequence and on the average similarity of the atoms appearing in such a subsequence. Finally, the similarity of clauses is computed according to their least general generalization, considering how many atoms and terms they have in common and their corresponding lower-level similarities. Since each of the four components ranges into $]0, 1[$, their sum ranges into $]0, 4[$. If needed, it can be normalized to $]0, 1[$.

An extension of this similarity measure, presented in Ferilli et al. (2009b), allows one to take into account also taxonomic information that may be associated to the terms in the description. This is particularly relevant when the clauses represent text, as in our case, because the words in the text can be associated to underlying concepts, and a taxonomy may be exploited (if available) to leverage further implicit relationships among these concepts. Using this additional perspective causes the overall similarity to range into $]0, 5[$ (which, again, can be normalized to $]0, 1[$ if needed). Note that the extremes of the intervals are not allowed, and that any similarity value returned by this measure can be turned into a distance value by just complementing it with respect to the maximum similarity value.

3 Relational author identification

After obtaining a relational description for each sentence in the available documents (both base and target ones), the author identification procedure evaluates the similarity between all pairs of sentences. As a result, an upper triangular matrix of similarities between all possible pairs of sentences is obtained. As shown in Fig. 2, three portions of this matrix are particularly relevant. The top-left sub-matrix (highlighted using a striped texture) contains the similarity scores between pairs of sentences both belonging to known documents (base). The bottom-right portion (highlighted using solid gray texture) includes the similarities between pairs of sentences both belonging to the unknown document (target). Finally, the top-right part reports the similarity scores between sentences from known and unknown documents.

Algorithm 1 *pairwiseClustering(M,T)* : Relational pairwise clustering

```
Require: $M$ : similarity matrix for observations $O$; $T$ : similarity threshold
  $averages \leftarrow \emptyset$
  $\mathcal{C} \leftarrow \{\ \{o\} \mid o \in O\}$ /* initial set of singleton clusters */
  $merged \leftarrow true$
  while $merged$ do
      $merged \leftarrow false$
      $candidates \leftarrow \{(C', C'') \mid C', C'' \in \mathcal{C}, C' \neq C'' \wedge \text{completeLink}(M, C', C'') \geq T\}$
      if $candidates \neq \emptyset$ then
          $(\overline{C'}, \overline{C''}) = \arg\max_{(C',C'') \in candidates} \text{avgSim}(M, C', C'')$
          merge$(\overline{C'}, \overline{C''}, \mathcal{C})$
          $merged \leftarrow true$
      end if
  end while
Ensure: $\mathcal{C}$ /* set of clusters */
```

completeLink$(M, C', C'') = \max_{c' \in C', c'' \in C''} M(c', c'')$: computes the complete link value for the pair of clusters (C', C'') according to similarity matrix M
avgSim$(M, C', C'') = \sum_{c' \in C', c'' \in C''} M(c', c'')/(|C'| \cdot |C''|)$: computes the average similarity between pairs of elements taken from C' and C'' according to similarity matrix M
merge$(C', C'', \mathcal{C})$: updates the set of clusters $\mathcal{C}$ by merging clusters $C', C'' \in \mathcal{C}$ into a single cluster

Then, two separate agglomerative clustering procedures are applied to the base and target sentences, respectively, according to Algorithm 1. Initially, each description yields a different singleton cluster. Then, the procedure works by iteratively finding the next pair of clusters to be merged according to a *complete link* strategy. This strategy states that “two clusters can be merged if the similarity of their farthest items is greater than a given threshold” (where the threshold takes values in the same range as the similarity function). In our algorithms, this computation is carried out by function *completeLink*(M, C', C''). Note that more than one pair might satisfy such a requirement. Since the ordering in which the clusters are merged affects the final model, in these cases the procedure merges the pair of clusters yielding the largest average similarity among all pairs of their elements (i.e., sentences). This computation is carried out by function *avgSim*(M, C', C'') in Algorithm 1. When the loop (and the algorithm) terminates, the resulting set of clusters is considered as a *model* of the writing style underlying the clustered documents. Since each cluster includes sentences

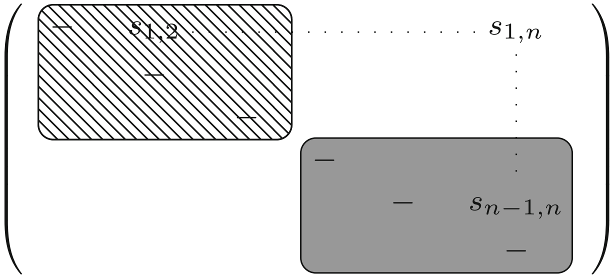


Fig. 2 Global similarity matrix. Each $s_{i,j}$ represents the similarity between sentences i and j calculated as explained in Section 2.4

having similar structure, the assumption is that each cluster identifies a particular kind of expressions used by the author.

Algorithm 2 *getBestModel*(M, T_{min}, T_{max}) : Best model identification

```
Require: $M$ : similarity matrix; $T_{min}$ : starting threshold, $T_{max}$ : maximum threshold to be
  attempted
  $t \leftarrow T_{min}$
  $models \leftarrow \langle\rangle$ /* empty list of candidate models */
  repeat
    $models \leftarrow \langle(\text{pairwiseClustering}(M, t), t) \mid models\rangle$
    $t \leftarrow t + \epsilon$
  until $t > T_{max}$
  $models = \langle(\mathcal{C}, t) \mid restModels\rangle$
  $maxGap \leftarrow 0$
  $bestModel \leftarrow (\mathcal{C}, t)$ /* current best model and associated threshold */
  $models \leftarrow restModels$
  $precModel \leftarrow (\mathcal{C}, t)$
  while $models = \langle(\mathcal{C}'', t'') \mid restModels\rangle$ /* nonempty list of candidate models */ do
    $precModel = (\mathcal{C}', t')$
    $gap \leftarrow |\mathcal{C}''|/|\mathcal{C}'|$
    if $maxGap < gap$ then
      $maxGap \leftarrow gap$
      $bestModel \leftarrow precModel$
    end if
    $models \leftarrow restModels$
    $precModel \leftarrow (\mathcal{C}'', t'')$
  end while
Ensure: $bestModel$ /* best model and associated threshold */
```

where $\langle F \mid R\rangle$ denotes a (nonempty) list having F as its first item, followed by a (possibly empty) list of items R

Now, the agglomerative clustering algorithm needs a threshold T to decide when to stop. This raises the question about how to properly determine such a threshold for each model that is being computed. Since it is unlikely that a single fixed threshold works well for all possible cases, we need a flexible approach that can determine a tailored threshold for each specific set of input data (for the base and for the target data). Larger thresholds make merging more difficult, and as a consequence yield more clusters. So, we apply clustering several times with increasingly larger thresholds, and consider the ratio of the number of clusters yielded by consecutive values of the threshold. The assumption is that a small difference means that clustering is proceeding smoothly, while large differences suggest that quite different kinds of structures have been merged. In the end, we take as a model the set of clusters returned by the threshold associated to the largest difference, as shown in Algorithm 2. In case of ties, the model including less clusters is considered, because the same difference value obtained on less clusters indicates more change than if obtained on more clusters.

Given a minimum and a maximum threshold, Algorithm 2 exploits Algorithm 1 to run clustering for each threshold in that range, with an increment step ϵ. The smaller ϵ, the more sensitive (but also the more time-consuming) is the threshold assessment procedure ($\epsilon = 0.005$ was used in our experiments). The **repeat** loop ensures that at least one clustering (i.e., one model) is obtained. As long as the models $\mathcal{C}_i$ for progressive thresholds t_i are generated, the pairs $(\mathcal{C}_i, t_i)$ are collected in a list. For n thresholds, the list of models (and associated thresholds) $\langle(\mathcal{C}_1, t_1), ..., (\mathcal{C}_n, t_n)\rangle$ is obtained. Then, the first item in the list is taken as the initial best model, and the rest of the list (if any) is scanned to find the widest gap in number of clusters according to the strategy outlined above. Since the number of clusters in the i-th model is $|\mathcal{C}_i|$, the gap function for each pair of subsequent models in the list can be defined as:

$$g(i) = \frac{|\mathcal{C}_{i+1}|}{|\mathcal{C}_i|}$$

for $0 < i < n$. Now, the widest gap is associated to the following index:

$$\bar{i} = \arg\max_{0<i<n} g(i)$$

This allows to determine the desired model $\mathcal{C}_{\bar{i}}$, and hence its associated threshold $t_{\bar{i}}$. Note that we take the model that was computed *before* the widest gap occurred (i.e., the one associated to index $\bar{i}$, not $\bar{i}+1$).

Algorithm 3 Author Identification

```
Require: $O_k$ : set of descriptions obtained from the known-author documents; $O_u$ : set of descriptions obtained from the unknown-author document; $T_{min}$ : starting threshold, $T_{max}$ : maximum threshold to be attempted
  $M \leftarrow \text{getSimilarities}(O_k \cup O_u)$
  $(\mathcal{C}_k, t_k) \leftarrow \text{getBestModel}(M\ |_{O_k}, T_{min}, T_{max})$
  $(\mathcal{C}_u, t_u) \leftarrow \text{getBestModel}(M\ |_{O_u}, T_{min}, T_{max})$
  if $|\mathcal{C}_k| \leq 3 \vee \text{items}(\mathcal{C}_k) \leq \text{items}(\mathcal{C}_u) * 0.2 \vee |\mathcal{C}_u| \leq 3 \vee \text{items}(\mathcal{C}_u) \leq \text{items}(\mathcal{C}_k) * 0.2$ then
      $answer \leftarrow$ ‘No decision’ /* gray zone */
  else
      $Score \leftarrow \frac{|\{C_u \in \mathcal{C}_u | \exists C_k \in \mathcal{C}_k \ni' \text{completeLink}(M, C_k, C_u) \leq \max(t_k, t_u)\}|}{|\mathcal{C}_u|}$
      if $Score \geq \tau$ then
          $answer \leftarrow$ ‘Same author’
      else
          $answer \leftarrow$ ‘Different authors’
      end if
  end if
Ensure: $answer$ /* classification outcome */
```

where $M\ |_S$ denotes the portion of similarity matrix M referred to items in S, and:
completeLink(M, C', C'') : computes the complete link value for the pair of clusters (C', C'') according to similarity matrix M
getSimilarities(S) : returns the similarity matrix between all pairs of objects in the set S
items$(\mathcal{C}) = \sum_{C \in \mathcal{C}} |C|$: returns the number of items composing the clusters in model $\mathcal{C}$

Then, the classification phase may take place, and consists in checking whether there is sufficient overlapping between the target clusters and the base ones, as reported in Algorithm 3. The amount of overlapping (*Score*) is computed as the ratio of non-singleton clusters in the target model that can be merged with at least one non-singleton cluster in the base model, using the distances in the top-right sub-matrix and the complete link strategy. Singleton clusters are discarded because they include just one sentence, and a structure appearing in just one sentence is not significant to shape a writing style. The maximum of the thresholds associated to the base and target models is used as a threshold for the merge. The assumption is that, if two clusters can be merged, then the two authors are using similar linguistic constructs, which is a hint of their having similar writing styles (and thus of their being the same person). Finally, the author of the target is classified as being the same person as the author of the base if *Score* is greater than a predefined value $\tau \in [0, 1]$, representing the amount of overlap between writing styles that is considered as sufficient to claim identity of the two authors. The smaller τ, the less overlapping is required to classify the target as being written by the same author as the base. The larger τ, the more difficult it is that the two authors are considered to be the same person. Using $\tau = 1.0$ implements an extremely cautious behavior that encourages accurate classifications. This is actually a very strict constraint, that can be softened by taking smaller values for τ, at the cost of a less reliable classification.

The gray zone. While the proposed approach is able to build reliable models even using few data, it may be expected that there is a limit under which the available text is really too poor. In these cases, which we call the *gray zone*, the system might decline returning an answer to the author identification problem at hand. This is just what a human would do when he is not sufficiently sure about his decision, and has some connection with what happens in the Active Learning field (Settles 2010). We empirically identified some of these cases. For instance, we noticed that the outcome is sufficiently reliable only for problems whose models consist of at least 4 clusters. Thus, we assigned to the gray zone the cases involving models made up of just one, two or three clusters. Also, it turned out that the system's prediction might be unreliable when the overall number of sentences in the known documents is much smaller than the number of sentences in the unknown one. After trying different values in the range between 10 % and 30 %, we decided to assign to the gray zone the cases in which the number of clusters in a model is less than 20 % of the number of clusters in the other. These cases seem actually very odd, and likely to be difficult to handle also for humans. When encountering the gray zone, the models are considered as unreliable, and thus the approach does not try a classification (see Algorithm 3).

4 Related work

A significant amount of work on automated author identification has been carried out in the last decade. Researchers focused on different properties of text, the so-called *style markers*, to model the writing style using several labels and criteria. Generally speaking, the features can be divided into five main groups, depending on the type of source or on the level at which they are extracted: character-level, lexical, syntactic, semantic and application-specific.

Lexical and character-level features consider a text as a mere sequence of word-tokens or characters, respectively. Argamon et al. (2007) defines taxonomies of various semantic functions of selected words or phrases, from which 675 lexical features are derived to be used in stylistic text classification. While reaching interesting results, this approach exploits language-dependent expertise and the definition of the lexical features and taxonomies seems quite arbitrary. Seidman (2013) evaluates the similarity between the given documents and a number of external (impostor) documents. Then, the documents are classified as being written by the same author, if they turn out to be more similar to each other than to the impostors, in several trials exploiting different lexical feature sets. Zheng et al. (2006) builds a suffix tree representing all possible character n-grams up to a given value of n, and then extracts groups of such n-grams as features. The choice of n is critical: larger values allow to capture more information but increase the dimensionality of the representation, while with small values it might be impossible to learn a useful model. Compared to these approaches, our method can take into account the structural aspects of the text, that may be very relevant in determining a writing style, and does not need to extract arbitrary or fancy features, nor to set complex parameters, except for a few thresholds.

Syntactic features are based on the idea that each author unconsciously tends to use the same syntactic patterns when writing. Therefore, his model might be based on information about Part of Speech (PoS) tags, sentence structure and phrase structure. Approaches that adopt this perspective have two major problems: they need robust and accurate NLP tools to parse the texts, and they must deal with a huge number of extracted features (e.g., about 900,000 in van Halteren (2004)). Feng and Hirst (2013) defines a set of coherence-based and stylometric-based features. Since such features are applicable to English only, texts in other languages are first translated using the Google Translate service. This causes additional problems due to the noise introduced in the representation by the automatic translation. Compared to these approaches, our method does not use complex syntactic features, but just the structure of sentences as obtained from the parser, nor it requires the definition of stylometric features by experts. So, it is language-independent, except for the parsing step.

Semantic approaches rely on semantic dependencies between text components, obtained by using external resources, such as thesauri, taxonomies or ontologies. E.g., Mccarthy et al. (2006) exploits WordNet (Fellbaum 1998) to detect 'semantic' relationships between words. Compared to these approaches, our method is completely independent of external linguistic resources. This is important because, although the use of these resources can improve the system's results, they are not always available, especially for very specific domains, or their quality may be insufficient.

Finally, there are special-purpose approaches, that define application-specific metrics to better represent the text style in a given domain. Such metrics are based on the use of greetings and farewells in the messages, types of signatures, use of indentation, paragraph length, and so on (Li et al. 2006). Again, compared to these approaches, our method is completely general and domain-independent, and does not require any high-level knowledge acquisition and formalization.

A common feature to all of the above approaches is their using a flat (vectorial) representation of the documents/phrases. Even syntactic and semantic approaches, albeit starting from syntactic trees or word/concept graphs, subsequently create new flat features, losing in this way the relations embedded in the original texts. For example, Vilariño et al. (2013) builds graphs based on PoS sequences and then extracts sub-graph patterns. This graph-based representation tries to capture information about the sequence of words and/or PoS-tags in the sentences, from which extracting relevant relational features by means of a graph mining tool. However, all relational information is lost afterwards, when a feature vector for each document is built upon those features and used as an input for a Support Vector Machine classifier. Conversely, our method directly exploits the structure of sentences as expressed by their relational description.

An approach that preserves the phrase structure was proposed in Raghavan et al. (2010). After building a probabilistic context-free grammar (PCFG) for each author, each target document is assigned to the author whose PCFG produced the highest likelihood for it. While this approach takes into account the syntactic tree of the sentences, it needs many documents per author to learn reliable probabilities. Thus, it is practically not applicable under our constraint of having available just a small set of documents of only one author, for which our method was designed. Moreover, we believe that the exploitation of parse trees only is not enough to characterize the author's style, and that the syntactic relationships should be enriched with grammatical ones.

A final remarkable difference between our method and all of these works is that it is unsupervised, and hence it can compare two (sets of) texts without any prior learning step, which would introduce the need for a training set and would make the learned model strictly related to such a set.

5 Evaluation

The author identification procedure was evaluated using the dataset provided in the '9th Evaluation Lab on Uncovering Plagiarism, Authorship, and Social Software Misuse' (PAN), held as part of the CLEF 2013 conference. It involves 3 languages (English, Greek and Spanish), for each of which it provides 3 sub-datasets: a training dataset ('Training'), an early-bird evaluation dataset ('Test 1') and the complete evaluation dataset ('Test 2') that is a superset of 'Test 1'. While the proposed approach can be in principle applied to any language, as long as suitable NLP tools for it are available, in this evaluation the English problems only were considered, because the current version of ConNeKTion uses the Stanford NLP tools, that cannot deal with Greek and Spanish. The composition of the three sub-datasets for the English language is shown in Table 2, along with the abbreviation that will be used for each sub-dataset in the following.

A first experiment aimed at analyzing the behavior of the approach and at suitably tuning its parameters. The Training set only was used for this. Indeed, since the approach does not require a training phase, the training set can be used for testing purposes as well. Details about this portion of the English dataset, which are useful to understand the amount of information with which the system must deal in order to carry out its task, are reported in Table 3. The first columns report statistics on the dataset, and specifically, for each problem, the original identifier (*ID*), the number of known documents (#*d*), and, for both known and unknown documents, the number of clauses generated by the corresponding sentences (#*c*) and the average length of these sentences ($\bar{l}$). It can be noted that, on average, the known documents consist of many short sentences, while the unknown documents are made up of few long sentences. Problem 'EN23' is peculiar: the sum of all sentences in the known documents is half of the sentences in the unknown one. Finally, the last two columns report, respectively, the expected (*Exp.*) classification outcome for the unknown document, and the *Score* for that classification. Using hard classification (i.e., with required score $\tau = 1.0$ — each cluster in the target must merge with at least one cluster in the base) yields an accuracy of $7/10 = 0.7$, a precision of $3/4 = 0.75$ and a recall of $3/5 = 0.6$ (0.67 F1-measure). Softening the classification using $\tau = 0.9$, one additional correct (positive) answer is obtained (for problem EN23, where full merging was not obtained for just 0.08), raising the overall performance to $8/10 = 0.8$ accuracy, $4/5 = 0.8$ precision and $4/5 = 0.8$ recall (0.8 F1-measure). Note once more that EN23 is the problem with less sentences to be clustered, hence it is more complex.

The second experiment concerned the whole English dataset, and aimed at quantitatively evaluating the technique. Table 4 reports some statistics about the overall English dataset,

Table 2 PAN2013 English dataset composition

Sub-dataset	No. of problems	Pos	Neg	Abbrev.
Training	10	5	5	Tr
Test 1	20	9	11	$T1$
Test 2	30	14	16	$T2$
Total	40	19	21	Tot

Table 3 English training set details and outcomes

ID	Known docs			Unknown doc		Outcomes	
	$\#d$	$\#c$	$\bar{l}$	$\#c$	$\bar{l}$	Exp.	Score
EN04	4	261	121.06	62	136.60	Y	1.0
EN07	4	260	121.48	44	195.47	N	1.0
EN11	2	109	185.87	39	160.41	Y	1.0
EN13	3	109	156.99	65	134.65	N	0.6
EN18	5	274	154.25	53	165.49	Y	1.0
EN19	3	139	164.35	56	210.05	Y	0.37
EN21	2	109	210.89	24	269.21	N	0.67
EN23	2	51	217.29	97	277.29	Y	0.92
EN24	5	242	147.06	89	169.08	N	0.69
EN30	2	95	189.87	33	322.09	N	0.8
Average	3.2	164.9	153.6	56.2	198.7		

and specifically: the minimum (*min*), the average (*avg*) and the maximum (*max*) values for the number of documents (*# docs*), the number of clauses/sentences (*# clauses*) and the average length (*avg length*) of their relational descriptions for the corresponding problems. The figures are reported for both known and unknown documents, and are useful to understand the amount of information the system must deal with. We investigated how good the approach is both with and without using the gray zone. These two settings are referred to as *smoothed evaluation* and *boolean evaluation*, respectively, and denoted in the following, when needed, by subscripts s and b, respectively.

For each setting and sub-dataset, Table 5 reports the experimental results in terms of Accuracy ($acc = TPP + TNP$) and Error Rate ($err = FPP + FNP$), computed from the percentages of true positives ($TPP = TP/N$), true negatives ($TNP = TN/N$), false positives ($FPP = FP/N$) and false negatives ($FNP = FN/N$) over the total of N examples. Since in the smoothed evaluation the system did not always return a decision, in

Table 4 Dataset details

Set	# docs			# clauses			avg length		
	min	avg	max	min	avg	max	min	avg	max
Known docs									
Training	2	3.20	5	51	178.87	274	121.06	166.82	216.37
Test 1	3	4.45	9	29	146.79	329	107.35	218.24	322.82
Test 2	2	4.27	14	29	145.59	367	100.81	209.55	319.59
Overall	2	3.96	14	29	157.08	367	100.81	198.2	322.82
Unknown docs									
Training				24	56.10	96	134.65	194.22	322.09
Test 1				9	117.31	238	117.01	228.11	358.41
Test 2				9	56.00	301	110.29	213.85	351.18
Overall				9	76.47	301	110.29	212.06	358.41

Table 5 Comparison between boolean and smoothed setting (acc = Accuracy, err = Error Rate, ste = Sample Test Error, NC = not classified)

Type	boolean		smoothed						
Set	acc	err	acc	err	ste	NC	Δ_{acc}	Δ_{err}	Δ
Tr	0.7	0.3	0.7	0.1	0.2	0.2	0	0.2	0.1
$T1$	0.7	0.3	0.65	0.15	0.25	0.2	0.05	0.15	0.05
$T2$	0.47	0.53	0.44	0.26	0.41	0.30	0.03	0.27	0.13
Tot	0.53	0.47	0.50	0.22	0.36	0.28	0.03	0.25	0.11

order to give the reader a complete picture of the outcomes we report also the portion of test data that were not classified (NC) and the Sample Test Error ($ste = err_s + \Delta$) with a loss function that assigns them error cost 0.5 ($\Delta = 0.5 \cdot NC$). So, while in the boolean evaluation it holds $TPP + TNP + FPP + FNP = 1$, in the smoothed evaluation we have that $TPP + TNP + FPP + FNP + NC = 1$.

As regards the boolean setting, Table 5 shows that the results are good for subsets Tr and $T1$, while on $T2$ the system returns slightly more wrong classifications than correct ones. This means that the performance is very bad on the 10 additional cases included in $T2$ with respect to $T1$, which significantly affects the overall performance on Tot as well. As regards the smoothed evaluation, we first computed accuracy and error rate with the usual formula, i.e. without considering the cases for which a decision is not taken (NC) neither as correct nor as wrong answers (i.e., $acc + NC + err = 1$), to see which of these metrics was more affected. For each sub-dataset (and hence for the entire dataset as well), the difference between boolean and smoothed evaluation is positive both in error rate ($\Delta_{err} = err_b - err_s$) and in accuracy ($\Delta_{acc} = acc_b - acc_s$). This means that the smoothed evaluation yields a *gain* in error rate, but a *loss* in accuracy, i.e. some undecided cases were associated to correct answers and some others to wrong answers. However, the gain is always much more than the loss, indicating that the cases in which a classification is not made often correspond to wrong decisions. E.g., in $T1$ the gain (i.e., reduction in error rate) is $0.3 - 0.15 = 0.15$, whereas the loss (in accuracy) is just $0.7 - 0.65 = 0.05$. Considering sample test error not only confirms the gain in error rate ($err_b \geq err_s + \Delta$), but also yields a gain in accuracy ($acc_s + \Delta \geq acc_b$) on all subsets.

Since in the above experiment the smoothed evaluation turned out to be more reliable, we used this setting when comparing the performance of our approach to those of the 18 participants to the PAN2013 Challenge on the English dataset. We used the results published in Juola and Stamatatos (2013) because those systems are not publicly available to run additional experiments with other benchmark datasets. The evaluation metrics exploited in the original competition were the following (Juola and Stamatatos 2013):

$$\text{Recall}: \quad R = \frac{\#\text{correct\_answers}}{\#\text{problems}}$$

$$\text{Precision}: \quad P = \frac{\#\text{correct\_answers}}{\#\text{answers}}$$

$$F_1\text{−measure}: \quad F = \frac{2 \cdot P \cdot R}{P + R}$$

that are applicable to techniques that may abstain from returning an answer on some problems. The full ranking and results are reported in Table 6. The performance of our approach on the official test set $T2$ is not encouraging: it shares the last position with another system as regards Precision, and it is below the baseline (even if not the last) as regards F_1-measure; as regards Recall, although above the baseline, it is in the bottom half of the ranking.

However, the situation is completely different considering the early-bird test set $T1$: for Precision it is still in the bottom half of the ranking, but now half way (0.65) between the baseline (0.5) and the winner (0.8); for Recall it is now the absolute winner, which also raises F_1-measure in the upper half of the ranking. This confirms that the hardest part of the test set is in $T2 \setminus T1$, and that $T1$ is not really representative of $T2$, which is quite strange, given that it should be an 'early bird' evaluation subset for $T2$. Thus, we decided to further explore the matter by running additional experiments. Since our system does not need a training phase, we could also check its performance on the training set Tr, which allowed us to still work on the same kind of data. The system even reached a much better Recall than on $T1$, and an F_1-measure very close to that of the winners; Precision is still in the middle of the ranking but better than for $T1$. We consider this a success, and an indication that the system's performance on $T2$ may not be representative of its general performance,

Table 6 Comparison with performances of PAN 2013 Challenge for English Datasets

Submission	F_1	Submission	Precision	Submission	Recall
Seidman	0.80	Seidman	0.80	Tr$_s$	0.88
Veenman	0.80	Veenman	0.80	T1$_s$	0.81
Tr$_s$	0.78	Layton	0.77	Seidman	0.80
Layton	0.77	Moreau	0.77	Veenman	0.80
Moreau	0.77	Ghaeini	0.76	Layton	0.77
Jankowska	0.73	Jankowska	0.73	Moreau	0.77
Vilariño	0.73	Vilariño	0.73	Jankowska	0.73
T1$_s$	0.72	Tr$_s$	0.70	Vilariño	0.73
Halvani	0.70	Halvani	0.70	Halvani	0.70
Feng	0.70	Heng	0.70	Feng	0.70
Ghaeini	0.69	Petmanson	0.67	Petmanson	0.67
Petmanson	0.67	Bobicev	0.66	Ghaeini	0.63
Bobicev	0.64	T1$_s$	0.65	Bobicev	0.63
Sorin	0.63	Sorin	0.63	Sorin	0.63
vanDam	0.60	vanDam	0.60	T2$_s$	0.60
Jayapal	0.60	Jayapal	0.60	vanDam	0.60
Kern	0.53	Kern	0.53	Jayapal	0.60
				Kern	0.53
baseline	0.50	baseline	0.50		
Vartapetiance	0.50	Vartapetiance	0.50	baseline	0.50
T2$_s$	0.48	Ledesma	0.47	Vartapetiance	0.50
Ledesma	0.47	T2$_s$	0.40	Ledesma	0.47
Grozea	0.40	Grozea	0.40	Grozea	0.40

because on different portions of the same dataset it obtains very different results. This is why also the performance on Tr and $T1$ was reported in Table 6, even if the figures for the other systems just refer to $T2$. Indeed, the fact that the results for Tr and $T1$ are close to each other at the top of the ranking may suggest that these two subsets are similar, and further supports the hypothesis that the additional cases in $T2$ are quite different from the others.

Summing up, our approach performs poorly on the PAN2013 competition's official test set, but has significantly better performance on other subsets of data from the same competition. In particular, using the smoothed evaluation setting it outperforms all the other systems for Recall on the official early-bird test set, and reaches even better results on the training set. This is obtained at the cost of not classifying some documents. So, our system is very cautious in its predictions, which makes it suitable for cases where one is not willing to take wrong decisions because they might have critical consequences (as in the legal or forensics domain). In these cases, it perceives the risk and does not return a decision. Another advantage of our approach is that, being unsupervised, it does not leverage a training phase, and thus it is applicable also when, due to the lack of training documents, others are not. This ensures that it may be considered relevant and worth attention, even if it is not the best in the competition. Indeed, since the proposed approach is quite original and different from the mainstream research in the field, the comparison should be considered as a proof of principle, as the results are comparable to systems exploiting different approaches.

6 Conclusions

This work proposed a novel approach to author identification based on relational representations. The syntactic structure of sentences is exploited, motivated by the assumption that it can somehow capture the writing style of an author. Sentences in natural language are translated into relational patterns, from which suitable models are extracted by a clustering technique. Then, a decision about the 'same-authorship' problem is taken by checking the amount of overlapping between the models of the known and unknown author. Experimental results have shown that this technique reaches results that are comparable with the state-of-the-art, while not requiring any training and being effective even for short texts. The technique can be applied to any natural language for which suitable linguistic resources are available. Moreover, it is able to autonomously identify cases in which the classification is less reliable, due to the available data being too poor or to poor outcomes of the clustering step. By abstaining from returning a decision in these cases, the method can significantly improve its performance statistics in the cases in which it takes a decision.

We are currently checking how to apply this approach, in a slightly modified version, to the strictly-related problem of plagiarism detection. Future work will investigate how the performance of the proposed relational approach can be integrated with statistical approaches to improve prediction performance, and how to better define the gray zone (possibly depending on the specific dataset at hand). Also, the use of other similarity functions, applied to a propositional encoding of the relational descriptions, will be studied.

Acknowledgments The author would like to thank Fabio Leuzzi and Fulvio Rotella for their contribution in implementing the proposed approach, and for the useful hints in setting up the strategy. This work was partially funded by the Italian PON 2007-2013 project PON02_00563_3489339 "Puglia@Service".

References

Argamon, S., Saric, M., Stein, S. S. (2003). Style mining of electronic messages for multiple authorship discrimination: first results. In Getoor, L., Senator, T.E., Domingos, P., Faloutsos, C. (Eds.) *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, (pp. 475–480): ACM.

Argamon, S., Whitelaw, C., Chase, P., Hota, S.R., Garg, N., Levitan, S. (2007). Stylistic text classification using functional lexical features: Research articles. *Journal American Society Information Science Technology*, *58*(6), 802–822.

Church, K., & Hanks, P. (1990). Word association norms, mutual information and lexicography. *Computational Linguistics*, *16*, 22–29.

De Marneffe, M., Maccartney, B., Manning, C.D. (2006). Generating typed dependency parses from phrase structure parses. In *Proc. Int'l Conf. on Language Resources and Evaluation (LREC)* (pp. 449–454).

Diederich, J., Kindermann, J., Leopold, E., Paass, G. (2003). Authorship attribution with support vector machines. *Applied Intelligence*, *19*(1-2), 109–123.

Fellbaum, C. (Ed.) (1998). *WordNet: An Electronic Lexical Database*. Cambridge: MIT Press.

Feng, V.W., & Hirst, G. (2013). Authorship verication with entity coherence and other rich linguistic features notebook for pan at clef 2013. In Forner, P., Navigli, R., Tufis, D. (Eds.) *CLEF 2013 Labs and Workshops - Online Working Notes, PROMISE, Padua, Italy*.

Ferilli, S., Basile, T.M., Biba, M., Mauro, N.D., Esposito, F. (2009a). A general similarity framework for horn clause logic. *Fundamenta Informaticæ*, *90*(1-2), 43–46.

Ferilli, S., Biba, M., Di Mauro, N., Basile, T., Esposito, F. (2009b). Plugging taxonomic similarity in first-order logic horn clauses comparison. In *Emergent perspectives in artificial intelligence, lecture notes in artificial intelligence* (pp. 131–140): Springer.

Ferilli, S., Leuzzi, F., Rotella, F. (2011). Cooperating techniques for extracting conceptual taxonomies from text. In *Proceedings of the workshop on mining complex patterns at AI*IA XIIth conference*.

van Halteren, H. (2004). Linguistic profiling for author recognition and verification. In *Proceedings of the 42nd annual meeting on association for computational linguistics, ACL '04*. Stroudsburg: Association for Computational Linguistics.

Juola, P., & Stamatatos, E. (2013). Overview of the author identification task at PAN 2013. In *Information access evaluation. Multilinguality, multimodality, and visualization. 4th international conference of the clef initiative (CLEF 2013)*. http://www.uni-weimar.de/medien/webis/research/events/pan-13/pan13-papers-final/pan13-authorship-verification/juola13-overview.pdf.

Klein, D., & Manning, C.D. (2003). *Fast exact inference with a factored model for natural language parsing. In Advances in neural information processing systems* Vol. 15: MIT Press.

Leuzzi, F., Ferilli, S., Rotella, F. (2013). Improving robustness and flexibility of concept taxonomy learning from text. In Appice, A., Ceci, M., Loglisci, C., Manco, G., Masciari, E., Ras, Z.W. (Eds.) *New frontiers in mining complex patterns - first international workshop, NFMCP 2012, Held in conjunction with ECML/PKDD 2012, Bristol, UK, September 24, 2012, Revised Selected Papers, CCIS* (Vol. 7765, pp. 232–244). Berlin Heidelberg: Springer-Verlag.

Li, J., Zheng, R., Chen, H. (2006). From fingerprint to writeprint. *Commun ACM*, *49*(4), 76–82.

Lloyd, J.W. (1987). *Foundations of logic programming*, 2nd Edn. Springer.

Lowe, D., & Matthews, R. (1995). Shakespeare vs. fletcher: a stylometric analysis by radial basis functions. *Computers and the Humanities*, *29*(6), 449–461.

Mccarthy, P.M., Lewis, G.A., Dufty, D.F., Mcnamara, D.S. (2006). Analyzing writing styles with coh-metrix. In Sutcliffe, G., & Goebel, R. (Eds.) *Proceedings of the Florida artificial intelligence research society international conference (FLAIRS)*, (pp. 764–769): AAAI Press.

Qiu, L., Kan, M.Y., Chua, T.S. (2004). A public reference implementation of the RAP anaphora resolution algorithm. In *Proceedings of the 4th international conference on language resources and evaluation, LREC 2004, May 26-28*, (pp. 291–294). Lisbon, Portugal: European Language Resources Association.

Raghavan, S., Kovashka, A., Mooney, R. (2010). *Authorship attribution using probabilistic context-free grammars. In Proceedings of the ACL 2010 conference short papers, ACLShort '10*, (pp. 38–42). Stroudsburg: Association for Computational Linguistics.

Seidman, S. (2013). Authorship verification using the impostors method – notebook for PAN at CLEF 2013. In Forner, P., Navigli, R., Tufis, D. (Eds.) *CLEF 2013 labs and workshops - online working notes, PROMISE*. Padua, Italy.

Settles, B. (2010). *Active learning literature survey*. Tech. Rep., Computer Sciences 1648, University of Wisconsin-Madison.

Tweedie, F.J., Singh, S., Holmes, D.I. (1996). Neural network applications in stylometry: the federalist papers. *Computers and the Humanities*, *30*(1), 1–10.

Vilariño, D., Pinto, D., Gómez, H., León, S., Castillo, E. (2013). Lexical-syntactic and graph-based features for authorship verification, - notebook for pan at clef 2013. In Forner, P., Navigli, R., Tufis, D. (Eds.) *CLEF 2013 labs and workshops - online working notes, PROMISE*. Padua, Italy.

Zheng, R., Li, J., Chen, H., Huang, Z. (2006). A framework for authorship identification of online messages: writing-style features and classification techniques. *Journal American Society Information Science Technology*, *57*(3), 378–393.