Sciences Methods and Technologies International Journal (SciMeTech)

ISSN: 3085-5284, journal homepage: www.scimetech.com

Hybrid Reasoning for Bank Fraud Detection Using RDF Ontology: Conditional Python and SPARQL

Khaoula El Ater[1], Ouayres Oumaima[1] , Abderrahman Chekry[2]

1 Cadi Ayyad University, UCA, Polydisciplinary Faculty of Safi, Department of Mathematics and Computer Science, Morocco.

2 Cadi Ayyad University, UCA, Graduate School of Technology of Safi, LAPSSII Laboratory, Morocco.

a.chekry@uca.ac.ma

How to cite :

Khaoula El Ater, Ouayres Oumaima , Abderrahman Chekry, "Hybrid Reasoning for Bank Fraud Detection Using RDF Ontology, Conditional Python and SPARQL", Sciences Methods and Technologies International Journal (SciMeTech), Vol 2, Issue 2, p 7-13

Keywords:

Bank fraud
Ontology
SPARQL
Python
OWL

Abstract

Detection of bank frauds continues to be a serious problem within the finance industry, which entails the loss of billions of euros per year. In this paper, an innovative framework is introduced which combines ontological modeling with the use of OWL, dynamic rule-based reasoning by applying conditional Python programming, and querying through patterns with SPARQL. The framework makes use of the Owlready2, and it has been implemented as part of the web interface created using Streamlit. It allows manual input or upload of transaction data from CSV files, automatic analysis, and visualization of results. In this research paper, we build on previous studies about fraud detection using ontology. We apply some basic concepts that include ISO 20022, SWIFT, and Customer Transaction Ontology (CTO). The main purpose of our model is to provide an effective and modular framework for fraud detection by incorporating ontological modeling and rules.

I. Introduction

Bank fraud poses a significant threat to banks in the modern age. The fast-paced adoption of internet banking and international payments has created more complex and voluminous financial transactions. Meanwhile, fraudsters consistently innovate their methods to take advantage of weaknesses, causing substantial economic damage and eroding consumer confidence. According to recent estimates, the amount lost due to fraud in the global financial industry amounts to billions of dollars per year.

Fraud detection mechanisms in the past have relied on methods that involve statistical analysis, anomaly detection, or supervised learning techniques [10,11]. Despite showing promising outcomes in various applications, these models suffer from a number of constraints. The first major disadvantage lies in the fact that these models lack clarity regarding their functioning, hence making it difficult for domain experts to understand the reasons behind categorizing certain transactions as frauds Finally, purely data-driven methods may not fully capture the structural and semantic relationships that exist between clients, accounts, and transactions.

Several researchers have attempted to solve these problems using knowledge-driven approaches. Ontologies capture information about the nature of the financial entities involved and the relationships between them in a form that can be processed by machines [1,5,6,7], allowing for reasoning and generation of fraud detection rules using RDF [7], OWL [9], and SPARQL [14]. Some of the contributions in this field include Customer Transaction Ontology, as presented by Auger et al. [1]; setting up systems guided by expert rules, as done by Purohit et al. [12]; and introducing probabilistic ontologies to deal with uncertainties, by Zhao et al. [13]. This research builds on these ideas, combining ontology-based reasoning with rule-based systems and a web interface, implemented in Protégé [14], Owlready2 [2], and Streamlit [3]. Unlike previous work, this study also focuses on practicality and complies with FATF standards [4].

II. Related Work

Detection of fraud cases in the financial industry has received considerable attention using various approaches. The early works have been either statistical or related to classical machine learning approaches. Some of the examples include the work by Kumar and Ravi [10], where hybrid approaches are used to improve credit card fraud detection. Moreover, Zanin and Papo [11] employed network theories to find anomalies in bank transaction flows. Despite showing good results in terms of performance, such approaches needed a lot of data and lacked transparency issues, making them less applicable in regulated contexts.

In recent years, there has been an increase in the use of semantic and ontology-based approaches, which can solve these shortcomings. In particular, ontologies allow accurate modeling of entities, relationships, and rules, which are crucial in performing reasoning tasks, and therefore, are transparent to stakeholders. One example is Auger et al. approach [1], who propose the Customer Transaction Ontology (CTO) that includes the modeling of individuals, accounts, and transactions used for detecting fraudulent behavior. Purohit and Purohit [12] present an ontology-based approach based on the rule-based reasoning method to include knowledge about the domain. Later, Zhao et al. [13] introduce probabilistic ontologies that support uncertainties in online transactions.

Technological progress achieved within Semantic Web has also contributed to the development of the discussed trend. Indeed, standards like RDF [7], OWL [8], and SPARQL [9] allow for representing and processing knowledge with efficiency. Technically, it is possible to implement an ontology reasoner within Python with the use of Owlready2 [2] tool. Interface-wise, Streamlit [3] library is often used for developing apps, thus allowing for evaluating and deploying ontology-based systems.

Compared to the presented works, our current research utilizes two methods of reasoning at once: semantic queries formulated in SPARQL language and rule-based procedures performed in Python. At the same time, focus is made on user-friendliness in that the reasoning is carried out through a web interface. This way, it is hoped to strike a balance between semantic capabilities and ease of implementation according to international best practices, including FATF recommendations [4].

III. Methodology

In this section, the instruments and processes used throughout the research will be introduced.

a. Tools

Below we list the main components used to build the system. These components were selected with modularity, ease of integration, and implementation considerations in mind. In sum, all these tools form the basis for the environment needed for ontology construction, inference, and user interaction.

- **Protégé** – used to design the OWL ontology by defining classes, properties, and individuals.
- **SPARQL** – used to search through the RDF knowledge base and find potential patterns of fraud.
- **Owlready2 (Python library)** – enabled for the manipulation of ontologies using Python by creating instances and annotating suspicious transactions.
  Python – provided the main framework in which the implementation of business logic and reasoning happened.
- **Streamlit** – used to build a simple web app that allows manual entry or upload of CSVs.

b. Steps

This system was developed in five main steps. Initially, the ontology was developed. Secondly, the reasoning part was integrated. Lastly, the system was implemented. The main steps include:

1. Ontology development using the Protégé tool based on Customer Transaction Ontology (CTO).
2. Development of two fraud detection techniques:
   - Rule-based detection method using conditional statements coded in Python.
   - Pattern-based detection using SPARQL queries on RDF data.
3. Combination of the two approaches into Python with Owlready2
4. Development of a GUI using Streamlit (manual input and CSV upload).
5. Analysis and visualization of the results of the fraud detection.

IV. Banking Ontology

The current section describes the modeling approach that was used to build the Customer Transaction Ontology (CTO) based on the methodology proposed by Bouix et al. [1]. The ontology was developed using Protégé and models the essential entities and relationships involved in banking transactions.

a. Class Hierarchy

The class structure is designed to reflect the main actors and entities in the banking domain:

Emetteur ⊑ Personne

Recepteur ⊑ Personne

PersonneFrauduleuse ⊑ Personne

TransactionSuspecte ⊑ Transaction

These subclass relationships enable fine-grained reasoning by distinguishing between normal and suspicious roles within the transaction system, as illustrated in **Figure 1**. **Figure 2** illustrates the OntoGraph of the main classes in the Banking Ontology.

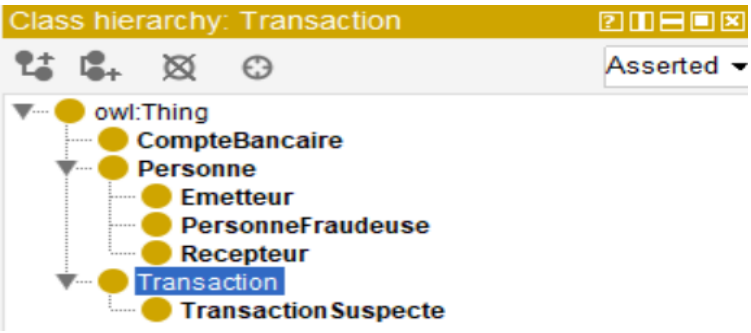

Figure 1. subclass relationships.

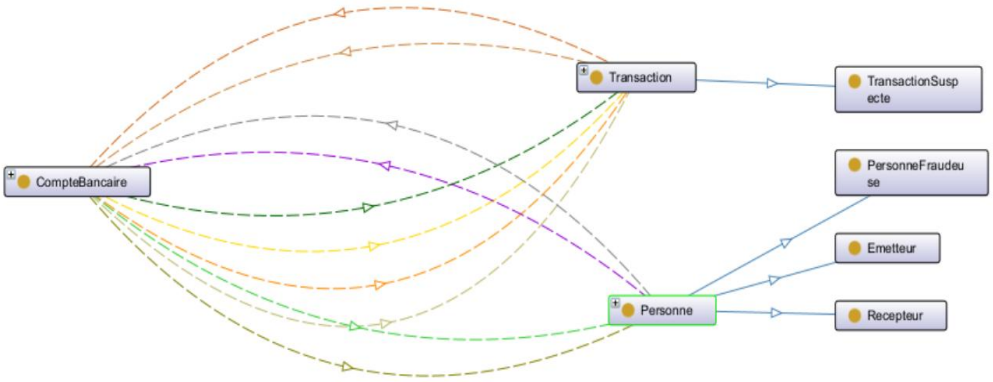

Figure 2. OntoGraph of main classes in the Banking Ontology.

b. Object Properties and Logical Constraints

The ontology defines domain-specific object properties to semantically capture the relationships between individuals (persons), bank accounts, and transactions. For example, the property **possede** links a **Personne** (person) to a **CompteBancaire** (bank account), and is defined as the inverse of **estPossedePar**. Similarly, **effectueTransaction** connects an account to the transaction it initiates, and is paired with the inverse **estEffectueePar**. The ontology also includes **recoitTransaction** and its inverse **estRecuePar**, as well as two unidirectional properties: **envoieVirement** and **recoitVirement**, linking **Emetteur** and **Recepteur** to a Transaction, respectively. **Table 1** shows the properties of the objects along with their respective domains, ranges, and inverses. **Figure 3** illustrates the object properties, and **Figure 4** illustrates the datatype properties. To promote logical coherence and reasoning in the ontology, axioms have been defined in OWL DL. These axioms include existential restrictions and cardinality restrictions on various properties. For instance, a cardinality restriction on the properties estPossedePar. Personne states that every bank account belongs to only one person (CompteBancaire ⊑ (= 1 estPossedePar. Personne)). On the other hand, an existential restriction exists that says that every person has at least one bank account (Personne ⊑ ∃ possede. CompteBancaire). Also, it is assumed that bank accounts can both send and receive transactions, as shown through existential restrictions on the properties envoieTransactionDe and recoitTransactionFrom. The equivalence among various properties and their inverses is also highlighted (e.g., possede ≡ inverse(estPossedePar)).

Table 1. Object properties of the ontology and their inverses.

Property	Domain	Range	Inverse Property
possede	Personne	CompteBancaire	estPossedePar
effectueTransaction	CompteBancaire	Transaction	estEffectueePar
recoitTransaction	CompteBancaire	Transaction	estRecuePar
envoieVirement	Emetteur	Transaction	—
recoitVirement	Recepteur	Transaction	—

Figure 3. Object property.

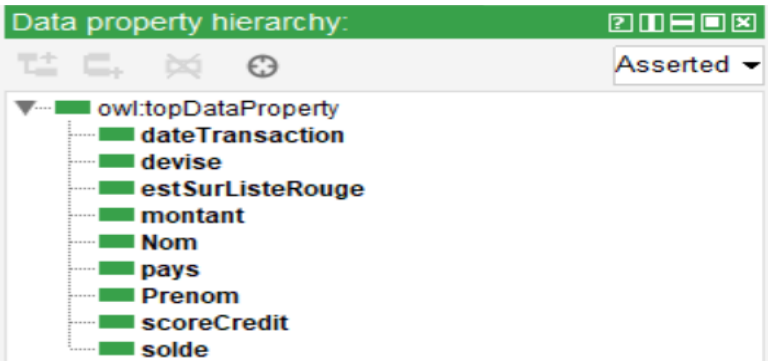

Figure 4. Datatype property.

V. Software architecture

a. SPARQL Query for Fraud Detection

Detection of fraud is done using SPARQL queries run on the RDF knowledge base created based on the ontology. This SPARQL query below detects possible cases of fraud using business rules such as transaction amounts, account balances, and country of origin.

```
PREFIX : <http://www.semanticweb.org/ontologies/banque_fraude#>
SELECT ?transaction
WHERE {
 ?t a :Transaction ;
    :montant ?montant ;
    :devise ?devise .
 ?c a :CompteBancaire ;
     :effectueTransaction ?t ;
     :solde ?solde .
 ?p a :Personne ;
     :pays ?pays ;
     :possede ?c .
 FILTER(?montant > 13000 && ?solde < 10000 && lcase(str(?pays)) = "iran")
 BIND(str(?t) AS ?transaction)  }
```

These transactions will be returned if they meet all of the defined criteria and will then be processed further.

b. Python Application and Interface

To implement the hybrid fraud detection system, a Python application that uses the Owlready2 package has been built. This application is able to use two modes for reasoning:

- **Using SPARQL: Fraudulent transactions can be detected using SPARQL queries on the RDF serialization of the ontology, followed by annotation in Python.**
- **Not using SPARQL: Detection rules are coded using conditional statements in Python, allowing reasoning without the need for querying.**

The application is deployed through a web interface developed using Streamlit, which allows:

- Manual data entry of transaction details;
- CSV file upload for batch processing;
- Visualization of detected fraudulent transactions and ontology instances.

This dual-mode interface provides flexibility and ensures accessibility for users with varied technical backgrounds.

The user interface allows manual data and CSV file upload, with the display of detection results illustrated in **Figure 5**. The implementation of the proposed system is publicly available on GitHub at https://github.com/oumas3/Hybrid-Reasoning-for-Bank-Fraud-Detection-with-RDF-Ontology-Conditional-Python-and-SPARQL.

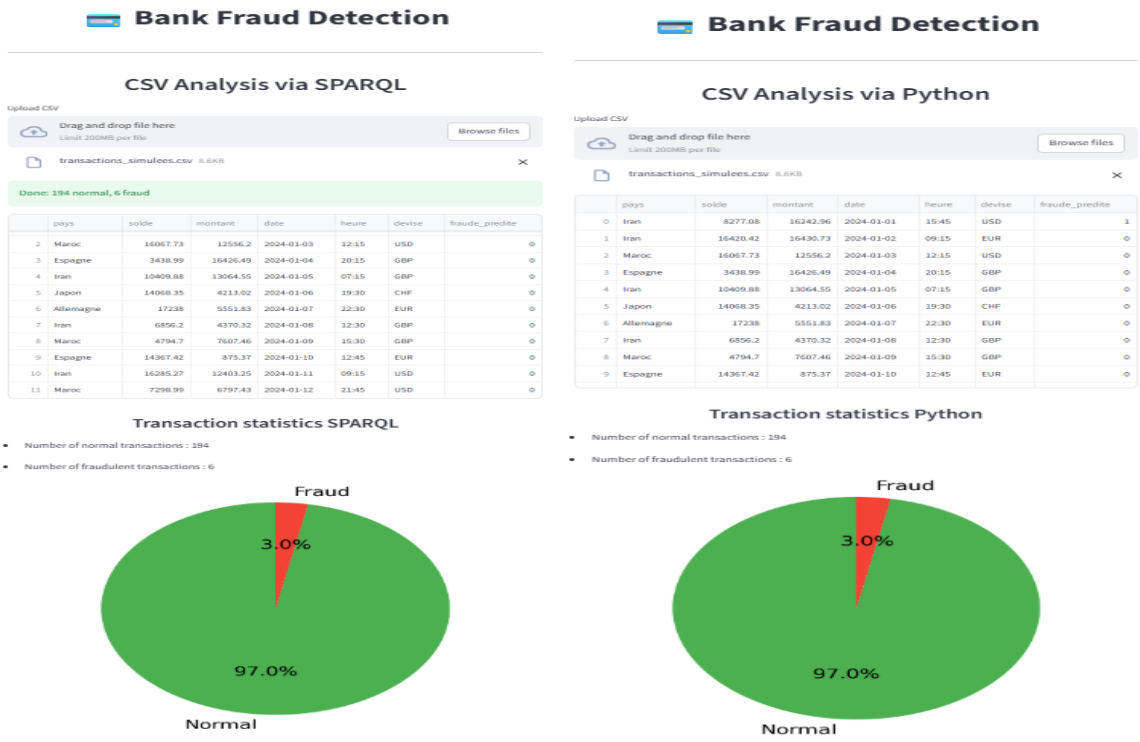

Figure 5. Python reasoning with SPARQL and without SPARQL.

VI. Experimentation and Results

a. Results

For testing purposes, the system was tested on a dataset made up of 100 transactions of banks, both genuine and fraudulent transactions. The results obtained from this test are indicated in Table 2.

Table 2. Fraud Detection Results on a Simulated Dataset of 100 Bank Transactions.

Detection Results		**Value**
Number of fraudulent transactions detected		19
Evaluation Metrics	**Normal Transactions**	**Fraudulent Transactions**
Precision	0.923	1.000
Recall	1.000	0.667
F1-Score	0.960	0.800

The outcomes show that there is high precision in detecting normal as well as fraudulent transactions, though in the latter case, some false negatives arise due to rule sensitivity.

b. Discussion

To guarantee domain relevancy and compliance with regulatory requirements, financial-domain experts were involved, and their input was highly important. These experts have defined the values used for risk threshold parameters (amount, balance, country) in detection rules, validated the ontology structure based on real-world practices from the field of banking, and interpreted the results raised by our fraud detection framework to evaluate their operational relevance. Moreover, these experts have helped us to understand the requirements related to standards such as FATF and ISO 20022.

From experiments performed, it can be stated that hybrid reasoning has great potential for fraud detection. We utilized two different strategies for implementing hybrid reasoning:

- **SPARQL** allows running semantic and declarative queries, that is useful when integrating with other semantic tools.
- **Python rules** allow for quick and easy adaptation, thus ensuring that the solution is customized to meet the business needs.

While working on the same ontology, the two solutions play different roles, SPARQL allowing for semantic accuracy and Python adding to flexibility. The use of the two in conjunction provided better results.

VII. Conclusion

In this research, we propose an approach to fraud detection in finance based on the use of two kinds of reasoning. The first one implies organizing the information about financial transactions in ontologies and using SPARQL queries to extract the needed information. The second reasoning is built upon Python rules, which can be created and modified relatively easily. Both kinds of reasoning are employed simultaneously in a web application designed using Streamlit. Our goal was to develop a tool which would be intuitive enough to work with even for non-professional users. The other crucial aspect of our research relates to creating a system which complies with the relevant international regulation, e.g., the FATF guidelines.

The proposed approach is superior to the existing approaches to fraud detection that are limited to either machine learning or ontology-based reasonings. An important advantage of our hybrid model is the fact that it allows combining precision (thanks to the ontology part) with flexibility (by virtue of the rules written in Python).

As a future research direction, we plan to experiment with different kinds of data, especially bigger and more realistic datasets than the one used for testing our system in this paper. Another line of investigation could include integrating our framework with ML models that will automatically recognize new cases of financial fraud. Furthermore, we would like to expand the current ontology with new classes and predicates.

References

[1] Auger, B., Chergui, H., Chehade, Y., El Kadri, J., Abrouk, L., & Cabioch, N. (2022). Construction of an ontology in the financial domain for fraud detection. INFORSID 2022.

[2] Lamy, J.-B. (2017). Owlready2: Ontology-oriented programming in Python. Knowledge-Based Systems, 93, 57–69. https://doi.org/10.1016/j.knosys.2015.12.024

[3] Streamlit. (n.d.). Streamlit documentation. Retrieved June 12, 2025, from https://docs.streamlit.io

[4] Financial Action Task Force. (2021). Guidance on the risk-based approach to combating money laundering. Retrieved June 13, 2025, from https://www.fatf-gafi.org/

[5] Antoniou, G., & van Harmelen, F. (2008). A semantic web primer (2nd ed.). MIT Press.

[6] Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. Scientific American, 284(5), 28–37. https://doi.org/10.1038/scientificamerican0501-28

[7] Brickley, D., & Guha, R. V. (2014). RDF Schema 1.1. W3C Recommendation. Retrieved June 14, 2025, from https://www.w3.org/TR/rdf-schema/

[8] W3C OWL Working Group. (2012). OWL 2 Web Ontology Language Document Overview (Second Edition). Retrieved June 14, 2025, from https://www.w3.org/TR/owl2-overview/

[9] Prud'hommeaux, E., & Seaborne, A. (2008). SPARQL query language for RDF. W3C Recommendation. Retrieved June 14, 2025, from https://www.w3.org/TR/rdf-sparql-query/

[10] Kumar, R., & Ravi, V. (2007). Fraud detection in credit card transactions using hybrid machine learning models. International Journal of Computer Applications, 21(1), 39–45.

[11] Zanin, M., & Papo, D. (2016). Detection of fraud in banking transactions using complex network theory. Scientific Reports, 6, 34190. https://doi.org/10.1038/srep34190
[12] Purohit, H., & Purohit, R. (2018). A study on ontology-based intelligent fraud detection system. International Journal of Computer Applications, 180(46), 6–11.
[13] Zhao, J., Song, Y., & Wang, H. (2013). Fraud detection in online systems using probabilistic ontologies. Expert Systems with Applications, 40(15), 5973–5983.
[14] Protégé Team. (2021). Protégé Ontology Editor (Version 5.6.5) [Software]. Stanford Center for Biomedical Informatics Research. Retrieved June 15, 2025, from https://protege.stanford.edu/
[15] Turban, E., Sharda, R., Delen, D., & King, D. (2018). Business intelligence, analytics, and data science: A managerial perspective (4th ed.). Pearson.