Watermarking for Ontologies

Fabian M. Suchanek[1], David Gross-Amblard[1,2], and Serge Abiteboul[1]

[1] INRIA Saclay, Paris
[2] LE2I CNRS, Université de Bourgogne, Dijon

Abstract. In this paper, we study watermarking methods to prove the ownership of an ontology. Different from existing approaches, we propose to watermark not by altering existing statements, but by removing them. Thereby, our approach does not introduce false statements into the ontology. We show how ownership of ontologies can be established with provably tight probability bounds, even if only parts of the ontology are being re-used. We finally demonstrate the viability of our approach on real-world ontologies.

1 Introduction

An ontology is a formal collection of world knowledge. Creating an ontology usually involves a major human effort. In the case of manually constructed ontologies, human effort is needed to collect the knowledge, to formalize it and to maintain it. The same applies to ontologies constructed by a community, such as Freebase or DBpedia. In the case of automatically constructed ontologies, human effort comes in the form of scientific investigation and the development of algorithms. Consequently, the creators of an ontology usually do not give away the ontology for free for arbitrary use. Rather, they request their users to pay for the content, to follow the terms of a specific license, or to give credit to the creators of the ontology. In most cases, it is prohibited to re-publish the data, or allowed only with proper acknowledgment.

This restriction is most obvious in the case of commercially sold ontologies such as [9]: The use of the data is restricted by the sale contract. The contract usually prohibits the re-publication of the data. Any dissemination of the data into other data sets constitutes a breach of contract.

One might think that the picture would be different for the public ontologies of the Semantic Web. The general spirit of the Semantic Web wants data to be shared across application and community boundaries[3]. However, even the ontologies of the Semantic Web are not available for arbitrary re-publication. Table 1 shows some popular ontologies mentioned together with their licenses. None of the ontologies is available in the public domain. All of them require at least an acknowledgment when their data is re-published. It is considered dishonest to sell or re-publish the data from an ontology elsewhere without giving due credit to the original.

[3] `http://www.w3.org/2001/sw/`

Some data sets are not freely available at all (see Table 1). The Wolfram Alpha data set[4], for example, can be queried through an API, but cannot be downloaded. Its terms of use prohibit the systematic harvesting of the API to re-create the data set. Any re-publication of a substantial portion of such data constitutes a breach of the terms of use. Similar observations apply to trueknowledge[5] or the commercial version of Cyc [9]. In all of these cases, the extraction and systematic dissemination of the data is prohibited.

License	Conditions	Ontologies
GFDL	attribution, copyleft	DBpedia [2]
GPL	attribution, copyleft	SUMO [10]
CC-BY	attribution	YAGO [15], Freebase, Geonames, OpenCyc [9]
CC-BY-ND	attribution, no derivatives	UniProt
–	access under restrictions	TrueKnowledge, KnowItAll, WolframAlpha, full Cyc [9]

Table 1. Common licenses for ontologies

This raises the issue of how we can detect whether an ontology has been illegally re-published. We call a person who re-publishes an ontology (or part of it) in a way that is inconsistent with its license an *attacker*. The attacker could, e.g., re-publish the ontology under his own name or use parts of the ontology in his own ontology without giving due credit. We call the source ontology the *original ontology* and the re-published ontology the *suspect ontology*. We want to solve the problem of ownership proof: How can we prove that the suspect ontology contains part of the original ontology? Obviously, it is not sufficient to state that the suspect ontology contains data from the original ontology. This is because ontologies contain world knowledge, that anybody can collect. Take the example of an ontology about scientists: the attacker could claim that he also collected biographies of scientists and that he happened to produce the same data set as the original ontology. A similar argument applies to ontologies that have been derived from public sources. YAGO [15] and DBpedia [2], e.g., have both been extracted from Wikipedia. An attacker on DBpedia could claim that he also extracted data from Wikipedia and happened to produce a similar output.

One might be tempted to assume that we could simply publish the original ontology with a time stamp (in the style of proof of software ownership). For example, we could upload the ontology to a trusted external server. If someone else publishes the same data later, then we could point to the original copy. However, this does not prove our ownership. The other publisher could have had the data before we published ours. The fact that he did not publish his data cannot be used against him.

Therefore, a more sophisticated approach is needed to enable ownership proofs. This is the goal of the present paper. We present an approach that uses digital watermarking to detect whether a suspect ontology contains data derived from an original ontology. Watermarking techniques aim at hiding some relevant information in a data set, in an invisible or robust way. Finding such information in a suspect data set acts as the proof of ownership. Several works

[4] `http://www.wolframalpha.com/`
[5] `http://www.trueknowledge.com/`

consider watermarking for relational databases by performing voluntarily alteration of data. These approaches could be extended to ontologies. However, data alteration invariably decreases the precision of the ontology.

Therefore, we develop an alternative method that is specifically adapted to the Semantic Web: We propose to watermark an ontology by removing carefully selected statements before publishing. The suspect absence of these statements in an ontology with a significant overlap will act as the proof of theft. We argue that this does less harm than altering statements, because the Semantic Web operates under the Open World Assumption: Most ontologies are incomplete.

More specifically, our contributions are as follows:

1. A formalization of the problem of ontological data re-publication,
2. An algorithm for watermarking ontologies, which allows detecting malicious re-publication without harming the precision,
3. Extensive experiments that show the validity of our approach.

The rest of this paper is structured as follows: Section 2 summarizes related work. Section 3 formalizes our scenario and lists different attack models. Section 4 presents our watermarking algorithm with a formal analysis. Section 5 details our experiments before Section 6 concludes.

2 Related Work

In [4], the authors introduce named graphs as a way to manage trust and provenance on the Semantic Web. Named graphs, however, cannot guard against the misuse of ontologies that are publicly available.

One of the oldest attempts to prove ownership of factual data is the use of fictitious entries in dictionaries. Since the 19th century, dictionaries, maps, encyclopedias and directories have had occasional fake entries. The New Columbia Encyclopedia, for example, contained an entry about a fictitious photographer called Lillian Virginia Mountweazel. If such an entry ever appeared in another encyclopedia, it was clear that the data was copied. To mark an ontology, however, it is not sufficient to add a single isolated entity, because an attacker can simply remove unconnected entities.

A classical way to achieve ownership proofs is to apply watermarking techniques. Some recent proposals have targeted ontologies [6]. This previous effort uses a purely syntactical rewriting of the RDF XML source layout into an equivalent layout to hide information. This approach can be circumvented by normalizing the XML file. This can be done, e.g., by loading the file into a tool such as Protégé [11] and then saving it again as an XML document.

Quite a number of approaches have targeted semi-structured data [13] and relational databases [1,14,7,8,12]. These works provide one way to prove ownership of ontologies. Most of them are blind, i.e., they do not require the original data set for detection. However, all of these approaches presume that the schema of the data is known. This is not necessarily the case on the Semantic Web. Some ontologies (such as DBpedia or YAGO) have hundreds of relationships. An attacker just has to map some of them manually to other relationships to obscure the theft. We will develop approaches that can still identify the suspect data, but in a non-blind way. Furthermore, the classical methods work by voluntarily altering data. Therefore, we call these methods *modification approaches* in the sequel. Such approaches could, e.g., change the birth year of Elvis Presley from the year 1935 to the year 1936. While the introduced errors are only gradual for numerical data, they are substantial for categorical data. Such an approach could, e.g., change Elvis' nationality from American to Russian. Apart from the fact that an ontology owner will find it controversial to voluntarily alter clean data, such an approach will also decrease the precision of the ontology with respect to the real world. Furthermore, the altered facts can lead to contradictions with other data sets. This is not only annoying to the user, but can also allow the attacker to detect and remove the altered facts. Therefore, we present an alternative approach in this paper, which works by deleting carefully selected facts. Since the Semantic Web operates under the open world assumption, the absence of true information is less harmful than the presence of false information.

3 Model

3.1 Watermarking

A watermarking protocol is a pair of algorithms $(\mathcal{M}, \mathcal{D})$, where $\mathcal{M}$ stands for the marker and $\mathcal{D}$ the detector (Figure 1). Given an original ontology O and a secret key $\mathcal{K}$, the marker algorithm outputs a watermarked ontology $O^* = \mathcal{M}(O, \mathcal{K})$. Given a suspect ontology O', the original ontology and the secret key, the detector decides if O' contains a mark, that is if $\mathcal{D}(O', O, \mathcal{K})$ is `true`.

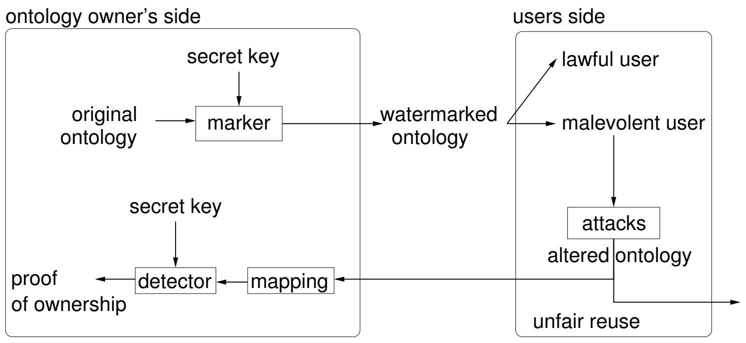

Fig. 1. Watermarking protocol for ontologies

If O' contains the mark, then it is assumed that O' has indeed been derived from O^*. However, the watermarking protocol may erroneously say that O' has been derived from O^*, even though O' just contains the mark by chance. For example, O' may be a totally unrelated ontology. In this case, O' is called a *false positive*. Watermarking protocols are designed so that the probability of a false positive is provably below a confidence threshold ξ. The parameter ξ is called the *security parameter* of the protocol and is typically in the order of $\xi = 10^{-6}$.

In an adversarial setting, the attacker can try to evade the detection of the mark in the ontology by various means, including:

- **Subset attack:** The attacker uses only a subset of the stolen ontology. For example, the attacker could choose a certain thematic domain from the original ontology (such as, say, sports), and re-use only this portion. While publishing just a few statements should be allowed, larger stolen parts should be detected by the protocol.
- **Union attack:** The attacker merges the ontology with another one. For example, the attacker could combine two ontologies about gene expressions, thereby hiding the stolen ontology in a larger data set. A naive union attack would keep facts from both ontologies, even if they are inconsistent with each other.
- **Intersection attack:** The attacker keeps only ontology elements that are found in another ontology. For example, an attacker could remove entities from the stolen ontology that do not appear in a reference ontology.
- **Comparison attack:** The attacker compares the ontology to another data source and eliminates inconsistent information. For example, the attacker could cross-check birth dates of one ontology with the birth dates in a reference ontology and remove inconsistent dates.
- **Random alteration attack:** The attacker alters values, relations or entities. This action is limited as the attacker still desires valuable data.

The ability of the watermarking protocol to withstand these attacks is called its *robustness*. Watermarking protocols are typically designed so that the probability of a successful attack is provably below their security parameter ξ.

An attacker may also publish someone else's ontology as part of an offensive or illegal data set. An attacker may, e.g., take an ontology about chemistry and publish it as part of a data set about weapons of mass destruction. If the ontology was marked, it will appear as if the creator of the original ontology authored the offensive data set. Our method can fend off this so-called *copy attack*.

3.2 Ontologies

An RDFS ontology over a set of entities (resources and literals) E and a set of relation names R can be seen as a set of triples $O \subset E \times R \times E$. Each triple is called a *statement*, with its components being called *subject*, *predicate* and *object*. We will write $e.name$ to refer to the identifier of the entity in the ontology. This can be the URI (in case of a resource) or the string value (in case of a literal). RDFS specifies certain *semantic rules*. These rules state that if the ontology contains certain statements, certain new statements should be added to the ontology. We call these added statements *redundant* and assume that redundant statements have been removed from the ontology [5]. RDFS rules are strictly positive. Therefore, an RDFS ontology cannot become inconsistent if a statement is removed or altered. This predestines RDFS ontologies for watermarking and we leave more sophisticated ontology models for future work.

To prove that a suspect ontology O' copied data from an original ontology O, we will first have to determine whether O' and O contain similar data. This is a challenging task in itself, because it amounts to finding a mapping from O' to O. This problem has attracted a lot of research in the context of the Web of Linked Data [3]. Finding such a mapping is outside the scope of the present paper. Here, we limit ourselves to 2 assumptions: (1) We know a partial mapping function σ, which maps (some of) the entities of O' to entities of O; (2) there exists a partial mapping function σ_R, which maps the relations of O' to relations of O. Note that we do not need to know σ_R. Our method has been designed so that it is transparent to the "schema" of the ontologies. We need the existence of σ_R just to assure that O and O' have some structural similarity. An investigator needs to find only σ. If the ontology was truly stolen, then the entities will probably be recognizable in some way, because otherwise the stolen ontology would be less useful. In the best case, σ will reflect mainly syntactic variations of the identifiers and values. We note that the modification approaches like [1] also require σ, because they determine the tuples to mark based on the value of the primary key, which has to be available at detection. In addition, the modification approaches also require the mapping of the schema, σ_R, which our approach does not require. In the sequel, we assume that σ has been found and has been applied to O'. We are now concerned with the question that follows: We want to prove that O' and O are similar not just by chance, but because O' copied data from O.

3.3 Ethical considerations

Ontologies represent world knowledge. Therefore, it is questionable whether one can "own" such data. Can the creator have a copyright on the ontology, given that it is nothing more than a collection of facts about the world? In this paper, we do not deal with the legal implications of owning or copying ontologies. We only provide a method to prove that one data source copied from another source – independently of whether such behavior is considered legal or not.

Our approach will remove facts from the ontology before publishing. Then the question arises whether it is honest to withhold information that one could publish. However, an ontology always represents only part of reality. In most cases, the creator of an ontology is not obliged to make it exhaustive. The Open World Assumption of the Semantic Web makes the absence of information a normal and tolerable circumstance.

In general, the watermarking of an ontology always remains a trade-off between the ability to prove ownership and the truthfulness of the data. One should not willfully alter ontologies that are highly sensitive to even small misrepresentations of reality, such as ontologies in the domain of medicine, security, or aeronautics. However, unlike the modification approaches [1], our approach of deleting facts can be applied even if truthfulness of the data has to be preserved, as long as some incompleteness can be tolerated.

4 Watermarking Ontologies

4.1 Watermarking Basics

Our starting point is a watermarking protocol from Agrawal et al. [1] for relational databases. Their statistical watermarking uses cryptographically secure pseudo-random number generators (CSPRNGs). A CSPRNG is a function which, given an integer seed value, produces a sequence of pseudo-random bits. The bits are pseudo-random in the sense that, without the seed, and given the first k bits of a random sequence, there is no polynomial-time algorithm that can predict the $(k+1)$th bit with probability of success better than 50% [16]. A CSPRNG is a *one-way-function*. This means that, given a CSPRNG and a sequence of random bits it produced, it is close to impossible to determine the seed value that generated it. More formally, given a CSPNG f, for every randomized polynomial time algorithm A, $Pr[f(A(f(x))) = f(x)] < \frac{1}{p(n)}$ for every positive polynomial $p(n)$ and sufficiently large n, assuming that x is chosen from a uniform distribution [17]. In the following, we use a given CSPRNG $\mathcal{G}$.

After $\mathcal{G}$ has been seeded with a value by calling $\mathcal{G}.seed(value)$, one can repeatedly call the function $\mathcal{G}.nextBit()$ to obtain the next bit in the random sequence. By combining multiple calls to this function, we can construct a pseudo-random integer value. The function that delivers a pseudo-random integer number greater or equal to 0 and below a given upper bound k is denoted by $\mathcal{G}.nextInt(k)$.

Our watermarking algorithm makes use of a *secret key*. A secret key is a integer number that is only known to the owner of the original ontology. We will use the key as a seed value for $\mathcal{G}$. We also make use of a *cryptographically secure hash function*. A hash function is a function which, given an object, returns an integer number in a certain range. A cryptographically secure hash function is such that it is infeasible to find two inputs with the same output, or the inverse of the output. We assume a given hash function, such as SHA, denoted *hash*. In order to resist the aforementioned copy attack, we will first compute the secure hash of the original ontology. All our subsequent computations will depend on this hash, so that no attacker can pretend to own the watermarked version by finding another ontology with the same watermarking.

4.1.1 Algorithm. Our watermarking method shall be transparent to relation names, because we do not want to require an investigator to find a mapping of the schema. Therefore, we define the notion of a *fact pair*. A fact pair of an ontology O is a pair of entities $\langle e_1, e_2 \rangle$, such that there exists r such that $\langle e_1, r, e_2 \rangle \in O$. Algorithm 1 marks an ontology by removing fact pairs. It takes as input the original ontology O, a secret key $\mathcal{K}$, and the number of facts $delTotal$ to remove. The secret key is an arbitrary chosen number. Section 4.2.1 will discuss how to find a suitable value for $delTotal$. For each fact pair of the ontology, the algorithm seeds $\mathcal{G}$ with the names of the entities, the hash of O, and $\mathcal{K}$. If the next random integer of $\mathcal{G}$ between 0 and the total number of fact pairs in O divided by $delTotal$ happens to be 0, the fact pair is removed. This removes $delTotal$ facts from the ontology[6]. Note that our algorithm does not consider the relation names at all. After running Algorithm 1, the marked ontology O^* is published. The original ontology O is kept secret.

To detect whether a suspect ontology stole data from an original ontology, we run Algorithm 2 on the original ontology O (without the mark) and the suspect

[6] If less than $delTotal$ facts got removed, we rerun the algorithm with a different key.

Algorithm 1 subtractiveMark(orig. ontology O, secret key $\mathcal{K}$, integer $delTotal$)

```
$O^* \leftarrow O$
**for all** $\langle e_1, e_2 \rangle \in \pi_{subject,object}(O)$ **do**
    $\mathcal{G}.seed(e_1.name \oplus e_2.name \oplus hash(O) \oplus \mathcal{K})$
    **if** $\mathcal{G}.nextInt(|\pi_{subject,object}(O)|/delTotal) = 0$ **then**
        Remove $\langle e_1, *, e_2 \rangle$ from $O^*$
    **end if**
**end for**
**return** $O^*$
```

ontology O' (after having applied the mapping from Section 3.2). The algorithm runs through all fact pairs of O and computes the proportion of published fact pairs that appear in the suspect ontology. It also computes the proportion of removed fact pairs that appear in the suspect ontology. Since we only consider fact pairs and not facts, a mapping of the relation names (the schema) is not necessary. It is possible that the suspect ontology contains some of the fact pairs that we removed from the original before publishing. This can be for two reasons: Either the suspect ontology is innocent and just happens to have a thematic overlap with our original, or the suspect ontology imported data from other sources, thus complementing the facts we removed. The algorithm then compares the ratio of removed fact pairs that appear in O' to the ratio of published fact pairs that appear in O. If O' is innocent, these ratios should be the same. If the ratio of deleted facts is lower than the ratio of published facts, and significantly so, this indicates a theft and the algorithm will return *true*. It seems highly counter-intuitive that the absence of a fact should prove theft of data. Yet, the proof comes from the fact that the removed statements form a pattern of present and absent facts in O. The probability that this pattern appears by chance in another ontology is extremely low.

Algorithm 2 subtractiveDetect(original O, suspect O', key $\mathcal{K}$, integer $delTotal$)

```
$O^* \leftarrow subtractiveMark(O, \mathcal{K}, delTotal)$
$pubFound \leftarrow 0; delFound \leftarrow 0;$
**for all** $\langle e_1, e_2 \rangle \in \pi_{subject,object}(O)$ **do**
    **if** $\langle e_1, e_2 \rangle \in \pi_{subject,object}(O')$ **then**
        **if** $\langle e_1, e_2 \rangle \in \pi_{subject,object}(O^*)$ **then** $pubFound++$
        **else** $delFound++$
    **end if**
**end for**
$pubTotal \leftarrow |\pi_{subject,object}(O^*)|$
**if** $delFound/delTotal \geq pubFound/pubTotal$ **then return false**
**return** $delFound/delTotal$ significantly different from $pubFound/pubTotal$
```

Let us detail the check of significance. The suspect ontology will cover a certain portion of facts of the original ontology. The central observation is that, if this overlap is by chance, then the suspect ontology should cover the same portion of published facts as it covers of the deleted facts, because the watermarking is randomized. Thus, we have to determine whether any difference between these two ratios is statistically significant. This is the last step in Algorithm 2. This significance is determined by a χ^2 test. Be $pubTotal$ the total number of published fact pairs. Be $pubFound$ the number of published fact pairs that appear in the suspect ontology. Be $delTotal$ the total number of deleted fact pairs. Be $delFound$ the number of deleted fact pairs that appear in the suspect ontology and be $N = pubTotal + delTotal$ the total number of fact pairs. We get

$$\chi^2 = \frac{N(delFound \times (pubTotal - pubFound) - pubFound \times (delTotal - delFound))^2}{(pubFound + delFound) \times (N - pubFound - delFound) \times delTotal \times pubTotal}.$$

If $\chi^2 > \chi^2(1, \xi)$, where ξ is the security parameter, then the two ratios are not independent. Since the removal of the fact pairs was purely random, any significant difference between the ratios indicates a dependence on the original ontology. In the extreme case, the suspect ontology reproduces the published facts and omits all (or nearly all) removed facts.

In order to be applicable, the standard χ^2 test requires the total number of samples to be greater than 100 and the expected number of samples for each case to be greater than 5. Therefore, our algorithm returns *true* iff $N > 100$ and $(pubFound + delFound) \times delTotal/N > 5$ and $\chi^2 > \chi^2(1, \xi)$.

4.2 Analysis

4.2.1 Impact. We are interested in how many fact pairs we have to remove in order to achieve significance in the χ^2 test. This number depends on the total number of fact pairs N. It also depends on the types of attacks against which we want to protect. The first property of an attack is the overlap ratio of found facts, $\omega = (pubFound + delFound)/N$. We choose $\omega = 1$ if we want to protect only against a theft of the complete ontology. We choose a smaller value if we want to protect also against a theft of a sub-portion of the ontology. The second property of an attack is the ratio of removed facts $\delta = delFound/delTotal$ that appear in the stolen ontology. If the attacker just republishes our published ontology, $\delta = 0$. If he adds data from other sources, this can complement some of the facts we removed. Ratio δ should be the proportion of removed facts that we expect in the stolen ontology. If δ is larger, and ω is smaller, the protection is safer, but the marking will remove more facts. Abbreviating $delTotal = d$, this yields $pubFound = \omega N - \delta d, delFound = \delta d, pubTotal = N - d$ and thus

$$\chi^2 = \frac{N((\omega N - \delta d)(1 - \delta)d - \delta d((1 - \omega)N - (1 - \delta)d))^2}{\omega N(1 - \omega)Nd(N - d)}.$$

For $\chi^2 > \chi^2(1, \xi)$, this yields

$$d > \frac{N\omega(1 - \omega)\chi^2(1, \xi)}{N(\delta(1 - \omega) - \omega(1 - \delta))^2 + \omega(1 - \omega)\chi^2(1, \xi)}.$$

We have to impose $N > 100$ as a precondition for the χ^2 test. We also have to impose $d > 5/(\omega(1 - \omega))$, i.e., $d > 20$ in the worst case. Finally, $\delta < \omega$, because we cannot prove theft if the ratio of appearing deleted facts is greater than the ratio of appearing published facts. As an example, take a choice of $\xi = 10^{-6}$, which leads to $\chi^2(1, \xi) = 23.9284$. Assuming an overlap ratio of $\omega = \frac{1}{2}$, a fault tolerance of $\delta = 0.2$, and $N = 30 \times 10^6$ fact pairs, we get $d = 67$, i.e., 67 fact pairs have to be deleted from the ontology.

4.2.2 Robustness. In general, the χ^2 test tends to err on the safe side, concluding independence only in the presence of overwhelming evidence. Thus, our algorithm will signal theft only in very clear cases ("in dubio pro reo"). However, our algorithm is also well-protected against attacks. First, a marked ontology is protected against intersection attacks. Intersection attacks can happen, e.g., when an attacker wants to misuse someone else's ontology to clean up his own noisy data set. Since an intersection does not add facts, our marks survive such an attack. The marked ontology is also protected against comparison attacks, because the ontologies we target generally suffer from incompleteness. Thus, a fact that is absent in the original ontology but present in a reference ontology will not raise suspicions. A marked ontology is also safe against subset attacks, if ω is chosen smaller than the proportion of stolen facts. A union attack, in contrast, could add information that fills up some of the removed facts. In this case, the marks will be reduced to those portions of the ontology that do not appear in the other ontology. By choosing $1 - \delta$ equal to the portion that we estimate to be proper to the ontology, and adjusting d accordingly, we can still guard against the union attack. Random alteration attacks fall into the scope of the classical analysis of robustness [1]: an attacker being ignorant on the positions of the deleted facts can only try at random to delete more facts or fill missing ones. For this attack to be successful, a large number of facts have to be altered, a much larger number than the watermark algorithm used. Finally, an attacker can try to modify the fact pairs. Deleted facts are of course not sensitive to this attack, but the number $pubFound$ can be altered. However, as the subset of published facts we are looking for at detection are chosen pseudo-randomly, the attacker has no way to locate them efficiently. The only valid strategy for the attacker is again to alter a huge amount of fact pairs, which reduces drastically the quality of the stolen ontology.

4.2.3 Comparison to Modification Watermarking. The modification approach [1] changes a fact from the ontology. In the majority of cases, it will change a correct fact to an incorrect fact. Thereby, precision invariably suffers. In contrast, our approach does not decrease the precision of the ontology at all. To see this, assume that O contains n statements, c of which are correct. If a correct fact is deleted, which will happen in $\frac{c}{n}$ of the cases, the precision drops to $\frac{c-1}{n-1}$. If an incorrect fact is deleted, the precision increases to $\frac{c}{n-1}$. Thus, on average, the precision is $\frac{c}{n} \times \frac{c-1}{n-1} + \frac{n-c}{n} \times \frac{c}{n-1}$, which is $\frac{c}{n}$, just as before. As a comparison, the modification approaches have an average impact of $\frac{c}{n} \times \frac{c-1}{\mathbf{n}} + \frac{n-c}{n} \times \frac{c}{\mathbf{n}}$ (a modified correct fact turns incorrect, and a modified incorrect fact will still be incorrect, while the number of total facts is the same).

Now let us consider the number of facts that have to be modified in the classical approach. Assuming that $\delta = 0, \xi = 10^{-6}$ and $N = 30 \times 10^6$, we used the estimates in [1] to compute the number of tuples (fact pairs, in our setting) that are required to be modified in order to resist a subset attack of parameter ω. This leads to the numbers in Table 2.

ω	50%	10%	5%	2.5%	0.5%	0.05%
Removing	24	215	456	935	4775	47900
Modifying	96	480	950	1900	9500	95000

Table 2. Number of facts that have to be marked
$\alpha = 0, \xi = 10^{-6}$ and $N = 30 \times 10^6$

The modification method hides a list of 0 and 1 on secretly chosen positions. If such a position is not selected by the subset attack, the marked bit is lost, whatever its value. But for our method, the subset attack has no impact on already deleted facts. Therefore, the modification approach has to modify more fact pairs than we have to delete. Overall, the number of facts that have to be modified is comparable to the number of facts that have to be deleted. Given that removal maintains precision, while modification does not, and that modification yields false facts, the ontology owner might decide to remove instead of to modify.

5 Experiments

5.1 Applicability

5.1.1 Impact. We were interested in how applicable our method is to real world ontologies. For this purpose, we collected 5 ontologies from the Semantic Web that cover a wide range of complexity, size, and topics (Table 3): The core part of YAGO [15], the manually supervised part of DBpedia [2], the Universal Protein Resource[7], an ontology about city finances (provided by the UK government[8]), and a subset of the IMDb[9]. For each ontology, we report the number of facts, fact pairs, relations and instances. We also report the number of entities that are connected to an object by more than one relation. This number is usually small, except for YAGO, where every entity has a label and a (mostly equivalent) preferred name. We compute the number of facts that have to be removed to protect against various subset attacks ($\xi = 10^{-6}$). As a comparison, the last column gives the number of alterations needed for the modification approach [1]. It is roughly independent of the size of the data set. Values for the modification method are not given for $\delta = 10\%$, because the scenario where the attacker irons out the marks has not been considered in [1].

5.1.2 Removing the Marks. We were interested in how an attacker could try to identify the missing facts in order to reinstate them. The attacker could, e.g., compare all instances of a class and see whether some of them lack a relation that all the other instances have. This entity is suspect from an attackers point of view, because he has to assume that we deleted that relation. More precisely,

[7] `http://www.uniprot.org/`
[8] `http://data.gov.uk/`
[9] `http://imdb.com/`

	YAGO	DBpedia	UniProt	Finance	IMDb	*Modification*
# relations	83	1,107	4	11	12	
# instances	2,637,878	1,675,741	1,327	1,948	4,657,880	
# facts	18,206,276	19,788,726	6,096	14,926	34,699,697	
dup. objects	3,529,697	450,171	0	0	14,907	
# fact pairs	14,676,579	19,338,555	6,096	14,926	34,685,090	
Facts to remove						
$\delta = 0, \omega = 50\%$	24	24	24	24	24	[97]
$\delta = 0, \omega = 5\%$	456	456	424	442	456	[975]
$\delta = 0, \omega = 0.5\%$	4,775	4,774	2,677	3,618	4,775	[9700]
$\delta = 0, \omega = 0.05\%$	47,820	47,825	-	-	47,909	[97500]
$\delta = 10\%, \omega = 50\%$	37	37	37	37	37	N/A

Table 3. Marking different ontologies

we call an entity e *suspect* in an ontology O, if there exists a class c, an entity e', and a relation r, such that

$$e \in c, e' \in c, |\{e'' : \langle e', r, e'' \rangle\}| > |\{e'' : \langle e, r, e'' \rangle\}|.$$

We call a fact *discreet* if we can remove it without creating a suspect entity. We computed the proportion of discreet facts, their relations and the proportion of instances with at least one discreet fact (Table 4). Roughly half of the facts are discreet, meaning that we can delete them without raising suspicions. Even if the attacker correctly identifies the fact we removed, he cannot simply discard the entity, because this would still keep the mark. Instead, he has to find the correct value for the missing link to plug in the hole. This may be hard or even close to impossible, because the attacker has no access to the original ontology and cannot run the detection algorithm. Also, he does not know the ratio of discreet facts. Furthermore, from the attacker's point of view, nearly all instances are suspect on the original data set already. Thus, nearly every instance could potentially have been marked. Filling up what could be a hole would amount to reconstructing the ontology. The only exception to this rule is the finance data set, which contains rather complete information (most entities have the maximal number of links in their class). We note, however, that removing facts might still be preferable to modifying facts in this ontology.

	YAGO	DBpedia	UniProt	Finance	IMDb
discreet instances	99%	92%	74%	69%	97%
discreet facts	74%	86%	48%	39%	75%
discreet relations	96%	99%	50%	45%	92%
suspect instances	99%	99%	100%	75%	100%

Table 4. The ontologies from an attackers' point of view

5.1.3 False Positive Detection. A risk with watermarking approaches is the occurrence of false positives. The probability of considering a non-marked ontology suspect has to be bounded. While this poses a problem for blind watermarking methods that do not rely on the original for detection, it is very unlikely that a random ontology is signaled as stolen by our approach, because our approach first checks that the overlap of the suspect ontology with the original ontology is significant. Even in the unlikely situation of a large overlap, the innocent suspect ontology will match with published facts and deleted facts uniformly. Thus the ratio of deleted and found facts will be similar, leading to a correct non-detection. This risk is taken into account and formalized in the significance level of the χ^2 test. We provide next some subset attacks whose overlap is so small that our method does not signal theft.

5.2 Robustness

5.2.1 Attack Simulations. To demonstrate the robustness of our watermarking, we simulated suspect ontologies that overlap only partially with the original ontology. This partial overlap could be due to an incomplete mapping σ or due to the fact that the attacker chose to steal only a subset of the original. We watermarked the Finance ontology with 30 removed facts. This should make the mark resistant to a partial overlap of $\omega = 50\%$ or more. We varied ω and created 10 random overlaps for each value of ω. Figure 2 shows the average rate of successful detection ($\xi = 10^{-6}, \delta = 0$). As predicted, any suspect ontology that overlaps more than half is identified as stolen.

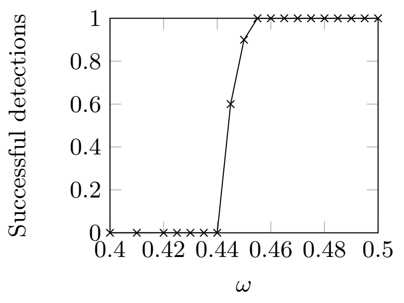

Fig. 2. Rate of successful detection with varying ω, ($\delta = 0$)

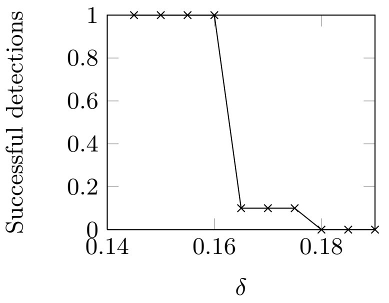

Fig. 3. Rate of successful detection with varying δ ($\omega = 0.5$)

We also simulate suspect ontologies that do contain portions of the deleted facts. This can be the case if the attacker merged the stolen ontology with another data set. We removed 50 facts from the Finance ontology. At an overlap of $\omega = 50\%$, this should protect the ontology up to $\delta = 15\%$. We varied δ and simulated 10 random suspect ontologies for each value of δ. Figure 3 shows that the rate of successful detection. As expected, the rate is 1 for $\delta \leq 15\%$.

5.2.2 Thematic Subset Attacks. Next, we analyze *thematic subset attacks*, i.e., attacks that steal a certain class with all its instances and all their facts. We call such a set facts a *theme*. Table 5 shows populous classes in YAGO together with their number of fact pairs. We computed the ratio ω and the number of fact pairs that would have to be deleted in total (at $\xi = 10^{-6}$). The numbers in brackets show the number of fact pairs that the modification method would consume. Table 6 shows the same characteristics for populous classes in DBpedia. The number of facts to delete fades in comparison to YAGO's 15m fact pairs in total and DBpedia's 19m fact pairs in total. We experimentally verified the subset attacks on DBpedia with 1000 removed facts, achieving significance levels of $\chi^2 = 68, 18, 38, 51, 38$, respectively. This means that, with 1000 removed facts, we could detect all thematic subset attacks except for the one on Television Episodes, because the subset is too small for the chosen number of marks. This experiment confirms our predictions.

Theme	# fact pairs	ω	Fact pairs to remove	
			$\delta = 0$	$\delta = 1\%$
Person	10,594,790	72.19%	20 [67]	20
Village	582,984	3.97%	580 [1,225]	1,037
Album	588,338	4.01 %	574 [1,215]	1,020
Football player	1,200,459	8.18 %	269 [596]	350
Company	400,769	2.73%	854 [1,785]	2,129

Table 5. Protecting different themes in YAGO

Theme	# fact pairs	ω	Fact pairs to remove	
			$\delta = 0$	$\delta = 1\%$
Album	120,7561	6.24%	360 [782]	511
TelevisionEpisode	342,277	1.77%	1,331 [2750]	7,035
Village	701,084	3.63%	637 [1345]	1,213
SoccerPlayer	924,299	4.78%	478 [1020]	764
Film	694,731	3.59%	644 [1360]	1,238

Table 6. Protecting different themes in DBpedia

5.2.3 Union Attacks. We wanted to evaluate how our method works if an attacker merges the stolen ontology with another, possibly similar ontology. This might fill up some of the removed facts and thus destroy our marks. We simulated attacks that steal a certain theme from DBpedia and merge it into YAGO. We merged by matching resources (DBpedia and YAGO use the same local identifiers with different prefixes), matching identical strings and identical numbers, and matching numbers that share the value (ignoring the unit). This yields an overlap of 1.6×10^6 fact pairs between the two ontologies.[10] This overlap is 8% of DBpedia. This means that 8% of the marks that we add to DBpedia can be filled up by merging with YAGO. Hence, we have to choose $\delta > 8\%$ in order to protect against a theft.

[10] Part of the reason for the small overlap is the rather crude mapping (YAGO normalizes numbers to SI units, while DBpedia does not). However, manual inspection also shows that YAGO knows many types for the entities, many labels, and some facts that DBpedia does not know. DBpedia, in turn, captures many infobox attributes that YAGO does not capture. The ontologies share just 1.4 million instances.

Table 7 shows the ontologies obtained from merging YAGO with a certain theme from DBpedia. The table shows the total number of fact pairs of these ontologies as well as the absolute and relative overlap with the original DBpedia. The relative overlap corresponds to ω. The last column shows the number of fact pairs that have to be removed from DBpedia to protect the theme, calculated from ω and $\delta = 8\%$.

YAGO + DBpedia Theme	# fact pairs	overlap with DBpedia	overlap as % of DBpedia (=ω)	Fact pairs to remove
Album	15,920,534	2,831,716	15%	624
TelevisionEpisode	15,108,014	2,019,289	10%	5398
Village	15,399,557	2,310,784	12%	1583
SoccerPlayer	15,585,500	2,496,744	13%	1085
Films	15,369,042	2,280,321	12%	1583

Table 7. Merging different themes of DBpedia into YAGO

We experimentally verified these theoretical results by marking DBpedia with the removal of 2000 facts. Then, we stole different themes from the marked DBpedia and merged them into YAGO. We ran our detection algorithm and obtained significance levels of $\chi^2 = 28, 15, 34, 47$, and 31, respectively. This means that we could successfully detect all thefts, except for the theft of the Television Episode theme. This set is smaller, so that it requires the removal of more facts, as predicted. This shows that our method works even if part of the mark is destroyed.

6 Conclusion

We have presented an alternative approach for the watermarking of ontologies. Instead of altering facts, we remove facts. Thereby, we do not lower the precision of the ontology. We have shown that even on large ontologies, only a few hundred facts have to be removed to guarantee protection from theft. Through experiments, we have shown that our approach is well applicable to real world ontologies. In the future, we intend to explore whether ontologies can also be watermarked by adding artificial facts.

Acknowledgements

This work was supported by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) / ERC grant Webdam, agreement 226513 (`http://webdam.inria.fr/`).

References

1. R. Agrawal, P. J. Haas, and J. Kiernan. Watermarking Relational Data: Framework, Algorithms and Analysis. *VLDB J.*, 12(2):157–169, 2003.

2. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the International Semantic Web Conference (ISWC)*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735, Berlin, Germany, 2007. Springer.
3. C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee. Linked data on the Web. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 1265–1266, New York, NY, USA, 2008. ACM.
4. J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs, provenance and trust. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 613–622, New York, NY, USA, 2005. ACM.
5. S. Grimm and J. Wissmann. Elimination of redundancy in ontologies. In *ESWC*, pages 260–274, 2011.
6. H. Kong, G. Xue, K. Tian, and S. Yao. Techniques for owl-based ontology watermarking. In *WRI Global Congress on Intelligent Systems (GCIS)*, pages 582–586, Xiamen, 2009.
7. J. Lafaye, D. Gross-Amblard, C. Constantin, and M. Guerrouani. Watermill: An optimized fingerprinting system for databases under constraints. *IEEE Trans. Knowl. Data Eng. (TKDE)*, 20(4):532–546, 2008.
8. Y. Li, V. Swarup, and S. Jajodia. Fingerprinting relational databases: Schemes and specialties. *IEEE Trans. Dependable Sec. Comput. (TDSC)*, 2(1):34–45, 2005.
9. C. Matuszek, J. Cabral, M. Witbrock, and J. Deoliveira. An introduction to the syntax and content of cyc. In *Proceedings of the AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, pages 44–49, Menlo Park, CA, USA, 2006. AAAI Press.
10. I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA, 2001. ACM.
11. N. Noy, R. Fergerson, and M. Musen. The knowledge model of Protégé-2000: Combining interoperability and flexibility. *Lecture Notes in Computer Science*, 1937:69–82, 2000.
12. M. Shehab, E. Bertino, and A. Ghafoor. Watermarking relational databases using optimization-based techniques. *IEEE Trans. Knowl. Data Eng. (TKDE)*, 20(1):116–129, 2008.
13. R. Sion, M. Atallah, and S. Prabhakar. Resilient information hiding for abstract semi-structures. In S. Verlag, editor, *Proceedings of the 4th Workshop on Digital Watermarking, IWDW'03*, volume 2939, pages 141–153, 2003.
14. R. Sion, M. Atallah, and S. Prabhakar. Protecting rights over relational data using watermarking. *IEEE Trans. Knowl. Data Eng. (TKDE)*, 16(12):1509–1525, December 2004.
15. F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A core of semantic knowledge - unifying WordNet and Wikipedia. In C. L. Williamson, M. E. Zurko, and P. J. Patel-Schneider, Peter F. Shenoy, editors, *Proceedings of the International Conference on World Wide Web (WWW)*, pages 697–706, Banff, Canada, 2007. ACM.
16. Wikipedia. CSPRNG, 2011.
17. Wikipedia. One-way function, 2011.