Privacy and Security in a B2B environment : Focus on Supplier Impersonation Fraud Detection using Data Analysis

Rémi Canillas

N°d'ordre NNT : 2020LYSEI118

THÈSE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de l'INSA de Lyon

Ecole Doctorale N° 512
InfoMaths

Spécialité de doctorat :
Informatique

Soutenue publiquement le 04/12/2020, par :
Rémi CANILLAS

Privacy and Security in a B2B environment: Focus on Supplier Impersonation Fraud Detection using Data Analysis.

Devant le jury composé de :

RONCANCIO, Claudia	Professeur	ENSIMAG	Rapporteure
GIANINI, Gabriele	Associate Professor	UNIMI	Rapporteur
BOUCHENAK, Sara	Professeur	INSA-LYON	Examinatrice, Présidente
CIMATO, Stelvio	Associate Professor	UNIMI	Examinateur
AL BOUNA, Béchara	Professeur	Antonine University	Examinateur

BRUNIE, Lionel	Professeur	INSA-LYON	Directeur de thèse
HASAN, Omar	Maître de conférence	INSA-LYON	Co-directeur
SARRAT, Laurent	CTO	SiS-id	Invité

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** http://www.edchimie-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr INSA : R. GOURDON	**M. Stéphane DANIELE** Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX directeur@edchimie-lyon.fr
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** http://edeea.ec-lyon.fr Sec. : M.C. HAVGOUDOUKIAN ecole-doctorale.eea@ec-lyon.fr	**M. Gérard SCORLETTI** École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 gerard.scorletti@ec-lyon.fr
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** http://e2m2.universite-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES secretariat.e2m2@univ-lyon1.fr	**M. Philippe NORMAND** UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX philippe.normand@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** http://www.ediss-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Curien - 3ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tel : +33(0)4 72 44 82 32 sylvie.ricard-blum@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** http://edinfomaths.universite-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tel : 04.72.44.83.69 hamamache.kheddouci@univ-lyon1.fr
Matériaux	**MATÉRIAUX DE LYON** http://ed34.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction ed.materiaux@insa-lyon.fr	**M. Jean-Yves BUFFIÈRE** INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 jean-yves.buffiere@insa-lyon.fr
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** http://edmega.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX jocelyn.bonjour@insa-lyon.fr
ScSo	**ScSo*** http://ed483.univ-lyon2.fr Sec. : Véronique GUICHARD INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 veronique.cervantes@univ-lyon2.fr	**M. Christian MONTES** Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 christian.montes@univ-lyon2.fr

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Acknowledgment

The author of this thesis would like to thank

Foremost, the steady rock that was always here for me during this time of my life: Charlotte. Thank you so much for being so wonderful.

I would also like to thank my family (and her) for all their love and support. Thank you Mom, Dad, and Lilian, you are everything I have ever needed in a family and even more. Thank you Monique for all these wonderful times at Mérey, where I cannot wait to be back. Thank you Olivier and Alexis for the Sunday lunches and car rides. Thank you Raphaëlle and Clémentine for accepting me as if I was part of the family !

While I was away during the last moments of this thesis, I fondly remember the people that shared an office with me at the lab. So thank you Tarek Awwad, Mohamed Maouche, Rania Talbi, Paul Lachat, Christine Solnon, Frédérique Biennier, Sara Bouchenak, and Mohand-Saïd Hacid, for your insights, discussions, and support. Special thank goes to Romain Deville, that taught me much about the magic of software engineering and system administration. And he is also one hell of a climbing buddy, along with Julien !

I would also like to thank the members and colleagues at SiS-id, for keeping up with my ideas that must have sounded pretty crazy at the time. So thank you Laurent Sarrat, without you none of it would have been possible ! Also many thanks to Rémi Demont, François Agier, Kévin Lainte, Ezaline Descombes, Camille Godart, Sandy Hasni for working with me for the best part of three years. I would also like to thank my new colleagues at Prismea for their understanding and support in the last moments of this thesis: Stéphanie, Dorian, thank you so much !

Also, my regards go to the member of Lampray: Papachef, Yzalium, Mamy & Caiu (and Souricette), for all the creative times we shared, and being such good listeners to the never-ending flow of complaints that a good PhD thesis requires. Similarly, many thanks to Sarah Beaulieu for introducing me to the fantastic world of screenwriting and narration. Music & writing helped me go through the darkest hours of my thesis.

Also a special note for all the people from #ligloo. Thank you for being so far away spatially, but close enough to offer me support. Caiu, Orangeade, clementinator, tiki, Sa8th, Emeraude, Blyste, PoissonRouge, Shaac, keep being so wholesome, and see you soon IRL !

Finally, last but not least, I would like to thank my PhD supervisors: Professor Lionel Brunie and Doctor Omar Hasan. Thank you for showing me what academic research was really about, and for all the other valuable lessons that you imparted to me, that helped me shape my perceptions and my goals for the future.

The wonderful people that impacted my life during this thesis are so numerous to be listed integrally, so, to all of you, from the bottom of my heart, thank you !

Extended Abstract

Supplier Impersonation Fraud (SIF) is a special type of fraud occurring in a Business-to-Business context, occurring when a company supplying goods and services to another company is impersonated by a fraudster in order to trigger a payment to an illegitimate bank account owned by the fraudster. While this type of fraud has been a threat for numerous years, the democratization of digitized B2B transactions and the increase of interconnection between companies through the wide adoption of the Internet have made the task of detecting and leveraging SIF harder and harder. Traditionally, fraud detection has been handled by a team of investigators, or automated expert systems. However, if the current trend in business digitalization holds, the number of transactions to investigate will soon exceed the capacity of human-based fraud detection. Thus, the need for automated fraud detection systems arises. Early automated fraud detection systems are expert systems, based on a set of rules hard-coded by human developers using expert knowledge. Such systems, while mitigating the issues of the number of frauds, are unable to cope with the second major issue introduced by the extensive digitization of B2B trades: the dynamic and rapidly changing nature of frauds. Rule-based automated systems still require extensive human inputs to adapt to new frauds and maintain reliable performances through time, and this cost appears prohibitive when many frauds occur, or when the frauds evolve rapidly. Finally, due to the confidential nature of SIF and the specificity of the B2B ecosystem in which it takes place, there isn't any publicly available formalized framework addressing SIF detection is available, and thus no collaborative effort between companies can take place, as each company wants to protect its expertise in fraud detection from both fraudsters and competitors.

In this thesis, we propose two fraud detection systems based on data analysis as a way to address the issues described above. We first introduce a comprehensive analysis framework for SIF detection systems, and postulate the bank account used for a payment is a distinguishing feature indicating fraud. We then propose two data-driven systems to perform SIF detection: ProbaSIF that uses Bayesian theory, and GraphSIF that relies on graph theory. Each of them uses the historical transactions conducted by companies in order to construct a model of a companys legitimate payment behavior, and then compares targeted transactions to these models in order to assert their legitimacy. As no reliably labeled dataset of legitimate and fraudulent transactions exist, the data models are constructed in an unsupervised fashion. This kind of unsupervised detection system is called Anomaly Detection, as the aim is to assert if a suspicious transaction is abnormal or normal with respect to the usual transactions performed by a company. These two systems are built using traditional Machine Learning processes. First, a training phase is performed, where the historical transactions are used to compute a model, and then a testing phase is conducted in order to assert the legitimacy of each targeted transaction.

The thesis is organized as follows: first, we introduce a comprehensive overview of the B2B transaction process, where client companies and supplier companies exchange payments in for goods and services. We then focus on the definition of a transaction and its different components. We describe the concept of Supplier Impersonation Fraud, using scenarios based on real-life fraud attempts, in order to formalize the issue. We also describe the expert system used by SiS-id, a company specialized in

Supplier Impersonation Fraud detection and mitigation, and the dataset they provide to create data-driven fraud detection systems.

We then introduce ProbaSIF, a SIF detection system that uses probabilistic theory and Bayesian inference to assert the legitimacy of a B2B transaction. ProbaSIF compute the probability of an account to be used by a specific client to pay a specific supplier, and assert the legitimacy of a transaction by comparing the account used with the probabilities. ProbaSIF is proposed with two distinct models: one that focuses on the account's probability to be used to pay a specific supplier by a specific client, and the other that aggregates this probability over all the clients involved with this supplier, thus representing an account's probability to be used to pay a supplier based on the collaborative knowledge of all the clients. We show that ProbaSIF is faster than SiS-id expert system when classifying transactions, and that its results in terms of anomalous frauds are consistent with the expert systems. We also show that the collaborative data of more clients allows to prevent false positives.

GraphSIF, the second SIF detection system that we propose, aims to infer knowledge from the relational properties created by the transactions of a company. First, a sequence of graphs modeling the links between companies and accounts is created by aggregating the transactions. It is called the behavior graph. A targeted transaction legitimacy is asserted by analyzing the patterns formed when it is added to the most recent graph of this sequence. Due to the potentially high number of patterns that can be found in the behavior sequence, a Self-Organizing Map is used to regroup graphs with similar patterns. The classification of a targeted transaction is performed by first clustering the graphs, and then comparing the similarity between the targeted transactions graph with the other graphs of its cluster. This approach introduces contextual features to SiS-id analysis. The model shows good efficiency in terms of computational time needed to create the behavior sequence and to classify the transactions, but requires careful tuning for the performances to be consistent with expert knowledge.

Finally, we propose two directions in order to further the research on the topic of Supplier Impersonation Fraud. Firstly, a major drawback in the adoption of data-driven fraud detection system is that companies owning data are worried to share their data with third parties and potential competitors. In order to mitigate this risk, several solutions have been proposed to ensure the confidentiality of the data during transmission and storage. However, maintaining the privacy of the data during data analysis remains a challenge. Providing privacy-preserving machine learning systems might provide the final tool needed to provide full confidentiality in supplier impersonation fraud detection. Secondly, sharing transactions and payments between a large number of companies allows the creation of a transaction map modeling the financial ecosystem in which these companies interact. The study of this map could yield interesting results not only for supplier impersonation fraud detection but also for economic and financial research.

Résumé

La fraude au fournisseur (Supplier Impersonation Fraud, SIF) est un type de fraude se produisant dans un contexte Business-to-Business (B2B), où des entreprises et des commerces interagissent entre eux, plutôt qu'avec le consommateur. Une fraude au fournisseur est effectuée lorsqu'une entreprise (fournisseur) proposant des biens ou des services à une autre entreprise (client) a son identité usurpée par un fraudeur. Cette usurpation a pour but de récupérer le paiement d'une transaction légitime par le fraudeur. Bien que ce type de fraude ne soit pas récent, la démocratisation massive des transactions numériques a provoqué l'accroissement exponentiel de transactions entre entreprises. Par conséquent, la tâche de détecter les fraudes aux fournisseurs est devenue de plus en plus difficile. Le périmètre de la détection de la fraude au fournisseur s'est développé à un point tel que les équipes spécialisées de professionnels ne suffisent plus à la tâche, et le besoin en systèmes de détection automatisés est devenu pressant pour de nombreuses entreprises. La détection automatique de fraude est historiquement effectuée par des systèmes experts, qui permettent de proposer une première réponse numérique face au volume important de transactions. Cependant, de tels systèmes restent dépendants d'un apport humain important pour la maintenance des règles nécessaires à leur bon fonctionnement. Or, en sus d'accroître le nombre de transactions, un second effet de la numérisation des transactions est d'augmenter la vélocité des échanges entre les entreprises. Les systèmes experts ne sont pas adaptés à la nature dynamique de ces nouvelles fraudes et peinent ainsi à maintenir leurs performances au fil du temps dans un contexte qui évolue aussi rapidement. Enfin, en raison de la nature critique des données nécessaires à la détection de fraudes au fournisseur, peu d'efforts collaboratifs de détection de fraude au fournisseur ont vu le jour à notre connaissance. La raison pour ce fait est la volonté des entreprises de protéger ces données de potentiels fraudeurs, mais aussi de potentiels concurrents.

Dans cette thèse, nous proposons, d'utiliser les techniques et outils récents en matière d'apprentissage (Machine Learning) afin de résoudre à ces différents points, en élaborant des systèmes de détection de fraudes se basant sur l'analyse de données. Nous commençons par élaborer une description formelle d'une fraude au fournisseur, et identifions le compte en banque utilisé dans une transaction comme une donnée permettant de distinguer dans la majorité des cas les tentatives de fraude. Nous décrivons ensuite les systèmes de détection de fraudes dans des domaines similaires à la fraude au fournisseur, et décrivons de surcroît deux systèmes de détection basés sur l'analyse de données: ProbaSIF et GraphSIF. Chacun d'eux utilise les transactions historiquement effectuées par une entreprise pour construire un modèle de données illustrant le comportement légitime de ces entreprises, et compare ensuite des transactions à contrôler à ce modèle afin de déterminer leur légitimité. Ce genre d'approche non-supervisée est appelée détection d'anomalie, car son but est de déterminer si un paiement est normal ou anormal pour une entreprise donnée. Ces deux systèmes se composent d'abord d'une phase d'entraînement où les transactions historiques sont utilisées pour calculer un modèle de données, puis d'une phase de test où la légitimité de chaque transaction considérée est déterminée.

Nous proposons en premier lieu une description du contexte des transactions B2B, où entreprises clientes et entreprises fournisseurs échangent des paiements contre des

biens et des services, en focalisant sur la notion de transaction B2B. Nous décrivons ensuite en détail une fraude au fournisseur, en nous basant sur plusieurs scénarios tirés de fraudes perpétrées sur des entreprises réelles. Enfin, nous décrivons le système utilisé par SiS-id, une entreprise spécialisée dans la détection de fraudes au fournisseur, ainsi que des données mises à disposition par l'entreprise afin de mettre au point les modèles d'analyse nécessaires à la détection de fraudes. Nous décrivons ensuite ProbaSIF, un système de détection de fraudes au fournisseur qui se base sur la théorie des probabilités ainsi que sur l'inférence bayésienne afin de déterminer la légitimité d'une transaction. ProbaSIF utilise la probabilité d'un compte en banque à être utilisé dans une transaction future d'une entreprise pour déterminer sa fiabilité. ProbaSIF est composé de deux modèles distincts: le premier se concentre sur la probabilité qu'un compte en banque soit utilisé pour payer un fournisseur particulier par un client particulier, et le second qui se concentre sur la probabilité d'un compte en banque d'être utilisé pour payer un fournisseur, en prenant en compte tous les clients de ce fournisseur. Nous montrons que nos deux modèles sont plus performants en termes de temps de classification, et plus souples dans la mise en place des modèles que le système expert proposé par SiS-id. Nous montrons de plus qu'utiliser les données de plusieurs clients de manière collaborative permet d'éviter l'apparition de faux négatifs.

GraphSIF, le second système de détection de fraude au fournisseur que nous proposons, a pour but d'analyser les propriétés relationnelles créées par l'échange de transactions entre une entreprise et ses fournisseurs. À cette fin, une séquence de différents graphes compilant tous les liens créés entre l'entreprise, ses fournisseurs, et les comptes en banque utilisés pour payer ces fournisseurs, appelée séquence de comportement, est générée. Une transaction est catégorisée en l'ajoutant au graphe le plus récent de la séquence et en analysant les motifs formés, et en les comparant à ceux précédemment trouvés dans la séquence de comportement. À cause du grand nombre de motifs qu'il est possible de trouver dans une séquence de comportement, une Self-Organizing Map est utilisée pour regrouper les graphes comprenant des motifs similaires. La catégorisation d'une transaction est effectuée d'abord en utilisant un algorithme de clustering sur la séquence de comportement, puis en comparant les similitudes entre le graphe contenant la transaction considérée et les autres graphes appartenant au cluster qui lui a été assigné. Cette approche montre une utilisation inédite du graphe pour modéliser les relations entre entreprises. De plus, en termes de vitesse de calcul, GraphSIF se montre plus performant que le système expert de SiS. En revanche, une optimisation des différents paramètres de l'algorithme est nécessaire afin d'obtenir des résultats cohérents avec ceux du système expert.

Enfin, nous proposons deux pistes pour la poursuite des recherches sur la fraude aux fournisseurs. En premier lieu, un des freins majeurs à l'adoption des systèmes de détection de fraudes basés sur l'analyse de données est la réticence d'une entreprise à partager des données sensibles avec une tierce personne en lien avec ses compétiteurs. Cette réticence pourrait être mitigée grâce à la confidentialisation des données. En second lieu, le partage de transactions financières entre un large nombre d'entreprises permet de modéliser une "carte" des relations que ces entreprises entretiennent entres elles. L'étude approfondie de cette carte pourrait apporter des résultats intéressants non seulement dans le domaine de la détection de fraudes au fournisseur, mais aussi dans le domaine de l'économie et de la finance.

Contents

1

Introduction

Supplier impersonation frauds have been on the rise in the past years, resulting in the loss of hundreds of thousands of Euros in 2018, and ranked 1st most frequent business fraud affecting French companies in the latest survey about criminality conducted in 2019 by Euler Hermes and DFCG , two associations of accountant and financial advisors of French companies.

A supplier impersonation fraud consists in a fraudster impersonating a member of a company providing goods and service to another (called a supplier company) in order to trigger a payment on an account controlled by the fraudster.

Nowadays, more and more companies are using digital tools to process, authorize, or even conduct payments due to the numerous advantages provided by digitalization, including the ability to conduct payment all over the globe in a timely fashion, and to connect with business partners all over the world.

However, digital payments make frauds against companies more effective, firstly due to the difficulty to formally identify and trust remote interlocutors that are sometimes geographically very distant from the company headquarters, and secondly due to the increased speed of wired payments, allowing money to be moved from account to account in a very short amount of time, and thus hindering the process of recovering it after a fraud.

The SiS-id company was created in order to answer this threat by providing a solution for fraud mitigation. SiS-ids solutions combine both fraud prevention, by securing the legitimate banking information of suppliers and acting as a trusted guarantee for B2B payments, and fraud detection by using its expertise to provide an assessment about the potential fraudulence of a payment.

Traditionally, an expert-based approach has been used to provide fraud detection as a service, in the form of rule-base engines built using the experience and domain knowledge of fraud analysts to describe fraud attempts by fraudsters. However, such rule-based detection is typically expensive to build, and even more difficult to maintain, since every new fraud case needs to be investigated and a new rule (or set of rules) have to be added to the system. Moreover, older rules might become obsolete. Finally, new emerging patterns are not automatically flagged or signaled. Thus, such systems require extensive maintenance to maintain their performance through time.

In order to circumvent these issues, a shift has taken place toward data-driven

approaches where the intensive human input needed for expert-based approached is replaced with statistical analysis and model constructed from data collected from the system. The use of such learning techniques allows fraud detection systems to derive knowledge directly from the data.

However, applying such data-driven techniques requires several processes such as feature engineering, and model selection and tuning, in order to build the data model most suited for the task at hand. Due to the sensitive nature of Supplier Impersonation Fraud, very few fraud detection systems are made available to the public, as the issue is usually dealt with by a team of hired collaborators in most companies, rather than researched publicly in research laboratories. Furthermore, due to fact that payments between suppliers and clients reveal valuable commercial information, most of the data needed to create accurate fraud detection systems is not publicly disclosed.

In this thesis, we investigate the problem of Supplier Impersonation Fraud (SIF) detection and design two SIF detection systems that aim to perform SIF detection based on different feature engineering processes and data models, each asserting the legitimacy of a payment based on different features extracted from the data available on the platform. The main idea behind this approach is that, due to the heterogeneity of payment behavior and frauds between companies, a single over-specialized data model is able to accurately detect and describe a single type of fraud, but might be largely missing a different type of fraud altogether.

1.1 Contributions and outline of the thesis

In Chapter 2, we describe in detail the B2B context in which Supplier Impersonation Frauds takes place, and we illustrate the issue posed by Supplier Impersonation Fraud by describing three attack scenarios based on real-life fraud attempts. We then propose a definition of SIF as a fraud targeting companies, where a fraudster takes advantage of the relationship built between a client company and a supplier company to obtain illegitimate payments. In this chapter we also highlight the bank account used to pay the supplier as a discriminant allowing a fraud to be detected. We propose to formalize the definition of SIF and conclude the chapter with a presentation of a SIF detection system's goals and design.

In Chapter 3, we present the tools and techniques used in data-driven fraud detection systems, with a focus on unsupervised techniques. As our work is, to the best of our knowledge, the first to address the issue of Supplier Impersonation Fraud in an academical context, we present systems that have been successfully employed in use-cases similar to Supplier Impersonation Fraud: credit-card fraud, insurance fraud, telecommunication fraud, and online auction fraud. We then highlight the differences between supervised and unsupervised systems and argue that while unsupervised systems are less frequently found in the literature they offer advantages such as a better resilience to concept drift and the ability to adapt to unseen attacks. As both these properties are beneficial for a supplier impersonation fraud detection system, we present in detail 4 families of unsupervised approaches for fraud detection (clustering, probabilistic, SOM-based, network-based) in the remainder of the chapter.

In Chapter 4, we present the expert system built by SiS-id to perform SIF detection. SiS-id is a company that specializes in protecting its clients (other companies) from Supplier Impersonation Fraud. This system is a rule-based expert system that uses rules defined by a team of fraud investigators to provide a legitimacy label to a payment. We describe in detail this fraud detection system. In this chapter we also provide information on the two datasets provided by SiS-id to build data-driven SIF detection system. The first dataset, dubbed the "History", contains payments from client companies to supplier companies performed by SiS-id's users during the last

three years. The transactions of this datasets are not labeled, meaning that we have no information about their legitimacy. A second dataset, dubbed the "Audit" dataset, is also described in this chapter. This dataset also contains transaction data about SiS-id's users, but this time the legitimacy label assigned to the payments by SiS-id's in-house fraud detection system are provided.

In Chapter 5, we introduce our first contribution to SIF fraud detection, in the form of ProbaSIF, a SIF detection system based on probability theory and Bayesian Inference. The goal of ProbaSIF is to distinguish between legitimate and anomalous payments based on the probability of use of the bank account involved in the payment. A first probabilistic model is proposed where this probability of account usage is computed using the knowledge of the client performing the transaction. A comparison with the results of the expert system is proposed. We then propose a second model that uses the knowledge of all of the clients with an history of interaction with the supplier, as a way to avoid unknown legitimate accounts to be classified as illegitimate by the previous model. We compare the results of this second model with both SiS-id's expert system's results and the first model's results, and find a decrease in the number of payments wrongly considered fraudulent (false positives), thus confirming the hypothesis that the collaborative knowledge of SiS-id's users can be used to improve the performance of a data-driven SIF detection system.

In Chapter 6, we describe GraphSIF, a SIF detection system based on the relationship network created by the interaction between a client companies with its suppliers. GraphSIF partitions the past transactions of a single client to create a sequence of graphs modeling its payment behavior through time. These graphs are composed of nodes representing both companies and accounts used to receive payment from the client. These graphs are then transformed into feature vectors based on the relationship created by the payments. These relationships are modeled in the form of connected subgraphs found in the clients transaction graph when the client is ignored. The combined use of a clustering algorithm and a z-score computation is finally used to detect anomalous graphs when a new transaction is introduced. Our explorative experimentation shows a good consistency with the results from SiS-id's expert system, and an analysis of the major differences between the two systems is provided.

In Chapter 7, we describe two research topics relevant to future work on Supplier Impersonation Fraud: privacy-preserving fraud detection and B2B exchange network.. Privacy-preserving fraud detection consists in adapting state-of-the art cryptographic tools such as homomorphic encryption in order to perform fraud detection in a privacy-preserving fashion. This kind of systems might alleviate a major issue in the rise of data-driven fraud detection systems: the sensitive nature of the data needed to build the fraud detection data models. The issue is especially important in supplier impersonation fraud, as the details of the relationship between a client company and its suppliers can be used by both a fraudster attempting to perform an attack, and competitors that might use the data to obtain economical advantages. By using a confidential fraud detection system, companies might be more willing to share their data with potential competitors and thus build more robust SIF detection systems.

The B2B exchange network finds its roots in the transaction graphs analyzed in GraphSIF. Instead of a small graph focusing solely on a single client company, the B2B exchange graph aims to model the interactions between all the companies interacting in a given B2B ecosystem. This new model might yield results not only for supplier impersonation fraud, where it can be used to pinpoint potential next victims of fraudsters, but also in a more exploratory fashion, as, to the best of our knowledge, there is no graph-based model of B2B interactions in the literature. The study of such a graph might yield interesting insights on economical behavior of companies.

1.2 Publication Related to the thesis

- Canillas, Rémi, Omar Hasan, Laurent Sarrat, and Lionel Brunie. **"GraphSIF: analyzing flow of payments in a Business-to-Business network to detect supplier impersonation."** Applied Network Science 5, no. 1 (2020): 1-31. [Can+20]
- Canillas, Rémi, Omar Hasan, Laurent Sarrat, and Lionel Brunie. **"Supplier Impersonation Fraud Detection using Bayesian Inference."** In 6th IEEE International Conference on Big Data and Smart Computing (IEEE BigComp 2020). 2020. [Can+a]
- Canillas, Rémi, Omar Hasan, Laurent Sarrat, and Lionel Brunie. **"Supplier Impersonation Fraud Detection in Business-To-Business Transaction Networks Using Self-Organizing Maps."** In International Conference on Complex Networks and Their Applications, pp. 599-610. Springer, Cham, 2019. [Can+b]
- Canillas, Rémi, Rania Talbi, Sara Bouchenak, Omar Hasan, Lionel Brunie, and Laurent Sarrat. **"Exploratory Study of Privacy Preserving Fraud Detection."** In Proceedings of the 19th International Middleware Conference Industry, pp. 25-31. 2018. [Can+18]

2

Supplier Impersonation Fraud

In this chapter, we describe the context of a Business-to-Business (B2B) transaction system and highlight its differences with the more usual Business-to-Customers (B2C) transaction system. We first define a transaction as the composition of a payment record and an issuance record. We then introduce the notion of transaction protocol, indicating how a client company and a supplier company interact in order to get payment processed. We then describe a payment behavior as the set of transactions emitted between a client and a supplier, using example from real life client/supplier interactions. We then illustrate several scenarios in which transaction protocols are compromised and how it affects the transactions between a client and a supplier. Finally, we describe the problem of Supplier Impersonation Fraud detection in a B2B environment and highlight the bank account used in a transaction as a telltale sign of fraud attempt.

2.1 Business-to-Business Transaction System

We first present a Business-to-Business (B2B) model that sets up the context of our work. "Business-to-Business" refers to all the exchanges of goods and services for payment that are performed by supplier companies to client companies, without direct interaction with individual consumers. It is commonly opposed with "Business-to-Customers" environment or "B2C", where companies offer their services and goods to individual consumers. They differ in the fact that while B2C transactions will tend to be one-shot and emotionally based, B2B transactions are the process of a more rational decision process, and relationships between client and suppliers tend to be continuous through long period of time ([Apr17]).

2.1.1 B2B Transaction

A financial transaction is generally defined as follows ([Bus19]):"Event which involves money or payment, such as the act of depositing money into a bank account, borrowing money from a lender, or buying or selling goods or property." In this section, we aim to complement this definition for the context of B2B transactions.

In this thesis we focus on digitized (or numeric) transactions. A digitized transaction is issued from a client company to a supplier company in order to provide payment for a good or a service. The money is transferred from an account owned by the client

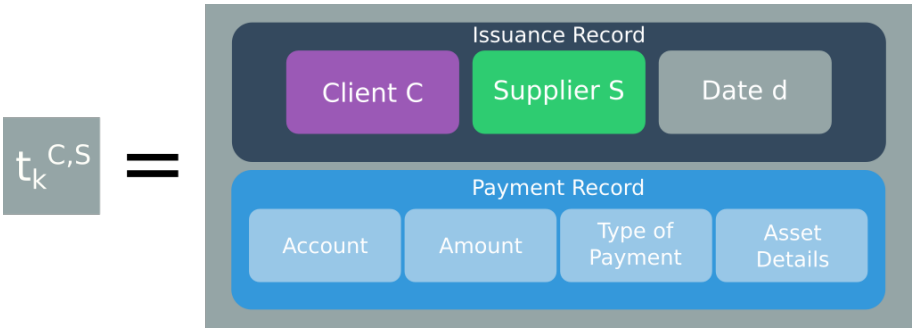

Figure 2.1: Overview of a Business-To-Business Transaction.

to an account owned by a supplier in a purely virtual fashion. A bank account is a financial account maintained by a bank or other financial institution, owned by either an individual or an entity such as a company. This account usually take the form of a digital ledger maintained by a financial institution. The transfer can be of different types: a card payment where a credit or debit card is used as a proof of identity for the transfer, a wiring where the account owner explicitly tell the financial institution to make a transfer to an other account owned by a supplier, or a direct debit where an other entity is allowed to withdraw a certain amount of money regularly on the account. Non-digitized transaction are cash withdrawal and payment by check.

In B2B transactions, the payment for the delivered good or service is usually delayed (as described in [Kap19]), as opposed to B2C transactions where the exchange of the asset and payment is usually performed at the same time. In this work, we focus on the second part of the exchange, namely the payment. In most of fraud attempts, fraudsters only try to capture this payment, and seldom tamper with the other steps of the transaction process.

Intuitively, the payment of an asset by a client company is linked with the nature of the asset, and thus does not exist without the previous interaction between the client and the supplier. In order to reflect this connection, a transaction contains a **payment record**. It is composed of information about the asset exchanged, the type of payment performed, the amount to be paid for the asset, and the account to which the amount needs to be transferred. All this information is usually found in the form of an invoice from the supplier to the client. We use the following definition of a payment record:

Definition 2.1.1 A **payment record** *Pr* is a vector containing the following data:
1. An account *Acc*
2. An amount *Amt*
3. A type of payment *Type*

The second component of a transaction payment is the **issuance record**. It sums up the information related to the emission of a payment from the client to the supplier. The elements of the issuance record are data identifying the client issuying the payment, data identifying the supplier receiving the payment, and the date of the payment. We define the issuance record as follows:

Definition 2.1.2 An **issuance record** *Ir* is a vector containing the following data:
1. A client *C*
2. A supplier *S*
3. A date *d*

Figure 2.1 shows a graphical representation of a transaction. On this representation, we see the distinction between the issuance record and the payment record.

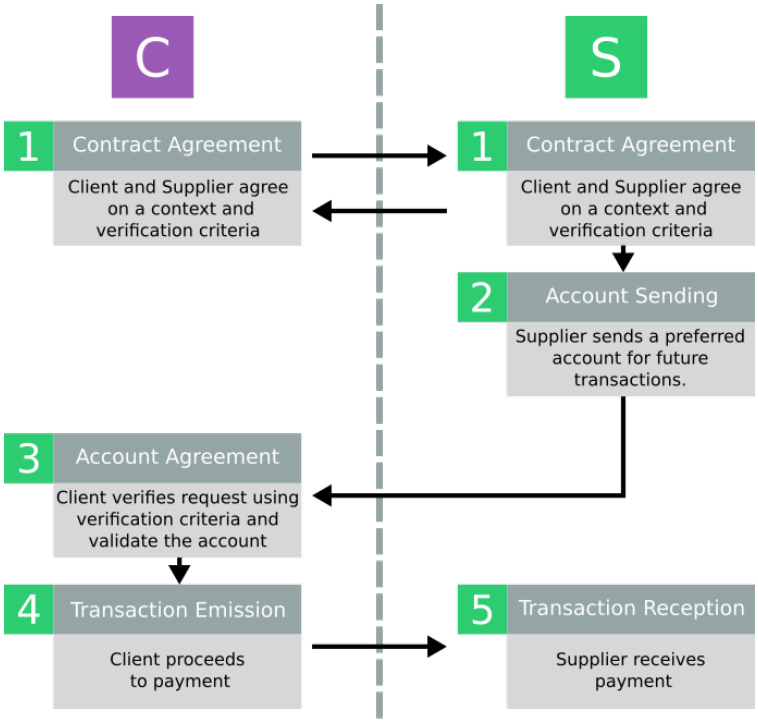

Figure 2.2: Example of transaction protocol - Preferred Account.

We now define our transaction:

Definition 2.1.3 A **transaction** t is a vector composed of 2 elements:
1. An issuance record $I = \{C, S, d\}$
2. A payment record $P = \{Acc, Amt, Type\}$

In this model, the issuing account of the client from where the money is withdrawn, does not appear. It is assumed that this bank account is always legitimately owned by the client. While an analysis of this account might be useful in order to detect money laundering frauds, in this thesis we focus on supplier impersonation and thus the issuing account is not included in the model.

2.1.2 B2B Transaction Protocol

In a B2C environment, it is common for customers to perform a single transaction to a single merchant and then never interact with him or her again. However, in B2B transactions clients and suppliers are usually involved in repeated commercial interactions that build a relationship over days, months or years, as described by Lilien & al. ([Lil16]).

For example, a company producing computers needs a steady supply of electronic parts, thus needs to order these parts from a manufacturer. Usually several of such manufacturers are available for business, but most of the time a company interacts with the ones most suited to its need, selected rationally to optimize the profits. If the company is sufficiently large, then several suppliers might provide the same goods. However, these suppliers are long-time partners with their clients. Moreover, most business projects usually offer rewards only in the long term. This create an extended temporality of business projects, and thus the commitment of the supplier with a client is an important factor. These facts favor the creation and reinforcement of long-term business relationships to build up trust between companies.

The ways of completing a transaction between companies, that we call **transac-**

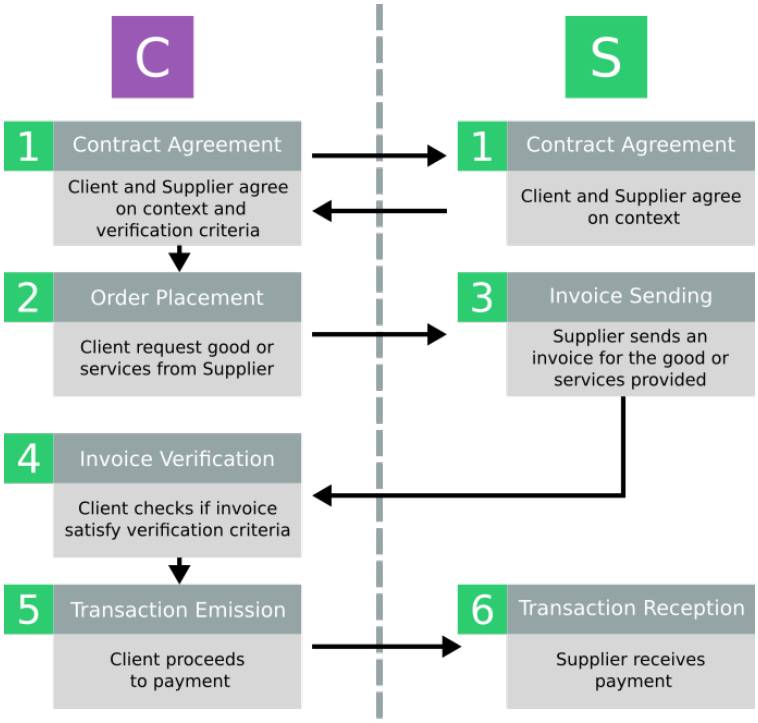

Figure 2.3: Example of transaction protocol - Order/Invoice.

tion protocols, might be as diverse (as described in the study by Lilien & al. [Lil16]) as the companies activity and workflows, and depends on the type of relationships build between client and supplier. Figure 2.2, Figure 2.3 and Figure 2.4 show examples of such transaction protocols.

Preferred Account

Figure 2.2 shows the steps of a transaction protocol dubbed "Preferred Account". In this scenario, the client *C* and supplier *S* agree on a contract defining the conditions of their involvement (Step 1) and then on an account preferred by the supplier (Step 2, Step 3) on which to emit all the following transactions (Step 4, Step 5). This type of transaction protocol is usually found when a client subscribes to a service such as the access to the electrical grid, or the use of a cloud-based storage platform.

Order-Invoice

Figure 2.3 shows the transaction protocol dubbed "Order-Invoice". In this scenario, the client *C* and supplier *S* first sign a contract defining the conditions of their involvement (Step 1) and then conduct transactions only if *C* issues an order for a good or service (Step 2). Upon receiving such an order, *S* produces an invoice for the transaction, and send it to *C* (Step 3). Upon reception *C* verifies if the invoice is valid based on some agreed-upon procedure (Step 4) and then emits the transaction (Step 5, Step 6). This type of transaction protocol is often found in industrial production when ordering materials for the production of goods.

Simple Invoice

Finally, Figure 2.4 shows the transaction protocol "Simple Invoice". In this scenario, the client *C* and supplier *S* define the contract defining the conditions of their involvement (Step 1), then *S* directly sends an invoice to *C* (Step 2). The same verification procedure and transaction emission step from the previous protocol are then repeated

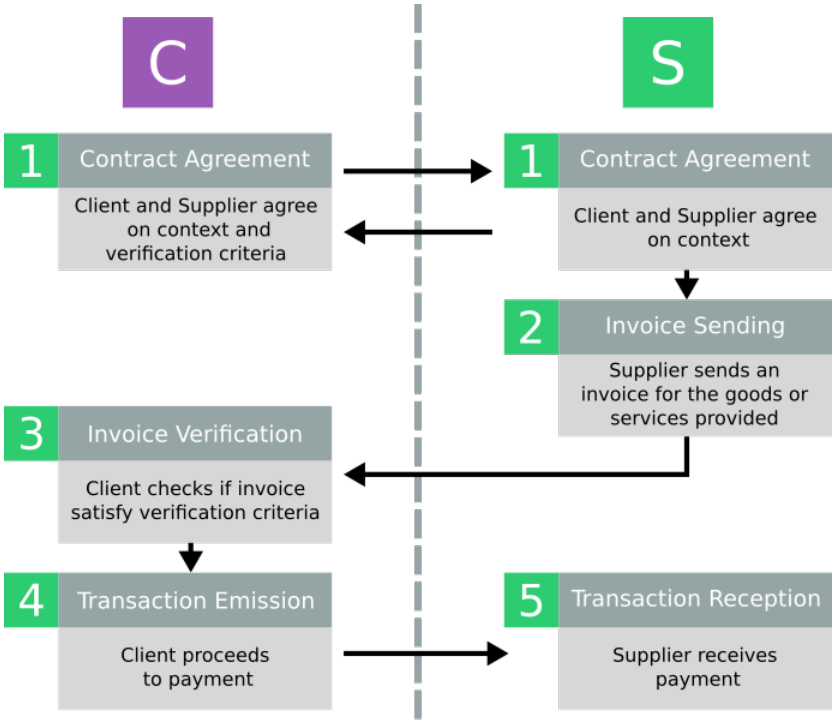

Figure 2.4: Example of transaction protocol - Simple Invoice.

(Step 4, Step 5). This type of transaction protocol is the simplest, and is often used when the supplier's goods or services are required in an irregular fashion.

2.1.3 Payment behavior

In a B2B context, a client can emit multiple payments for transactions involving multiple suppliers. We regroup the transactions emitted by a specific client to a specific supplier in a sequence called **payment behavior**. This sequence is ordered by the date of transaction, and capture the historical behavior of a company with a specific supplier.

Definition 2.1.4 The **payment behavior** P of the client **C** and the supplier **S** is the set of transactions such that $P(C,S) = < t_1, t_2, t_3, ...t_T >$ where $\forall t_i \in P(C,S), t_i.I.C = \mathbf{C}$, and $t_i.I.S = \mathbf{S}$. T is the total number of transactions.

In order to ease the reading of this thesis, each of the component of the issuance record I or the payment record P of a given transaction t_i will be designed with a subscript indicating the transaction associated. For example, the client C of transaction t_i will be noted C_{ti} instead of $t_i.I.C = C$.

The payment behavior represents the business relationship between a client and a supplier. By sorting the transactions in a payment behavior temporally one can then describe the evolution of their relationship.

2.1.4 Business Context

Every transaction protocol of a B2B environment initially needs to set up conditions specifying the relationship created between a client and a supplier company. This step is required to define their business partnership. Usually, the agreement over the business context of the transactions takes the form of a contract signed by both the client and the supplier. The **business context** models this set of conditions over the elements of the payment record.

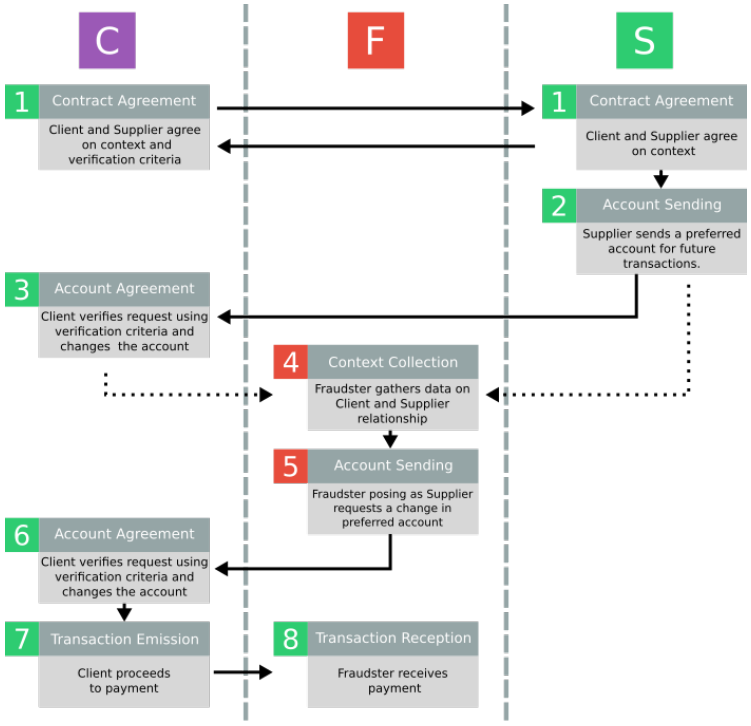

Figure 2.5: Example of attack - Payment Diversion.

These conditions can be diverse as they need to suit the needs of both the company involved in the business relationship. An simple example of such condition can be that the amount of the transaction can not exceed a certain amount a. In this case a condition can be expressed in the form $c_a : Amt_t \leq a \; \forall \, t$.

Establishing a business context between a supplier and a client requires the companies involved to agree on a set of explicit or implicit verification procedures concerning the transactions involved. Most of the time these verification protocols are implicit, as the supplier is trusted implicitly: a representative of the supplier is identified by the client as the interlocutor of reference of the business context, and if any discrepancy is detected in the supplier's behavior the representative is contacted by the client. In these cases, a fraudster only needs to perform a fraud and to pose as a convincing representative of the supplier in order to fool the client. Sometimes, explicit agreement exists between the client and the supplier in order to enforce the business context, in the form of legislative or cryptographic guarantees. In these case a fraudster have to forge such guarantees in order to perform its fraud.

Definition 2.1.5 The **business context** bc between a client C and a supplier S is a set of conditions such as $bc(C,S) = \{c_1, c_2, .., c_N\}$ for any transaction between C and S.

This business context represent the agreement upon by both the supplier and client accept to perform transactions. A business context can contain all the exchange between a client and a supplier, or be negotiated for each good or service provided by the supplier.

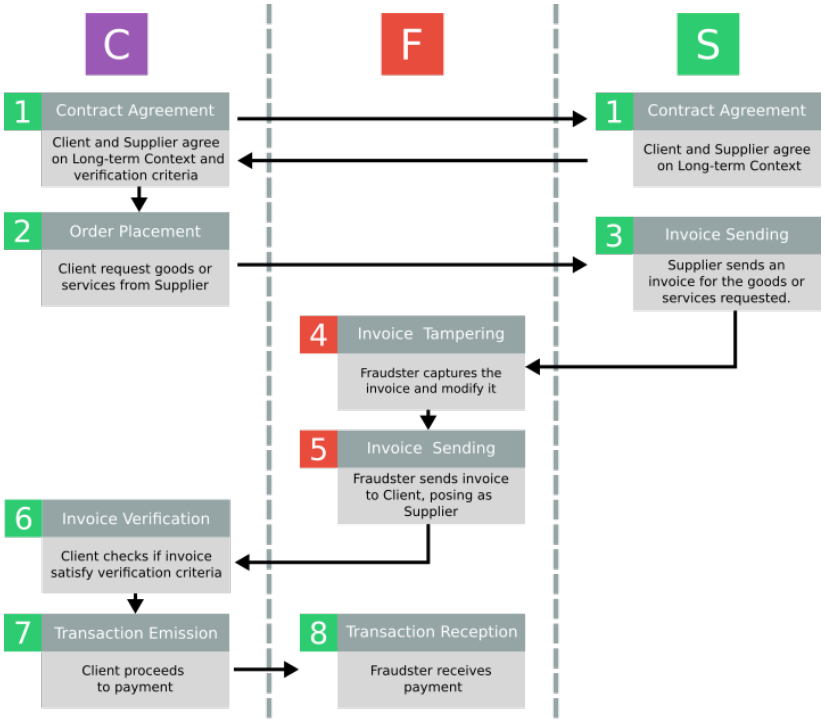

Figure 2.6: Example of attack - Invoice Interception.

2.2 Supplier Impersonation Fraud

A Supplier Impersonation Fraud (SIF) occurs when a fraudster attempts to impersonate a supplier in order to trigger a fraudulent transaction. This attempt can be translated in the model previously defined by tampering with the payment record of a transaction. In order to do so, the fraudster has to convincingly impersonate the supplier so that the client will agree on the change or changes proposed by the fraudster. This change will then trigger a modification of the payment record. As for the supplier itself, he has no knowledge of the impersonation taking place. He thus only notices that a tampering has taken place when the expected payments fail to reach it. In the case of some frauds, a fraud attempt is detected only when the client notices a discrepancy between the invoice and the inventory (for example, if the fraudster charged imaginary goods to the client). In other words, if not detected before (e.g., when the payment record was modified), a fraud can be discovered at the time the client emits a transaction if this latter does not satisfy the business context.

2.2.1 Examples of Attack

In order to perform a SIF, a fraudster usually finds a way to compromise a transaction protocol. Figure 2.5, Figure 2.6 and Figure 2.7 describe real-life case of frauds as described by this study by AIG ([AIG19]) and documentation from the FBI ([Inv17]).

Payment Diversion

In Figure 2.5, the fraudster F inserts a fraudulent actions (red numbering) in between the steps 3 and 4 of the legitimate transaction protocol "preferred account". First, he gathers information about the client C and supplier S in order to gain access to data he can use to give credibility to the fraud attempt. More specifically, this investigation focuses on two aspects: the details of the verification procedure performed by C in order to assert that a supplier account's change request is legitimate (credentials, token exchanged, etc.), and any element that can help the fraudster in its impersonation of S

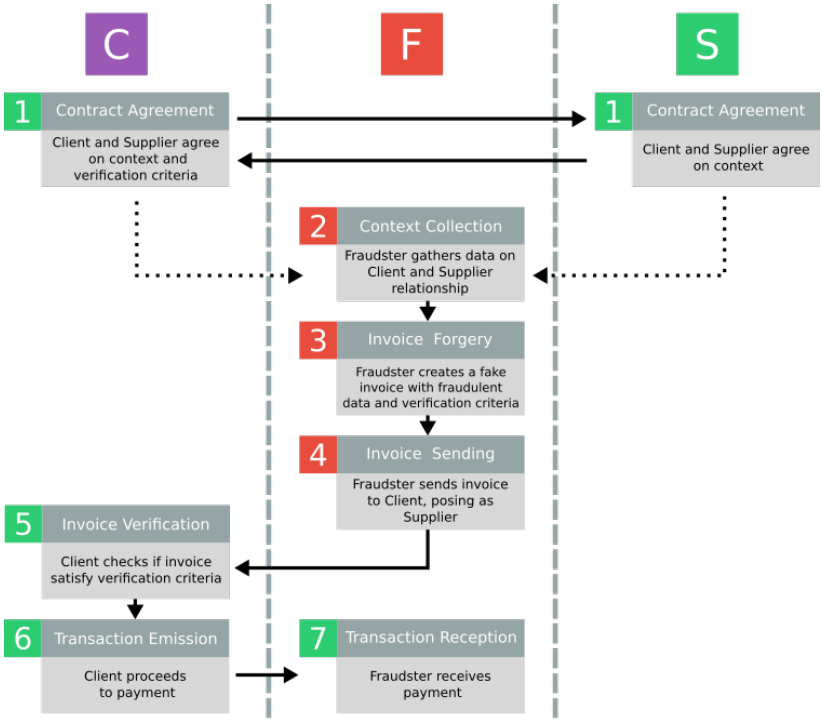

Figure 2.7: Example of attack - Invoice Forgery.

(logo, name of CEO, etc.). The specifics on how the fraudster gathers this data is left out of scope of this thesis, but we assume that the fraudster has the means to successfully perform the fraud attempt. It is a realistic assumption, as such frauds have been performed in the past [Inv17]. Once the data is gathered (Step 4), the fraudster then craft a credible request for preferred account change and sends it to C, while pretending to be S (technically by using S credentials or identity in the interaction). With no way to distinguish between the fraudulent request and a legitimate one, C changes the preferred account and thus all subsequent transactions are then emitted on the fraudulent bank account. This type of fraud is usually called "payment diversion" [Inv17].

Invoice Interception

Figure 2.6 shows how the transaction protocol of "order-invoice" is compromised. In this scenario, the fraudster F also inserts its fraudulent action in between the steps 3 and 4 of the legitimate transaction mechanism. This time, the fraudster intercepts the invoice sent by the supplier S following the order placed by the client C and then modifies it so that it contains fraudulent information (usually a fraudulent bank account) (Step 4). As for the previous fraud, we assume that the fraudster possesses the means to intercept the invoice without being detected, and to tamper with it without breaking the integrity of the potential verification criteria. It then simply sends the tampered invoice pretending to be acting on the behalf of S. With no means to detect that the invoice has been tampered with, the client C then proceeds with the payment of the invoice on the fraudulent bank account (Step 7, Step 8). This type of fraud is often called "invoice interception" ([Inv17]).

Invoice Forgery

Finally, 2.7 presents a scenario where a fraudster compromises the transaction protocol "simple invoice". Contrarily to the previous fraud scenarios, in this one the fraudulent activities can take place directly after the initial agreement between the client C and the supplier S. In this scenario, the fraudster F first gathers information about C and

S as previously described (Step 2). It then uses the information to craft a fraudulent invoice (Step 3) with all the required validation criteria, that is then sent to the client (Step 4). With no means to detect the forgery even though a verification takes place (Step 5), *C* emits the payment (Step 6, Step 7). This type of fraud is usually referred to as "invoice forgery"([Inv17]).

2.2.2 Payment Behavior Tampering

In the previous example, the goal of the fraudster is to divert the payment to an account that he owns. Thus, a supplier impersonation fraud is the action of tampering with the bank account of the transaction to divert the payment to a fraudulent account.

Other type of frauds can occur, for example the fraudster can also tampers with the amount of transaction. This type of fraud is extensively studied in the context of B2C such as in the work of Bolton & al. ([BH+01]). However, this type of fraud is considered to be less important in our model as the fraudster does not obtain a financial gain from the fraud. Its objective might be to alter the trust of the client towards the supplier, or to provoke problems in the B2B ecosystem. This type of destructive fraud is not considered in this thesis.

We thus propose the following definition for the concept of supplier impersonation fraud:

Definition 2.2.1 A **Supplier Impersonation Fraud** or SIF takes place when a fraudster F alters the payment record of a transaction t by changing the account Acc_t of the transaction to an account Acc_F controlled by the fraudster.

As mentioned earlier, other components of the transaction can be tampered with, but such frauds fall outside the scope of this thesis.

2.3 Supplier Fraud Detection in a B2B system

In this section, we give a high-level overview of how Supplier Impersonation Fraud is performed in a B2B context. Based on the definition found in the book by Baesens & al, [BVV15], fraud detection refers to "the ability to recognize or discover fraudulent activites". SIF fraud detection is thus a subset of the broader topic of fraud detection.

In our context of Supplier Impersonation Fraud, we consider that "fraudulent activities" corresponds to the tampering of the account of a payment record, as defined in Definition 2.2.1. In the next sections, we describe how Supplier Impersonation Fraud Detection is performed in a the context of a Business-to-Business ecosystem where several companies exchange goods and services for payments.

2.3.1 Fraud Detection System

In order to protect a user from a fraudster, several strategies can be used. Fraud protection consists in designing systems and verification process that will hinder the fraudster in his attempt to perform a fraud. Such system includes biometric verification, strong encryption, and so on. However, as it is often the case for cyber-protection measures, there is no perfect protection and thus this kind of system can be breached. It is then mandatory to implement fraud detection systems. These systems focus on monitoring the critical process of a target and assess if a fraud is occurring. Thus, a SIF detection system's goal is to detect attempts to tamper with the account of a payment in a specific B2B ecosystem. A more comprehensive description of these type of fraud mitigation is provided in Chapter 3.

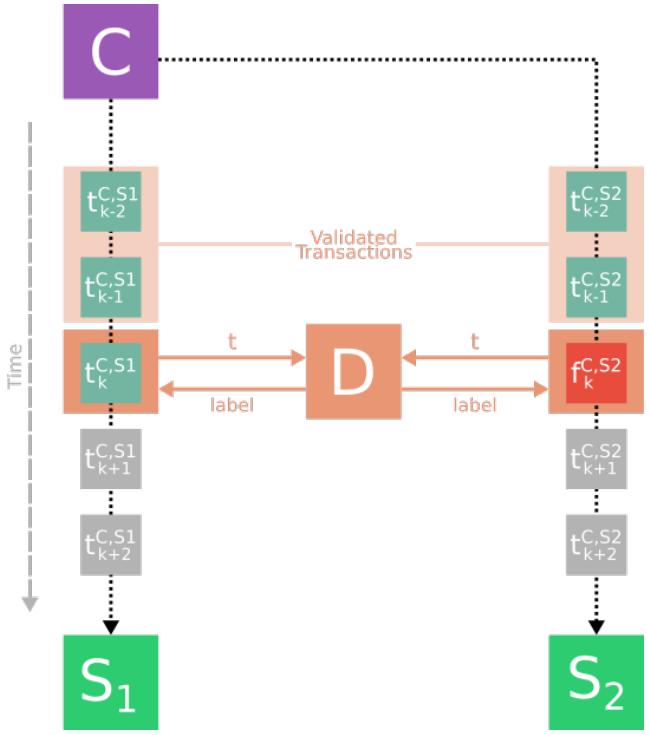

Figure 2.8: Fraud Detection in B2B system.

Traditionally, the detection of a fraud can be performed in two ways (as defined in Baesens & al.'s book [BVV15]): an "ex ante" approach, where the model is used to predict which transaction the client is most likely to perform next, and the legitimacy of the next real transaction is asserted by studying the discrepancy between the predicted transaction and the real one. On the other hand, in the "ex post" approach, there is no predicted transaction, and the real transaction is compared to the constructed model to assert its legitimacy. Furthermore, several modes of prediction exists: batch, near-real time and real-time. In batch prediction, a large set of transactions are analyzed at the same time, usually well after they have been issued. In near-real time, a single transaction is analyzed as soon as it has been issued, but there is not strong guarantees on the analysis time (usually between 1 second and 1 minute). In real-time analysis, the time taken to conduct the analysis is of the essence, so strong guarantees are enforced (response time less than 10 ms).

In this thesis, we focus on the design and implementation of an ex ante, near real-time fraud detection system. This type of system is the more common in the survey conducted on general purpose fraud detection systems, as described in the surveys of Chandola & al. ([CBK09]), Abdallah & al.([AMZ16]) and the book of Baesens & al.([BVV15]). We use this model as the basis of our systems focused on SIF detection as they are the ones that are the most similar to the expert system implemented by SiS-id to detect SIF.

Figure 2.8 shows how a SIF detection system performs. First, an analysis of past transactions emitted between a client and a supplier (i.e a payment behavior) takes place, and a model representing legitimate and fraudulent payments is determined. Then, when a new transaction occurs, its characteristic features are compared with the model previously established. Finally, the detection system assigns a label to each of the new transaction reflecting the potential fraudulence of this transaction based on the model.

2.4 Summary

In this chapter, we first described a B2B transaction in detail by defining the concepts of payment record and issuance record. We then described the properties of a B2B transaction system: the transaction protocols that defines the way a client and a supplier company interact, and the payment behavior that represents a sequence of transaction through time. We presented how the transaction protocols could be compromised by a fraudster, and how this attack is mainly targeted on the account and amount of a transaction. We showed how fraud can be described as a tampering of the payment record of a transaction in order to trigger fraudulent transactions. We then presented the problem of supplier impersonation fraud detection in a B2B system.

3

State of the Art

In this chapter, we propose a review of the available literature on fraud detection. We first give an overview of the problem of fraud detection, then describe different fields related to Supplier Impersonation Fraud where fraud detection systems are studied. We then describe the two main approaches for fraud detection, namely supervised and unsupervised approaches. We then provide a description of the different techniques from the literature focused on unsupervised fraud detection. Finally, we propose a synthesis focused on the specific issue of Supplier Impersonation Fraud.

3.1 Fraud Detection

Fraud is defined by [Bol+02] as criminal deception; the use of false representations to gain an unjust advantage. [Bol+02] argue that the development of new technologies has provided new openings for fraudulent activities. [BVV15] proposes a more detailed definition of fraud as **"an uncommon, well considered, imperceptibly concealed, time-evolving and often carefully organized crime which appears in many types of forms"**. *Uncommon* means that only a minority of cases in a specific setting will be fraudulent. *Imperceptibly concealed* reflects the fact that the fraudsters try to blend in and behave as legitimate users. *Well considered* emphasizes the preparation undergone by the fraudster before performing a fraud. *Time-evolving* means that fraud techniques evolve through time. *Carefully organized* means that the fraudsters often construct complex structures in order to hide their actions, and sometimes involve accomplices and fake accounts. Finally, *many types and forms* means that there exist many applications in which fraud is likely to occur. Mitigating fraud is a major concern for most companies, and usually relies on two complementary strategies ([Bol+02]): **Fraud prevention** (where security guarantees such as credential signatures and reputation systems are implemented in order to prevent a fraud from happening), and **Fraud detection** (where the interactions between elements of the system are monitored in order to detect if a fraud takes place).

Traditionally, fraud detection has been performed using expert knowledge, such as a team of investigators ([Bro+02]). As it is often the case when trying to automate expert-based approaches, the first computer-based fraud detection systems have taken the form of expert systems. These systems use decision rules derived from the investigator experience (i.e., for credit card fraud). However, the last decade has seen the

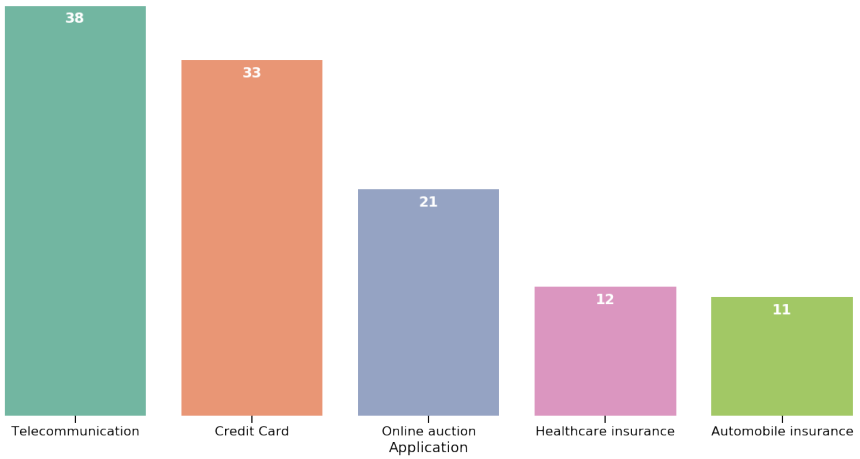

Figure 3.1: Number of fraud-detection related papers from 1994 to 2016 as found in Abdallah et al.'s survey [AMZ16]

appearance of more and more data-driven automated fraud detection systems. In these systems, the knowledge of fraud is not derived from human experience but rather from careful processing and modeling of all available data relevant to the fraud. This knowledge is derived from both internal (i.e., transaction logs for credit card fraud) or external sources (i.e., IP addresses of known fraudsters). The advantage of these systems is that they often offer statistical guarantees in terms of precision, allow detection in a shorter time, and are often less costly to maintain than a set of expert rules ([BVV15]).

In the next subsection, we describe several industrial fields where data-driven systems have been successfully applied for fraud detection.

3.1.1 Fraud Detection Applications Overview

Data-driven fraud detection is a topic that has gained traction over the last years, growing steadily from 333 papers containing this keyword in 2015 to 759 in 2019, according to the website Dimensions [Dim20]. Bolton et al. [Bol+02] and Abdallah et al.[AMZ16] both offer reviews of fraud detection systems based on their domain, and the related monetary loss attributed to fraud in these domains. While fraud is an underlying threat for all business activities, these specific economic fields seem to be the most closely related to the context in which SIF takes place. These domains are the following: credit cards, telecommunication, healthcare, automotive insurance, and online auction. Figure 3.1 indicates the number of research papers related to fraud detection for each domain from 1994 to 2016, as found in [AMZ16]. According to [AMZ16], the most active application field in terms of publication is the field on telecommunication, followed by the credit card application domain, and online auctions. Finally, insurance is divided in two sections: healthcare insurance and automotive insurance. They both have the lowest numbers of related publications. Fraud detection is thus a transverse field spawning numerous research in different domains.

The dynamicity of a fraud domain in terms of research is, however, often more linked to the availability of datasets related to frauds, rather than the research activity in these fields. The distribution of research papers found in Figure 3.1 seems to validate this hypothesis as the telecommunication, credit card and online auction fields are traditionally very digitized. Thus, datasets can be created more easily than healthcare insurance and automotive insurance where the processes are still developing from paper to virtual transactions.

We now describe the types of fraud occurring in these fields, based on the survey performed by Bolton et al. and Abdallah et al. [AMZ16; Bol+02]:

Credit Card Frauds

In the context of credit card fraud, two major kinds of frauds exist: *offline credit card fraud* where the physical card has been stolen and then used by the fraudster, and *card-holders not present* fraud where the card information necessary for an online or phone purchase is copied from a card and then used by the fraudster without the knowledge of the card holder. Offline credit card fraud is usually easily thwarted as the issuing banks will lock the stolen card as soon as the card holder reports it missing ([AMZ16; Bol+02]). However, the cardholder not present fraud is much more difficult to detect, as the cardholder has no knowledge of the fraud until suspicious transactions appear in his bank records and inform the bank. During this time the fraudster can perform any number of fraudulent transactions.

Telecommunications Frauds

Telecommunications frauds are more diverse. One of the most prevalent frauds is the *subscription fraud* where a fraudster subscribes to a telephone service using fake personal information, and with no intention of paying. The *superimposed fraud* occurs when a fraudster takes over a legitimate account to perform calls. A similar fraud is *premium rate fraud*, where the fraudster calls to a premium-rate service number using a subscribers account without their knowledge. Other kinds of frauds involves roaming fraud (the fraudster accumulate roaming charges with no intention of paying), prepaid fraud (a fraudster sells a fake prepaid card without service), SIM surfing (a fraudster steals and use a legitimate users SIM card), SIM cloning (a fraudster duplicate a users SIM card), SIM box fraud (a fraudster uses a device to make calls appear as local), PBX fraud (a fraudster physically hacks into a legitimate users phone line) and voucher fraud (a fraudster tries multiple time to guess a voucher number for free network access), and are thoroughly described in [AMZ16].

Healthcare Insurance Frauds

In this type of frauds a fraudulent patient or healthcare provider tries to obtain illegitimate gains by abusing the insurance system. We focus on the frauds performed by the healthcare supplier as they relate closely to the context of SIF: *phantom claims*, occurring when an healthcare provider presents a bill for a service not provided, *duplicate claims* where an healthcare provider presents the same claims twice, *bill padding* where unneeded services are billed. These kinds of frauds are similar to the ones a fraudster trying to impersonate a legitimate supplier would perform. Other types of fraud can also be found, such as upcoding (presenting claims whose reimbursement value more than the services provided), unbundling (presenting excessive numbers of claims for different services that should be charged as one service), excessive or unnecessary services, kickbacks (collusion between providers and patients to take commission for illegal service), claims in short time (reporting numbers of claims in for same insured in short time), unpaid installments (reporting claims for an insured that has not paid any installments), incorrect dates (reporting claims with incorrect dates that could be prior to or after than the beginning of the insurance period), medication without examination (invoices for medications without the medical check-up or examination), and excessive numbers of small bills (excessive numbers of manual invoice claims whose amounts are smaller than the usual inspection limit). They are also described in detail in [AMZ16].

Automobile Insurance Frauds

The most usual automotive insurance frauds are *ditching*, when fraudsters work to dispose of their vehicles to gain funding, *phantom vehicles* where insurance is issued for accident involving fake, non-existing vehicles, and *staged accidents* where fraudsters organize fake accidents. Other types of fraud are past posting (fraudsters try to get the compensation from insurers based on old accidents occurred before they obtain the insurance), vehicle repairs (billing of new parts on a vehicle when used parts were actually used as replacements), vehicle smuggling (report the car as stolen to collect money from the insurance company.), VIN switch (following a sale and a fraudulent claim, two different vehicles have the same insurance policy and one Vehicle Identification Number), and rental car fraud (fraud perpetrated using rental cars). They are covered in the appropriate section of [AMZ16].

Online Auction Frauds

Finally, online auction frauds contains *competitive shillings*, when two fraudsters collude by competing in order to raise the price of an item, *bid shielding* when a fraudster colludes with two others: the first place a low bid and the other two immediately bid high and keep bidding higher, which is intended to eliminate all other interested parties. At the last minute, the two high bidders drop out, and the low bidder wins by default. *Multiple bidding* occurs when a fraudster use fake accounts to place multiple bid on the same item. Other types of frauds are non-delivery of goods (an auctioneer gather the bids but do not send the goods), false bids (a seller cheats in an auction by looking at the bids before the auction clears and submitting an extra bid), bid shading (the fraudster bids on the auction for an item with a price far below than the item is worth) and credit card phantom (fake transactions for illegal loan sharking through illegal conspiracy between the seller and buyer), as described in [AMZ16].

3.1.2 Predictive and Descriptive Analysis for Fraud Detection

As highlighted by [BVV15], [AMZ16] and [NWY19], data-driven fraud detection systems can be divided into two categories: predictive or descriptive, depending on the way they build their underlying data model: supervised and unsupervised systems. Systems using supervised models are also known as performing predictive analysis for fraud detection, whereas system using unsupervised systems are performing descriptive analysis ([BVV15]).

Predictive Analysis for Fraud Detection

Predictive analysis for fraud detection aims to extract patterns from a set of labeled data, in order to find discriminating features between normal and fraudulent behavior. The goal is to fit the model by maximizing an error function, until the system is able to accurately distinguish frauds from legitimate actions.

The perks of this kind of system are that they are able to identify fraud patterns that might not differ significantly from normal user behavior, and to accurately detect well-known frauds. The drawbacks of these systems are that they need to rely on accurately labeled historical data in order to be fitted, and they are not able to detect frauds that are not part of this training set.

Descriptive Analysis for Fraud Detection

Descriptive analysis for fraud detection consists in finding behavior that shows discrepancy from a normal behavior. These techniques use historical observations to create representation of normal behavior, and then try to identify anomalies (also known as outliers) and report them for further analysis.
The perks of such systems are that they do not rely on a previously labeled set of data

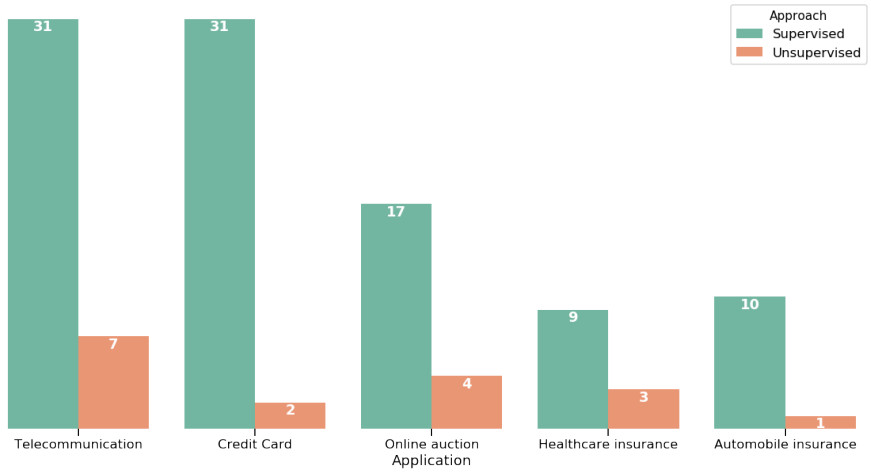

Figure 3.2: Number of supervised and unsupervised systems for each application.

in order to perform their analysis, and thus are able to detect new or unknown frauds. However, such systems have two major drawbacks:

- They often have a high false positive rate. Unsupervised systems model the usual behavior of the underlying system and consequently qualify everything outside of it as an anomaly. If the thresholds indicating how rigid the normal model is too high, legitimate but unusual behavior will be considered as a fraud [NWY19].
- Well-camouflaged frauds might be able to fool such systems. Depending on the features used to describe the system, a camouflaged fraud can be emitted so that it does not immediately appear as unusual and thus is not discovered by the unsupervised fraud detection system [BVV15].

Figure 3.2 shows the distribution of supervised and unsupervised systems for fraud detection based on the research papers surveyed in [AMZ16]. An immediate inference from Figure 3.2 is that unsupervised systems are underrepresented in comparison to supervised systems. This might be explained by the fact that when labeled data is available, supervised systems have better results than unsupervised ones ([NWY19]). The lack of literature concerning unsupervised systems is consistent across all application domains, but the gap is more noticeable in the field on credit card fraud detection (31 supervised systems against only 2 unsupervised).

Fraud Detection for SIF

To the best of our knowledge, the specific application of Supplier Impersonation Fraud has not been explicitly studied in the literature [Dim20], and thus no prior knowledge of the type of frauds occurring in this setting has been compiled. This is most likely explained by the following facts:

- The relationship created by a client company with its supplier is also a very valuable information for the company's competitor. They can use this knowledge in two ways: in a non-malicious way, they identify a reliable supplier in their field. In a malicious way, they can try to subvert the supplier in order to disturb the client company's supply chain.
- Publishing SIF detection system means disclosing the information that a company has been targeted by fraud. All kinds of frauds are perceived by customers and partners as a deficiency in the processes of the company, potentially damaging its reputation [Str19]. Thus, most companies tend to deal with fraud attempts confidentially in order to maintain their reputation.
- SIF occurs in a competitive B2B environment where companies try to outperform their competitors in every way. Thus, not sharing an efficient fraud detec-

tion system with its competitors allows a company to secure an advantage, and motivates secrecy.
- Publicly describing a SIF detection system means that a fraudster can potentially gain knowledge of it and thus design new ways to circumvent the systems. Most companies prefer to keep their system confidential as an additional layer of protection.

These reasons all advocate for the elaboration of in-house, confidential SIF detection systems, and thus explain the lack of publicly available studies in the scientific literature.
Therefore, in the context of Supplier Impersonation Fraud, to the best of our knowledge, no dataset providing reliable labeled fraud exists. In this thesis, we thus focus on unsupervised systems for fraud detection, for the following reasons:

1. We have no labeled dataset containing frauds, thus preventing us from using supervised model.
2. Applying descriptive analysis for fraud detection is a natural first step in order to detect SIF, as it is exploratory in nature. It is a standard practice in the field of fraud detection to combine both predictive and descriptive analysis for fraud detection [BVV15].
3. As described in ([AMZ16]), fraudulent items in a dataset are usually only a fraction of the total population. It is thus reasonable to assert that the data provided allows for the creation of descriptive models will contain more legitimate transactions than fraudulent ones. Thus, the dataset is suited for the construction of unsupervised models of the companies' usual payment behavior.
4. In the case of SIF, we possess a long history of interaction between client companies and suppliers companies, with enough data to infer meaningful behavioral patterns of exchanges and construct pertinent unsupervised systems.

In the rest of this work, we will focus on unsupervised systems. According to [AMZ16] and [BVV15], such systems perform fraud detection by detecting anomalies in data compared to a previously established model. Indeed, fraud is an uncommon phenomenon, and thus considered an anomaly. In order to ease understanding, the words anomaly and fraud will be used interchangeably for the rest of this work. It is technically an abuse of language, as not all anomalies are frauds. However, selecting all anomalies as potential frauds focuses the investigation to only a subset of the data rather than the entire dataset, thus saving resources and time.

3.2 Unsupervised Approaches for Fraud Detection

In the literature, unsupervised fraud detection systems follow the principle of outlier detection. Thus, in order to construct unsupervised fraud detection systems, one has to use an outlier detection algorithm. These algorithms use dense clusters of similar data points (i.e., records from the dataset) to construct a model representing the representative classes during a training phase [Dom+18]. Predictions are then applied to new data by comparing it to the trained model and assigning an anomaly score to the new observations. Surveys [Abd+08; BVV15; Dom+18] from the literature shows that five main approaches are mainly used to perform outlier detection:

- Probabilistic approaches, where the data points are used to fit probabilistic models capturing the behavior of the system, and outliers are elements that have a low probability of occurrence according to the model. Gaussian Mixture Models (GMM), such as the one proposed by [Dom+16], belong to this approach.
- Distance-based approaches, where a distance metric is used to group data points, creating a model composed of high density clusters of data points. Outliers are detected when they appear far from the training data points. Cluster-based approaches and neighbor-based approaches as discussed in [Dom+18] belongs to the distance-based family.
- Neural network approaches, where a model composed of several perceptrons is trained in order to detect an outlier. More specifically, we focus on Self-

Organizing Maps (SOM) based approaches, where the data is used to train a one-layer competitive neural network in order to regroup the data points according to their similarity, and outliers are detected when they activate unusual nodes of the network [ZS06].
- Network analysis approaches that use the relationships created by the data points to model and exploit the interaction between the agents of the system, and outliers are detected when unusual patterns are created in the network. This type of outlier detection approach is illustrated in [Van+15].

In this section we give an overview of the recent literature in these fields, as they are the most closely related to the contributions proposed in this thesis.

3.2.1 Probabilistic techniques for outlier detection

This type of algorithm evaluates the probability density function of a given dataset X, by inferring the model parameters θ that most closely fit the dataset distribution. Each data points belonging to X is assigned likelihood $P(X|\theta)$, and data points having the smallest likelihood are given an outlier label [Dom+18].

Probabilistic techniques (also called "Bayesian" due to the seminal work of Thomas Bayes [Dal05]) have been steadily gaining popularity due to their relative simplicity, and the straightforward interpretability of their results [KL18; RBL19]. A popular type of probabilistic algorithm for outlier detection is Gaussian Mixture Models (GMM) where a selected number of Gaussian distributions are fit to a dataset [Tar+95]. As a reminder, a Gaussian distribution (also known as normal distribution) is a type of continuous probability distribution for a real-valued random variable, whose general form of density function is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

where μ is the mean of the distribution and σ is its standard deviation. The Expectation-Maximization (EM) [Liu11] algorithm is used to train the model by maximizing a lower bound of the likelihood iteratively. This technique is successfully used [Tar+95] to detect suspicious masses in mammograms, and more recently [Dom+16] proposed an application for fraud detection in the context of flight booking. However, a major drawback of the algorithm is that the number of components of the mixture needs to be determined beforehand by data exploration, which can be a complex task. Several nonparametric algorithms based on GMM have thus been developed, such as Dirichlet Process Mixture Model (DPMM) [BJ+06] where a Dirichlet Process is used to infer the number of components at runtime, and Kernel Density Estimator [Tar+95] where the density function of a dataset is approximated by first assigning a kernel function to each data point, then aggregating the local contribution of the kernel. This approach shows good results when applied to outlier detection problems, but are sensitive to the presence of outliers in the dataset when the models are trained.

A recent work of Yusoff & al. [MMA13] showcases the use of GMM to accurately and efficiently construct an accurate account owner's profile in a telecommunication network. They classically use the EM algorithm [Liu11] in order to efficiently perform this fitting of the Gaussians, but propose a pre-processing phase using kernel method in order to select the best hyperparameters for the Gaussian Mixture. However, the authors do not provide the algorithm detecting an outlier from the fitted GMM, only showing the accuracy of their approach by comparing the relative error between the trained GMM and the actual distribution of data points in their dataset.

Probabilistic techniques have also been used in order to perform feature engineering (enriching existing data by computing useful new features from them). For example, in a recent work Lucas et al. [Luc+19], Hidden Markov Models (HMM)

[Rab89] were used to create a set of features that focus on the sequential aspect of fraud attempts in a credit-card fraud detection dataset. The authors propose to enrich a transaction by construction 8 new features. They first propose three perspectives for modeling a sequence of transaction: a sequence can either be composed of genuine transactions or include at least of fraudulent transaction, it can come from a cardholder or from a merchant terminal, and it can consist of the amount of transaction or the time difference between two consecutive transactions (time delta). 8 HMMS were trained, modeling the 4 types of behavior (genuine terminal behavior, fraudulent terminal behavior, genuine cardholder behavior and fraudulent cardholder behavior) for both target variables (amount and time delta). The 8 features added to the feature vector are the likelihood that the sequence of transaction has been generated by a given HMM. While the authors use the labels in order to create 2 of their 4 HMMs, they use unsupervised HMMs to create their features. However, they reintegrate them to their labeled dataset and rely on a supervised algorithm (Random Forest [LW+02]) to perform their fraud detection.

3.2.2 Clustering techniques for outlier detection

This set of techniques uses a distance function to compute pairwise distance matrices containing the distance between each pair of data points in the dataset. This matrix is then used as the basis for a clustering or nearest-neighbor algorithm to create a model of the data. An outlier is detected when its distance to the training data points is computed is high.

The k-Nearest-Neighbor algorithm is widely used for both classification and outlier detection in the literature [CMK08]. In the case of classification, the k-Nearest Neighbors algorithm assigns to the tested data point the label most commonly found among its k nearest neighbors according to a given distance metric. An outlier can be detected by computing a scoring function from the distance between the tested data point and its k^{th} neighbors. The approach has been successfully implemented in [CMK08] with an average accuracy of 0.70 on several publicly available datasets, outperforming more complex models.

The k-means algorithm [DH+73; Ish00] is another popular distance-based algorithm. This algorithm aims to partition the data points into k clusters, where each data point belongs to the cluster with the nearest mean (also called cluster centroid), thus minimizing within-cluster variances. The problem is NP-hard but efficient heuristic algorithms [DH+73] allow for quick convergence to a local optimum. One of the major drawbacks of this algorithm is that the parameter k needs to be determined through extensive data analysis. Ishioka et al. [Ish00] proposes a way to compute the number of clusters autonomously by using Bayesian Information Criterion [KW95].

K-means has been successfully applied by Chang et al. [CC14], where it was used on a set of known fraud attempts in online auctions in order to characterize sequence of known fraudulent behaviors. The authors use a combination of price-related attributes, relationship-based attributes and reputation-related attributes to construct a feature vector recording to a set of transactions that occurred in a specific time frame. They then use clustering ([Ish00]) to characterize the behavior of different types of fraudulent users, and to identify characteristic changes in the behavior indicating when the fraudster goes from dormant to active. The detailed algorithm provides a characterization of four different kinds of fraudsters and proposes a system that analyzes change in behavior of a user in order to assert if they belong to one of these fraudulent cases, by first computing a centroid for each of the fraudulent class, and then comparing the user's behavior with known fraudulent patterns. They evaluated their solution using real data, and achieved high results in their detection of aggressive fraudulent behavior (1.0 precision and 1.0 recall for one type, 0.95 and 0.97 for the second type). The two other types of fraud were less successfully detected (0.12 precision / 0.60

recall and 0.48 precision / 0.81 recall) due to the fact that the fraud attempt contained a "dormant" phase where the fraudster acted as a legitimate user for some time. The authors propose a backward monitoring system to refine the accuracy of their system, adding previous transactions to the analysis in order to detect this kind of profile.

K-means is also impaired by the fact that only numerical attributes can be used in order for the algorithm to perform. Recent work by Budalakoti et al. [BSO09] in the domain of air flight safety proposes the use of the *k*-medoid algorithm [PJ09]. This algorithm is very similar to *k*-means but instead of using the arithmetic mean of a cluster as a center, it uses a data point from the dataset called *medoid* and performs Expectation-Maximization [Liu11] in order to find the medoids which minimize the sum of distances between the medoid and the points in the cluster for each cluster. This algorithm thus computes clusters from both categorical and numerical data, and even show more tolerance to outliers [PJ09].

3.2.3 Neural network techniques for outlier detection

Neural network algorithms for fraud detection are systems composed of a set of artificial neurons organized in layers and connected to each other. Each neuron contains a function that provides a certain output depending on the weights attributed to its inputs. In most of the cases, the activating weights of the neuron are determined using data points available as a training dataset, usually through back propagation [BGC17] when labeled data is available.

Self-Organizing Maps (SOM) [Bul04; Koh89], also called Kohonen networks, are competitive one-layer neural networks composed of a set of neurons (also called nodes) organized as a grid (either rectangular or hexagonal), and each node is fully connected to all the source nodes in the input layer. A training phase first occurs where the weights of each node are updated according to the closest data points found in the training dataset. Then, when a data point is fed to the SOM, the neuron whose weights are the closest to the values of the data point is activated. More details on the workings of SOMs are provided in Chapter 6. SOMs are especially useful to project high-dimensional feature vectors into a 2-dimensional space where distance metrics can then be computed, thus providing a solution to the issue that distances between points starts to lose their meaningfulness in high-dimensional spaces [AHK01].

SOMs have been applied in the context of fraud detection by Brockett et al. [BXD06]. The authors propose to use a SOM to detect fraudulent automobile insurance claims related to bodily injury. In order to do so, they use a set of 127 claims previously labeled by two experts teams: a team of insurance adjusters and a team of specialized investigators. Approximately half (62) of these claims are fraudulent while the others are legitimate claims. The claims consist of a set of 85 boolean indicators derived from the filled-in reports. The proposed system trains a SOM with all the available claims in order to observe if regions with high density of fraudulent claims. Then, when a new claim needs to be classified, the closest node of the SOM is activated and the region to which it belongs is asserted: if it is a fraudulent region then the claim is considered as a fraud attempt. In order to validate their approach, the authors train a set of three feed-forward neural networks (supervised algorithms) that aims to emulate a decision taken by a fraud expert, based on the distribution of fraudulent and legit claims on the SOM. The supervised neural networks are each trained on regions of the SOM with different borders trying to capture a significant behavior in the fraud expert. The results are then compared with the real assessments given by the expert teams. Results show that the neural networks trained on the SOM show accuracy of 62.3%, 59.5% and 63.5%.

Another example of using SOM for fraud detection is provided by Zaslavsky et

al. [ZS06]. They propose a fraud detection system for credit card frauds based on the creation of a cardholder's profile using a Self-Organizing Map (SOM). A transaction's similarity with a profile is computed as the distance between a transaction's features and the weights of the node activated by the transaction. This distance is then compared to a user-defined threshold so that a label corresponding to the fraudulence of the transaction is emitted. A similar approach is used to compare the transaction with a fraudster's profiles if the data is available. The proposed system shows results in terms of average classification error and max classification error using a simulated dataset, based on a very low number of transactions (100). They use three different datasets to validate their results: a set with similar legitimate transactions, a set with legitimate transactions with a few anomalous transactions, and a set with a higher proportion of anomalous transactions. Accuracy fluctuates with the number of transactions used to train the model, but the best average error is 0.0034, 0.0381 and 0.0576 for each of the dataset respectively. In addition, the authors propose a visual interpretation of the SOM in order to define fraudulent and legal zones where specific types of transactions are found.

3.2.4 Network analysis techniques for Fraud Detection

Network Analysis consists in representing the data points and how they relate with each other using a graph. A graph is a data structure composed of edges and nodes, and they are constructed by analyzing the interaction between the various entities in the studied systems. Graph theory [WM03] proposes several different metrics and algorithms to describe the graphs and infer new features for its analysis.

In the context of fraud detection, network analysis is a relatively recent research area that aims to detect collusion of several fraudulent entities, or unexpected changes in the relationships of a user that could indicate a usurpation of identity. In the next section, we describe research papers that use network analysis for fraud detection.

Van Vlasselaer et al. [Van+15] proposes a graph-based detection system combining intrinsic and network features in order to detect credit card frauds. Intrinsic features correspond to features that relate solely to the investigated transaction, while network features are features that can be extracted when investigating the relationship created by a cardholder's past transactions. In order to detect frauds, the proposed system complements 60 RFM (Recency, Frequency, Monetary Value) features with 3 fraud exposure scores (user score, merchant score, transaction score) that represents the graph-wise distance between the cardholder and previously recorded fraud attempts (using the PageRank algorithm). A transaction classification is performed using a random forest algorithm. The authors evaluated the system using a real-life dataset of 3.3 million transactions from an anonymous Belgian credit card issuer, containing 48.000 frauds. The proposed system shows the accuracy of 98.77% and an AUC (area under ROC curve) of 0.986, demonstrating the high performance of the model. Their approach shows the strength of using graph-based features for fraud detection model. However, they use a supervised algorithm and thus their approach is not directly transferable to the SIF detection problem.

Akoglu et al. presents Oddball [AMF10], an outlier detection system that also uses a graph-based approach in order to detect anomalies in real-life networks such as email communications [KY04] or blog post referencing each other. In this latter application, the proposed system was able to detect fraudulent blog posts that keep cross-referencing each other in order to acquire an artificial popularity. The authors focus on the study of egonets in a weighted graph, which are all the nodes directly connected with an edge to a target node. They define several properties of an egonet and verify them experimentally. These properties are Egonet Density Power Law (EDPL), stating that the number of nodes and the number of edges of the egonet follow

a power law [CSN09], Egonet Weight Power Law (EWPL) stating that the total weight of the edges and the number of edges follow a power law, Egonet Eigenvalue Power Law (ELWLP) stating that the principal eigenvalue of the weighted adjacency matrix and the total weight of the edges follow a power law, and Egonet Rank Power Law (ERPL) stating that the rank and the wedge of an edge in the egonet follow a power law (the rank is the place of the edge in the so rted list of edge weights). They use these metrics in order to construct an expected value for each egonet in the considered dataset, and highlight outliers as egonets whose behavior is significantly different than their expected value. They do not provide a statistical analysis of the performance of their model, but instead present their results and discuss the pertinence of the most significant outliers found by their method.

3.2.5 Discussion

As mentioned in Chapter 2, to the best of our knowledge, there is no publicly available research work conducted on Supplier Impersonation Fraud detection. All the papers presented in this chapter present usable unsupervised models for fraud detection, but each of them rely at some point on domain-based knowledge in order to propose systems that are performative. This fact is highlighted in [Dom+18], where it is shown that the studied unsupervised approaches exhibit differences in performances when applied to dataset from different domains. A technique suitable for detecting fraudulent phone calls might prove unusable for credit card frauds detection. The major issue concerning the state of the art in SIF detection is that there is no publicly available frameworks and models for the domain of SIF.

In the absence of such domain knowledge for SIF detection, the questions that arises are: 1) which techniques are suitable for SIF detection ? 2) can we propose a system tailored for SIF that uses these techniques to perform accurate and time-efficient fraud detection ?

In this thesis, we propose an exploratory work using two sets of transactions provided by SiS-id, a company that specializes in SIF mitigation. We base our work on the systems and techniques presented in this section, by adapting them to the domain of SIF. This works consists in providing a basis for future work by designing and evaluating two SIF detection systems specifically tailored for SIF. These systems based on the approaches presented earlier in this chapter, but their goal is mainly to provide the first directions for SIF detection.

4

SiS-id Fraud Detection System

Due to the specific challenges of competitive relationships between companies in a B2B ecosystem, Supplier Impersonation Fraud (SIF) is, to the best of our knowledge, not a publicly researched phenomenon. However, as SIF is a threat to the well-being of the B2B ecosystem ([DFC19]), mitigating solutions have been developed privately. These solutions usually take two forms: either investigation teams inside the company that monitor the transactions according to some internal knowledge, or third-party companies that specialize in SIF prevention and detection, and provide their knowledge as a service for other companies. Such companies include SiS-id [1] and TrustPair [2] in France. As fraud detection is usually dealt with inside a company, the market for third-party fraud detection companies is still in its infancy (both companies were created in 2016).

SiS-id proposes to perform Supplier Impersonation Fraud detection for its clients. Its first goal is to prevent SIF by creating a secure repository linking banking information to a company and ensuring the integrity and reliability of the data. Its second goal is to provide a SIF detection system that asserts a legitimacy level to transactions issued by a company, using a set of past transaction given to SiS-id by the client company.

SiS-id is one of the leaders of SIF detection in France. The remainder of this chapter consists in describing SiS-id's SIF detection system, and the data they shared for this thesis, and identifying issues and challenges for data-driven SIF detection systems.

The next section focuses on the fraud detection system used by SiS-id and provides a detailed overview. Then the raw transactional data provided by the SiS-id's clients is provided. We then describe a dataset composed of transactions evaluated by SiS-id's expert system. Finally, a summary of the issues and challenges of the system is proposed.

[1] https://sis-id.com
[2] https://trustpair.fr/en/

	Feature	Type	Description
1	client-payment-history	$\mathbb{N}$	Number of times the supplier and account are linked in the client history.
2	community-payment-history	$\mathbb{N}$	Number of times the supplier and account are linked in the community history.
3	company-exists	Bool	True if the supplier is registered on the platform.
4	is-administration-company	Bool	True if the supplier belongs to a governmental administration.
5	is-administration-iban	Bool	True if the account belongs to a governmental administration.
6	is-iban-not-fraudulent	Bool	True if the account is not listed in an internal database of fraudulent accounts.
7	match-company-iban-country	Bool	True if the supplier and account are located in the same country.
8	payment-identity-exists	Bool	True if there is a link between the account and the supplier on the platform.
9	ping-iban-siret	$\mathbb{N}$	Score indicating if the account is linked with the supplier in the community history
10	valid-company-managers	$\mathbb{N}$	Score indicating the number of registered users in SiS-id's platform allowed to change the account identification number.
11	legitimacy score	Categorical	Target variable indicating the potential legitimacy of a transaction. Can be "green"(high legitimacy), "medium"(dubious legitimacy) or "low"(fraud).

Table 4.1: Features used by SiS-id's rule engine.

4.1 SiS-id Detection System

The fraud detection system SiS-id currently runs on its platform[3] is an expert-based fraud detection system, where a potentially fraudulent transaction is examined in order to assert its legitimacy, using knowledge available on the platform. This kind of system inherits directly from the tradition of fraud detection teams ([KS12]), and aims to formalize their knowledge in order to efficiently process a large number of transactions. The fraud detection system designed by SiS-id consists in two separate steps: a feature engineering steps where the features from the tested transaction are used to gather additional information, and then the gathered data is matched against a set of expert-defined rules in order to assert the transaction's legitimacy.

4.1.1 Feature Engineering

Figure 4.1 shows an overview of the fraud detection process implemented on the SiS-id platform. First, a transaction is set as input. It contains the identifier of the client emitting the transaction, the account on which the money is transferred, the supplier that is being rewarded with the transaction and the date of the transaction. This transaction first undergoes a Feature Engineering step (A). In this step, 10 unique features are extracted using the transaction data and various data sources. Table 4.1 provides a detailed summary of these features.

[3] https://my.sis-id.com

Algorithm 1: Computation of Feature 1

Inputs :
- C: Identifier of Client
- Acc: Identifier of Account
- S: Identifier of Supplier
- T_C: Set of transactions involving C
- Th: Threshold

Outputs :
- L: Score

Variables :
- $N = 0$: Number of occurrences of couple Supplier-Account

```
foreach T in T_C do
    if A in T and S in T then
        N = N + 1;
if N = 0 then
    L = 0
else
    if N < Th then
        L = 75
    else
        L = 100
```

Contractual Features

Contractual features representing the relationship between the platform and the entity involved in the transaction are computed (Feature 3, 8 and 10 in 4.1). These variables indicate if the supplier issuing the transaction is registered on the platform (Feature 3), if the supplier used the platform to register the account as one of the accounts he owns (Feature 8) and if the supplier is managed by more than one user on the platform (Feature 10). These variables all rely on the same data source: data gathered from the platform logs and databases.

Behavioral Features

Behavioral features representing how the supplier and account are linked in the History dataset are also computed (Feature 1, 2 and 9). Feature 1 is the result of Algorithm 1. This algorithm simply counts the number of occurrences of transactions involving the supplier and the account found in the tested transaction, in a subset of the dataset containing only the transactions emitted by the client. The algorithm outputs a score depending on an external threshold arbitrarily determined by SiS-id fraud detection experts. This score can take the value of 0, meaning that the account and supplier have not been found in the subset of transaction, 75 if the account and the supplier have been found fewer times than the specified threshold, and finally 100 if they have been found more times than the specified threshold. While these values are numerical, they correspond to encodings of ordinal values such as "high", "medium" and "low". In reality, there exists two thresholds Th and Th' with $Th' < Th$. In the configuration chosen by SiS-id's expert, the first threshold Th' is set to 0, and thus only the threshold Th has an impact on the algorithm. This is a strong assumption, as it means that any account associated at least once with a supplier is given at least a "medium" legitimacy.

The computation of Feature 2, described in Algorithm 2, is processed in a similar fashion, with the exception that the subset of transaction used to compute the sum of occurrence of the supplier and account is the whole dataset without transactions involving the client. This second algorithm provides a collaborative perspective on the tested transaction, where the knowledge of other companies familiar with the supplier can have an impact on the final legitimacy score.

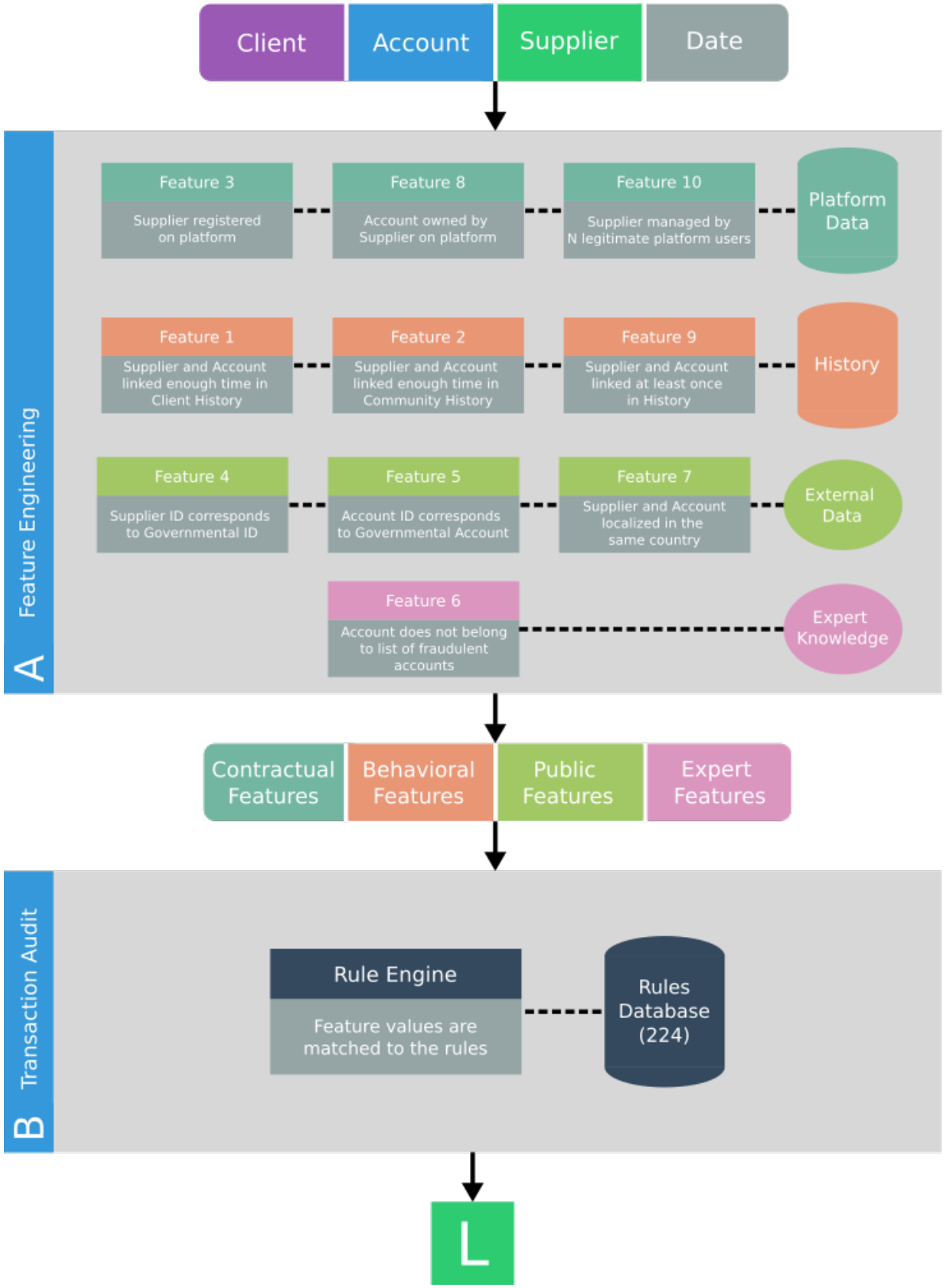

Figure 4.1: Description of SiS-id's rule-based fraud detection system.

Algorithm 2: Computation of Feature 2

Inputs :
- C: identifier of Client
- A: identifier of Account
- S: identifier of Supplier
- $T_{H|C}$: History dataset, without the transaction involving C
- Th: Threshold

Outputs :
- L: Score

Variables :
- $N = 0$: Number of occurrences of couple Supplier-Account

```
foreach T in T_{H|C} do
    if A in T then
        if S in T then
            N = N + 1
if N = 0 then
    L = 0
if N < Th then
    L = 75
else
    L = 100
```

Finally, Feature 9 is simply a binary feature indicating that the supplier and the account have been found together in at least one of the transactions in the History dataset.

External Features

Variables depending on external data source are also computed (Features 4, 5, and 7). These features rely on calls to various openly accessible data repository maintained by governments or public organizations. More specifically, Feature 4 and 5 uses open-access government database in order to derive a binary variable indicating if the supplier is identified as a government organization, and if the account is identified as a government account respectively. Feature 7 relies on the open official company registry of different countries, to identify the supplier's country, and match it with the account country information often supplied as part of its identification number. This country information is provided the identification number is an IBAN (International Bank Account Number), but that is not the case for other local identification numbers (for example, ACH routing number in the USA).

Expert Features

Variables that rely on expert knowledge are also computed (Feature 6). These variables usually depends on structured expert knowledge. In our case, the single Feature 6 is a binary variable indicating if the account is part of a list of fraudulent accounts already found either through expert analysis or data gathering (e.g., from the "dark web").

4.1.2 Rule Engine

Once the feature engineering phase is complete, the newly created feature vector is then fed to the rule engine in the Transaction Audit phase (B). A rule engine is a system composed of a set of If-Then rules devised by fraud experts based on encountered fraud attempts, as described in [BVV15]. These rules are then tested for every new transaction in order to trigger a signal indicating that a SIF has been detected. This process is similar to the Near-Real Time (NRT) fraud detection phase classically used in credit card fraud detection systems, as described in [Gia+20]. In the context of the SiS platform, as of today 224 such rules have been declared in the rule engine. For confidentiality reasons, these rules cannot be publicly shared. However, these rules

aims to assert if the behavior of the supplier companies is the history of transaction is consistent with the data gathered by both the platform and external sources. The rules emphasizes on the stability of the payment process of a supplier, and on its status on the platform.

The output of SiS-id fraud detection system a "legitimacy score" of the transaction (L), which is an ordinal target variable (Feature 11) described in Table 4.1. This target variable can take the three following values: "High" if the transaction is considered legitimate, which is to say no fraudulent input was given to the rule engine. "Medium" if the transaction is considered as somewhat suspicious. This means that given the inputs, the rule engine considers that it does not have enough information to rule out the transaction are either legitimate or fraudulent. "Low" means that the transaction is not legitimate. However, not being legitimate slightly differs from being fraudulent: indeed, transactions whose bank account ID is badly formatted and thus invalid are given a low legitimacy score, whereas a true fraud attempt would have a valid bank account ID.

4.1.3 System Analysis

Rule-based engines are typically time efficient as the transactions can be assigned a score in almost real time as usually rule engines are implemented for this purpose. They tend to be accurate as they rely on carefully gathered expert knowledge of fraud mechanisms. They are also interpretable as the rules, even when they are numerous, are easy to understand due to their simple "If-Then" expressions. However rule engines as fraud detection system suffer from several drawbacks. First of all, as discussed in [Gia+20] they are difficult to maintain. For example, due to the dynamic nature of fraud, new rules have to be added when fraudsters discover new ways to cheat the system. However, when a new rule is added, the consistency of the entire set of rules must be asserted so that no rules are in conflict, or redundant with the new rule. This strongly impacts the maintainability of rule-based fraud detection systems. However, this knowledge is can only be input to the rule engine through the intensive human task of updating all the rules, as the system does not possess a way to correct itself.

Furthermore, the proposed rule-engine uses exactly zero transaction from the history as a base for its decision[4]. This is an issue, as new data is not used by the rule engine to learn new fraud cases. One might argue that the rules are such rules are based on past transactions, as fraud detection experts base their conclusion on analyzing past situations, but in fact, the amount of transaction data now far exceeds the capacity of human analysts. Concretely, this completely neutralizes the capacity of a rule-based system to autonomously discover new fraud patterns and adapt the system accordingly. This can be a serious disadvantage as the new types of frauds would be noticed when end-users start to notify the frauds. Furthermore, simply adding new rules is not as trivial as it sounds, as the new fraud attempts need to be carefully studied, and the issues of rule redundancy and compatibility evoked in the previous paragraph still apply.

Finally, not every relationship between the features and legitimacy level of the transaction can be easily captured in a set of "If-Then" rules. A data-driven model can encompass such complexity more accurately. All these issues strongly advocate for the use of data-driven fraud detection systems. However, these systems should not come as a replacement of the previously existing rule-based engine: they should aim to complement it by providing more variables for the decision process. This approach has already been successfully implemented in the domain of credit card fraud [Luc+19]. Concretely, this can mean complementing the proposed rule-based system with one or several independent, self-contained data-driven systems using more efficiently the

[4]For the case of SiS-id rule engine, the History dataset is used during the Feature Engineering process.

dataset of transactions.

4.2 History Dataset

In its feature engineering process, SiS-id SIF detection system uses a set of historical transactions performed by SiS-id's clients. In this section, we describe this dataset in detail.

The set of B2B transactions used by SiS-id detection system is an aggregation of the payments performed by SiS-id's client companies carried between July 2016 and July 2019. These transactions have the form a feature vector of 4 features : client identification number, supplier identification number, target account identification number, date of the payment. All of the transactions involving the same client, supplier and destination account during a single month are aggregated, resulting in the creation of a fifth feature representing the number of similar transactions issued during the month. The unitary transactions are not kept in the final dataset. The time granularity of the transaction is thus a month. This aggregation is motivated by technical considerations concerning the size of the database needed to store all the activity from SiS-id's clients in a manageable way. Furthermore, B2B unitary payments from clients to suppliers usually have a monthly granularity (especially when dealing with supplies or renting) and thus this aggregation was considered by the experts to have no significant impact on the results of the expert system. With no access to the unitary transactions, there is no means to challenge this assumption. However Figure 4.2 shows an overview of the features.

In order to preserve the confidentiality of the data, a secure hash function with a salt is applied to the three distinct identifiers (client, supplier, account, each with its own salt) so that no link can be established between the data in the history and real-life companies. The link between "clear" and "hashed" company identifiers is thus broken by the added salt. While the combined use of the salt and secure hash function mitigates the risks of damage in case of data leak, it also means that the same company will have a different identifier whether it has issued a transaction, or received a transaction. This means that no "loops" of payments will be found (where two companies are both supplier and client of each other) as the same company will be differently identified when assuming the role of a client, and assuming the role of a supplier.

At the time of writing, 3 712 001 transaction records are available in the history dataset. These transactions are issued by 6 063 unique companies. This number is more than the number of SiS-id's client companies. This is explained by the fact that SiS-id's client companies can represent a group of several companies such as a multinational group. In this case, each firm possesses its own identification number, but only a global entity will be SiS-id's client. The mean number of transactions by client company is 612.24, ranging from 244 214 to 1 transaction(s) per client.
215 056 unique supplier companies are also found in the dataset, along with 262 157 unique bank accounts to which payment was transferred. The fact that more bank accounts exist than supplier companies indicates that some suppliers use more than one bank account to be paid.

This dataset represents all the transactions performed by all of SiS-id clients for two years. However, there is no available information about the legitimacy of these data, and thus no knowledge of which are frauds and which legitimate.

Feature	Type	Description
Client	Nominal (ID)	Identification number of the client issuing the transaction.
Supplier	Nominal (ID)	Identification number of the supplier receiving the transaction.
Account	Nominal (ID)	Identification number of the bank account to which the money is transferred.
Date	Timestamp (Month)	Timestamp indicating the date when the transaction took place.

Table 4.2: Structure of the History data.

4.3 Audit Dataset

A second set of transactions is available through SiS-id. It consists in the list of transactions that were analyzed using the expert system in the past 2 years (July 2017 - July 2019). The dataset, called the "Audit" dataset, is composed of 218 325 suspicious transactions submitted by 317 unique client companies. The transactions underwent the fraud detection process previously described, and labeled with a legitimacy score associated with a color: green indicates that the transaction has a high chance of being legitimate, orange meaning that the rule engine lacks the necessary information to assert if the transaction is legitimate or not, and red meaning that the transaction's legitimacy is low, which can be the case if the transaction is either invalid or fraudulent.

This dataset possesses the following properties:

1. It contains classification requests for fraud detection made by client companies of SiS-id, representing real-life scenario of SIF suspicion.
2. Each of the transaction of this dataset contains a label corresponding to the classification done by the rule engine.

The target variable of the dataset might lead us to use this dataset to perform supervised learning. However, this approach as a major drawback: by using data analysis on a dataset that is the result of the rule engine, we will only manage to "rediscover" the rules. However, this dataset might be used as a validation set for other fraud detection systems, in order to compare their findings with the ones from SiS-id's expert system, and thus investigate the potential convergence of their results with expert knowledge.

4.4 Dataset Analysis

We perform a preliminary analysis of the two datasets to derive some insights of the underlying transaction system.

The distribution of risk labels in the 218 325 transactions processed by the SiS-id expert system was computed. This dataset contains 62.84% (137 188) of the transactions labeled as "green", 22.53% (49 205) labeled as orange, and 17.20% (37 562) labeled as "red". A transaction with a "red" label might be a fraud, but it might also be an invalid transaction whose Account ID is not consistent with the correct format.

We also investigate the overlap of data between the History dataset and Audit dataset. The leftmost part of Figure 4.2 shows that the two datasets contains transactions cumulatively emitted by 6 380 distinct clients, 6 063 from the history dataset and 317 from the audit dataset. 154 of these clients are shared between the two datasets, and 163 are found only is the audit dataset, meaning that they haven't issued any transaction in the history dataset yet.

Additionally, we see that the two datasets contains transactions addressed to 303 036 different accounts, 262 157 from the history dataset and 40 879 from the audit dataset. 30 108 accounts are shared between the two datasets, and that 10 771 of the

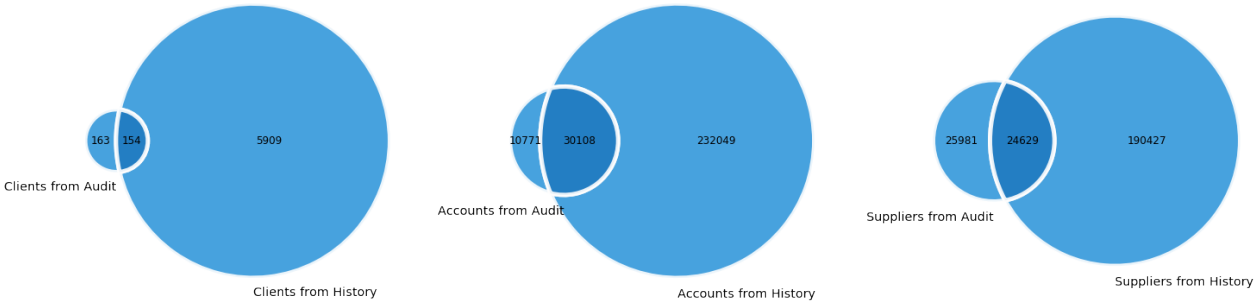

Figure 4.2: Distribution of client companies, supplier companies, and accounts in the two datasets.

accounts are found only in the audit dataset.

Finally, the last figure shows that 265 666 suppliers can be found cumulatively in the two datasets, 215 056 from the history dataset and 50 610 from the audit dataset. 24 629 of these suppliers are shared between the two datasets, while 25 981 are found only in the audit dataset.

4.5 Discussion

While encountering unknown suppliers or accounts in the audit dataset is not surprising, as they indicate that new relationships have been established, the fact that some clients are not present in the history dataset indicates that we do not have previous historical data about these companies. Thus it is not possible to construct models from their past behavior, as no data is available to train these models. Furthermore, if a client company does not have transactions in the audit dataset, there is no way to compare the precision and recall of any new data-driven model with the decision from SiS-id's expert system. We thus focus on the subset of 154 client companies shared between the History and Audit datasets in the rest of this thesis. Suppliers and accounts from the Audit dataset not found in the History dataset means that our fraud detection system has to be able to adapt to unseen values.

The label for transactions used in the Audit dataset was emitted by SiS-id's expert system. As seen in the description of the expert system, both *fraudulent* transactions and *invalid* transactions are issued a "low" legitimacy label. This fact explain the unusually high amount of transactions labeled with the "low" legitimacy label (20.3%). Thus, it raises the question of the quality of these labels. We chose to consider them as a baseline from a "state of the art" fraud detection system, rather than a ground truth. Furthermore, these labels have been emitted in a production setting of SiS-id's fraud detection platform, and as such their results have been accepted by SiS-id's clients. However, no dataset with ground truth on truly fraudulent (and not invalid) or legitimate transactions is yet available for SIF detection.

4.6 Challenges & Issues

While extensively used in the last decades to perform classification tasks, rules-based systems such as the ones used by SiS-id suffer from several drawbacks. [BVV15] accurately list these drawbacks:

1. Rules engines are cost-intensive to maintain: adding or removing a new rule must be done manually and any modification has to be checked for potential in-

consistency or regression in the results. In the context of SIF, such maintenance is necessary because of the constant evolution of frauds. This means that the precision of the system will decrease as new fraud cases arise and are labeled as legitimate, if they appear as such. Furthermore, when attempting to construct a legitimacy score for a given company, SiS-id's expert system tries to capture the legitimate behavior of the targeted company. However, this behavior also evolves through time as the company grows or shrink depending on how successful it is. Concretely this means that new legitimate but unseen behavior will be labeled as fraudulent, and thus the recall of the system will decrease and more "false alarms" will be detected.

2. Rules engines usually monitor systems as a whole, as opposed to local profiles such as the ones in ([BH+01]). However, due to the complexity of relationship between companies, rules that might be true for the whole system might prove false for a specific company.
3. Fraudsters also evolve and adapt their strategies to avoid detection. Thus, a skilled fraudster might be able to infer some rules by trial and error, and thus circumvent the fraud detection system.
4. Rules are usually set up by experts *after* a fraud has been detected and identified. This fact, paired with the time it takes to update the system, means that a fraudster can continue to perform similar frauds for a certain amount of time before the fraud is effectively neutralized.

These reasons advocate for the use of a data-driven system in collaboration with the expert-driven system. Indeed, the maintenance of data-driven systems consists in training a model instead of manually adding or deleting rules, and thus can be performed timely without human intervention. Furthermore, data-driven systems are also prone to discover evidence or patterns of frauds overlooked by human experts, as they process far more data points than humans. Such new evidence can then be added to the decision process and thus refine the expert system.

5

Supplier Impersonation Fraud Detection using Bayesian Inference

In this chapter, we introduce ProbaSIF, an unsupervised supplier impersonation anomaly detection system based on two probabilistic models. ProbaSIF uses a set of historical payments from a Business-to-Business ecosystem involving several companies acting as clients and suppliers. The goal of ProbaSIF is twofold: Firstly, identify the behavior of a company emitting payments (called the client company) and of a company receiving payments (called supplier company). Secondly, produce an alert if an unusual transaction is emitted by a client company for a supplier company with respect to both of their behavior models, indicating a potential impersonation fraud. This alert can then be transmitted to both companies' fraud investigation teams in order to validate or cancel the payment.

In order to be consistent with the expert system that considers the *legitimacy* of a transaction rather than its *fraudulence*, legitimate transactions are in our case considered as "positives" and fraudulent transactions are considered as "negative. While slightly unusual in the context of fraud detection, using this terminology has no impact on the experimentation.

Our system first uses probability theory to compute probability distributions representing the underlying payment behavior of a client and a supplier. ProbaSIF proposes two distinct models: one relying only on the information available for the client conducting the transaction, in order to reflect a single company's view when assessing the potential fraud, and the other one using all the information related to the supplier receiving the transaction. A probabilistic distribution is used for both model, and the probability of occurrence of the account used in a new transaction is determined. This probability is then compared with a risk threshold in order to assert the legitimacy of the transaction.

The motive behind using a model relying on the supplier company's knowledge rather than the client company's knowledge is that a supplier company can use multiple legitimate accounts with different clients, and that such a legitimate account might

appear illegitimate if it has seldom been used with a specific client. The direct result of using only the client company's knowledge is that legitimate transaction for the supplier is categorized as fraud when compared with the model. Thus, by using the collaborative knowledge of several clients of a supplier, we assume that less false negatives will occur. We propose to use the following experiment to verify that hypothesis.

ProbaSIF is evaluated using a set of real transaction data provided by the SiS-id platform. A first evaluation is performed on a dataset previously labeled by SiS-id expert system. The results of the first model are compared with expert knowledge to provide an exploratory analysis based on its consistency with expert knowledge. Then, in a second time we evaluate our second model on the same dataset and show that the results show a reduction in the number of transactions labeled with a low legitimacy label, thus validating our hypothesis in the presented case.

The contributions of this chapter are the following:

1. A probabilistic model describing the occurrences of an account in a set of transaction between companies interacting in a B2B transaction ecosystem, based on a client company's knowledge.
2. A probabilistic model describing the occurrences of an account in a set of transaction between companies interacting in a B2B transaction ecosystem, based on a wider supplier company's knowledge.
3. A classification system attributing a legitimacy label to a new transaction in an interpretable way based on a probabilistic model..
4. An exploratory analysis comparing the results of data-based SIF detection systems to an expert system.
5. An experimental result showing that using a collaborative approach reduces the risk of false negative in the context of SIF.

5.1 Background and Motivation

In this section we provide a focus on the use of probabilistic models in fraud detection. State-of-the-art examples of probabilistic fraud detection systems have been given in Chapter 3 Section 3.2.1. As a reminder, this type of algorithm evaluates the probability density function of a given dataset X, by inferring the model parameters θ that most closely fit the dataset distribution. Each data point belonging to X is assigned likelihood $P(X|\theta)$, and data points having the smallest likelihood are given an outlier label [Dom+18]. We draw from the frameworks developed in the described systems in order to propose a system adapted to SIF detection, and especially from credit-card detection systems such as the one presented in [Luc+19].

The study of the probabilistic models for fraud detection in the literature yields the following motivation to use them:

- These models stand out for their relative simplicity and interpretability [GS13], which is a major point when investigating fraud as they have to be studied by investigators and explained to victims.
- These techniques have also been proved efficient in terms of computational time, even when used with large datasets [Den+02]. The issue of computation time arises, because the dataset used in this thesis (the History dataset described in Chapter 4) is composed of real-life transactions, is bound to grow as more transactions will be performed in the B2B ecosystem, and more companies will be monitored.
- Probabilistic models do not require any pre-processing of categorical data in order to construct their model. This fact is particularly important in our case as most of our data is composed of the company and account identifier which are categorical data.

Additionally, probabilistic models have also been democratized thanks to the worldwide adoption of the Naïve Bayes Classifier [Ris+01] and thus lead to extensive use

and research in various topics.

The successful application of the Bayesian model can be found not only in fraud detection but also in various applications and topics such as modeling animal survival [BCM+00], detecting fake ratings in an online shop [Hoo+16], or detecting faulty batteries in satellites [GHS+19]. While not directly related to SIF detection, this numerous literature means that probabilistic models can be adapted to various use cases and scenarios. Thus, it is a good starting point for a SIF fraud detection system.

5.2 Naive Probabilistic Model

In this section a probabilistic model called the Naive Probabilistic Model is described. This model aims to detect if the account used by a client company to pay its supplier is anomalous with respect to the previous transactions issued by the client. This approach is motivated by the fact that the clients are particularly well informed about how they pay their own suppliers, and thus aims to model this knowledge in a useful way for fraud detection. This model is divided into two phases: the training phase and the testing phase.

We presented in Chapter 2 several examples of Supplier Impersonation frauds. In order to illustrate the design of the probabilistic fraud detection system, we use the following scenario: let's define two companies C S that have previously exchanged N historical transactions $\{t_1, t_2, ..., t_N\}$. C has paid S on the same account Acc_l. A fraudster wants to attempt a fraud by impersonating S and trigger a payment from C to the account Acc_f. This triggers all future transactions $t_{N+1}, t_{N+2}, ...$ to be performed on Acc_f instead of Acc_l.

Most of the papers found in the literature that use a probabilistic model are based on Gaussian-Mixture Models (GMMs). These models represent the possible values of the features in the form of a composition of Gaussian (also known as Normal) models which parameters are the mean and the variance of the feature's distribution. However, these models only work in the case of numerical values, and not when the features are categorical values. Thus another probabilistic model needs to be found. An illustration using an urn model [JK77] can be used to help in this choice of distribution. Assume that we have an infinite urn containing all the transactions between all the suppliers in our B2B ecosystem, represented by identical balls, and each account used in at least one transaction is represented by the color of the ball. We isolate all transaction involving C and S, in a separate urn. We wish to determine the likelihood of an account Acc_i to appear when we draw the next ball from the urn, knowing that we already made $T^{C,S}$ independent draws from this urn.

There are several ways to model payments made by the client company to its supplier. If we only consider the test draw, one can use a categorical distribution. Categorical distribution is a discrete probability distribution that describes the possible result of a random variable that can take one of the multiple categories. The parameters of a categorical distribution is $\boldsymbol{p} = (p_1, p_2, ..., p_K)$ where p_i is the probability to draw the category k. However, the categorical distribution can be seen as a special case of the multinomial distribution, which models the probability of a random variable to take one of the multiple categories when challenged n times. If $n = 1$ then we have a categorical distribution. Using the multinomial rather than the categorical distribution helps use generalize our model, for example if there is need to perform batch prediction. Thus we chose to model our transactions using a multinomial distribution. This model does not need any pre-processing of the data, as the multinomial distribution is able to handle categorical data. The parameters for the multinomial distribution are $\boldsymbol{p} = (p_1, p_2, ..., p_K)$ where p_i is the probability to draw the category i, and n, the num-

ber of trials.

Let's consider a multinomial distribution $Mul(\boldsymbol{p},n)$, where $\boldsymbol{p}$ is a vector of probability. $Mul(\boldsymbol{p},n)$ can be used to model the $T^{C,S}$ transactions already observed between a client C and a supplier S. The parameter n thus becomes $T^{C,S}$. The need arise to estimate the second parameter $\boldsymbol{p}$. Using conjugate prior, as illustrated in [Min03], the unknown parameter $\boldsymbol{p}$ can be itself considered as a random variable. This means that it can be given a prior distribution in the form of a Dirichlet distribution. A Dirichlet distribution is a continuous multivariate probability distribution parameterized by a vector $\boldsymbol{\alpha}$ of positive reals. A comprehensive definition of the Dirichlet distribution can be found in ([BJ+06]). The posterior distribution of the parameter, after incorporating knowledge from the observed transaction, is also a Dirichlet. If we express this relationship in the case of a categorical distribution (for the sake of simplicity), given (with K the number of possible categories, N the number of samples) a model

$$\begin{aligned}\boldsymbol{\alpha} &= (\alpha_1,...,\alpha_K) = \text{concentration hyperparameter}\\ \boldsymbol{p}|\boldsymbol{\alpha} &= (p_1,...,p_K) \sim Dir(K,\boldsymbol{\alpha})\\ \mathbb{X}|\boldsymbol{p} &= (x_1,...,x_K) \sim Mul(K,\boldsymbol{\alpha},n)\end{aligned}$$

then [Min03] shows that

$$\boldsymbol{c} = (c_1,...c_K) = \text{number of occurrences of category } i = \sum_{j=1}^{N}[x_j = i]$$

$$\boldsymbol{p}|\mathbb{X},\boldsymbol{\alpha} \sim Dir(K,\boldsymbol{c}+\boldsymbol{\alpha}) = Dir(K,c_1+\alpha_1,...,c_K+\alpha_K)$$

Using the methods found in Bayesian statistics [Van+14], we use this relationship to estimate the parameter the posterior probability of the multinomial distribution of account values using the available set of $T^{C,S}$ historical transactions performed between C and S. A detailed overview of the process is given in Section 5.2.1.

In order to determine the legitimacy of a transaction, the process in more straightforward: we consider the account found in the transaction. If the value of $\boldsymbol{p}$ falls under a user-defined risk threshold, then the payment is deemed anomalous and considered to be a fraud.

In the remainder of this section, we first describes the process performed to determine the probabilities values of our multinomial distribution. Then explain the algorithm used in our model to assert the legitimacy of an unknown transaction is presented.

5.2.1 Model Description

In this section, the estimation of the parameter $\boldsymbol{p}$ is described. Let us denote the number of times a particular account Acc_i $i \in \{1,...,H_{Acc}^{C,S}\}$ has been seen among all the draws as n_{Acc_i}, and $\sum_1^{H_{Acc}^{C,S}} n_{Acc_i} = T^{C,S}$. We thus have a set of $T^{C,S}$ categorical variables $\mathbb{A} = a_1,...,a_{T^{C,S}}$ (the list of accounts used in every payments made by C towards S) that can be represented as a single vector-valued variable $\boldsymbol{x} = (n_i,...,n_{H_{Acc}^{C,S}})$ (the number of time each unique account was used in the list of payments), where n_i is the number of occurrences of accounts Acc_i. Let's assume that $\boldsymbol{x}$ is distributed according to a multinomial distribution $Mul(\boldsymbol{p},T^{C,S})$. The parameters of the multinomial distribution are $\boldsymbol{p} = (p_1,p_2,...,p_{H_{Acc}^{C,S}})$ where p_i is the probability to draw account Acc_i, and $H_{Acc}^{C,S}$ the number of unique accounts found in $T^{C,S}$. As detailed earlier, instead of estimating $\boldsymbol{p}$ directly, we use a conjugate prior distribution [Min03] and thus consider $\boldsymbol{p}$ as a random variable following a Dirichlet distribution with parameter vector $\boldsymbol{\alpha} = (\alpha_1,...,\alpha_{H_{Acc}^{C,S}})$. Using the formulas provided in [JKB97; Min03], the conditional

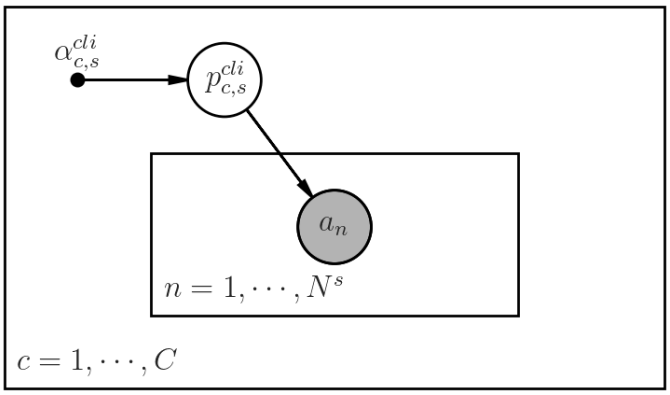

Figure 5.1: Visual representation of the naive probabilistic model in plate notation. $\boldsymbol{\alpha}_{c,s}^{cli}$ is the parameter of the Dirichlet prior on the probability distribution $p_{c,s}^{cli}$, which is itself the prior of the Multivariate distribution of accounts $a_1, ..., a_{N^s}$. The model is computed for each of the C clients found in the database.

distribution of a given variable a_i, conditioned on all the other account variables (denoted $\mathbb{A}^{(-i)}$) in $T^{C,S}$, is the following:

$$Pr(a_i = Acc_i | \mathbb{A}^{(-i)}, \boldsymbol{\alpha}) = \frac{n_{Acc_i}^{(-i)} + \alpha_i}{H_{Acc}^{C,S} + T^{C,S} - 1} \tag{5.1}$$

where $n_{Acc_i}^{(-i)}$ is the number of counts of account Acc_i in all variables other than a_i. This formulation is useful as it allows use to avoid the use of costly sampling methods such as Markov Chain Monte Carlo (MCMC) methods [Gey92]. Figure 5.1 shows an overview of the proposed Dirichlet-multinomial model in plate notation[1]. As we propose two different models in this chapter, for the sake of clarity we use $\boldsymbol{p}_{c,s}^{cli}$ and $\boldsymbol{\alpha}_{c,s}^{cli}$ to refer to the parameters of the distributions used in the naive probabilistic model.

Training Phase

While it is possible to compute Equation 5.1 every time a new transaction is submitted by the model, it is possible to reduce the computational overhead by calculating beforehand each of the possible probabilities for the unknown account. This technique is similar to the training of a model in the traditional data science, as we prepare a fixed representation of the data that will be queried later in order to assert the legitimacy of the transaction.

In order to create a model of the payment behavior of C towards S, let's define the account $a_{T^{C,S}+1}$ as the account potentially used in this new transaction between C and S. As it is a new point, Equation 5.1 can be written as follows:

$$Pr(a_{T^{C,S}+1} = Acc_i | \mathbb{A}, \boldsymbol{\alpha}) = \frac{n_{Acc_i} + \alpha_i}{H_{Acc}^{C,S} + T^{C,S}} \tag{5.2}$$

where $\mathbb{A}$ is the set of account values in the set of transactions $T^{C,S}$, and n_{Acc_i} the number of occurrences of account Acc_i in $T^{C,S}$. We use this formula to compute the probability of occurrence of each account Acc_1, Acc_2, ..., $Acc_{H^{C,S}}$.

First, let's consider the value of $\boldsymbol{\alpha}$. In our case, it consists in a pseudo-count of the occurrences of Acc_i representing our belief in the likelihood of this account. It is

[1] https://en.wikipedia.org/wiki/Plate_notation

useful in the case of a Categorical distribution, as it is updated by new evidence. In our case, however, the evidence is directly introduced by the Multinomial distribution, and thus we use a vector $\boldsymbol{\alpha} = 0$ to represent the fact that we have no prior knowledge about the accounts.

The computed prior probabilities give us an absolute chance of $a_{T^{C,S}+1}$ to take the value Acc_i. Let's consider the fact that a supplier S have been paid 300 hundred times on two accounts: 150 times on Acc_1 and 150 times on Acc_2. We thus have $Pr(a_{(T+1)^{C,S}} = Acc_1|\mathbb{A}, \boldsymbol{\alpha}) = Pr(a_{(T+1)^{C,S}} = Acc_2|\mathbb{A}, \boldsymbol{\alpha}) = 0.5$. However, it is useful to indicate that both Acc_1 and Acc_2 are very likely to occur in a transaction involving S, relatively to one another. In order to convey this fact, we normalize each posterior probability by the maximum value found. These results are no longer probability values, as their sum is greater than 1, but they represent the chance of occurrence of an account Acc_i with respect to the maximum of occurrences of any given account. In the example mentioned above, if Acc_1 and Acc_2 are both used in 150 transactions each, then their probability score would be 1.0 for each of them. Under the assumption that enough payments have been performed, it tells us that both account are as likely to be used by the client to pay the supplier, whereas if an unbalance in the accounts occurrences were to exists, the corresponding probabilities would be skewed as well (with the probability corresponding to the most occurring account still at 1.0). The main interest of this is to convert absolute values of probabilities into relative ones, and thus alleviate the need to handpick specific risk thresholds (introduced in the next subsection) corresponding to the needs of a given client, in favor for more explicit ones. This set of normalized probabilities is dubbed

$$Sc(a, \boldsymbol{\alpha}) = \frac{Pr(a = Acc_i|\mathbb{A}, \boldsymbol{\alpha})}{max(Pr(a = Acc_i|\mathbb{A}, \boldsymbol{\alpha}))}$$

for $Acc_i \in 1, \ ..., \ H_{Acc}^{C,S}$

Inference Phase

In the testing phase, a new transaction $(T+1)^{C,S}$ is tested in order to assert its legitimacy. Let's assume that $(T+1)^{C,S}$ is performed using the value a_{T+1} corresponding to account Acc_{T+1}. Two possible cases arise:

- Acc_{T+1} is an account that has been previously encountered in $\mathbb{A}$. In this case, $Sc(a_{(T+1)^{C,S}}, \boldsymbol{\alpha})$ is retrieved from the set of normalized probabilities described earlier.
- Acc_{T+1} is an account that hasn't been previously encountered in $\mathbb{A}$. In this case, $Sc(a_{(T+1)^{C,S}}, \boldsymbol{\alpha}) = 0$

This operation is a simple lookup as the inference needs to be performed as fast as possible.

Once the normalized probability score is retrieved, a threshold function is applied to the result in order to produce a green-light answer that is consistent with the ones from the expert system. $Sc(a_{T+1})$ is compared with two risk thresholds $\delta 1$ and $\delta 2$ where $0 < \delta 1 < \delta 2 < 1$. If $Sc(a_{T+1}) < \delta 1$ then $(T+1)^{C,S}$ is assigned a low legitimacy label, if $Sc(a_{T+1}) < \delta 2$ then $(T+1)^{C,S}$ is assigned a medium legitimacy label and if $Sc(a_{T+1}) > \delta 2$ then $(T+1)^{C,S}$ is assigned a high legitimacy label.

The determination of risk thresholds is a complex exercise, and in this thesis we relied on the expertise of SiS-id's fraud detection team to assign acceptable risks thresholds for the experiment. Indeed, a variation in the values of the risk thresholds has a major impact on the results, as they correspond to the level of tolerance over fraud one is willing to have. Thus, tuning and selecting appropriate risk thresholds is an important task. A cost-based analysis, that selects risk thresholds based on the monetary losses incurred by a false negative or a positive (as described by [BVV15]) might be used to select risk thresholds. However, due to the domain-specific issues found in

this task, plausible fixed risk thresholds will be used in the remainder of this chapter.

5.2.2 Experimentation Goal

In the remainder of this section, we propose an experimental evaluation of the naive probabilistic model. In the absence of a dataset containing reliably labeled fraudulent and legitimate payment, we do not have the means to assert the fraud detection performance of our system. Furthermore, as ProbaSIF is, to the best of our knowledge, the first data-driven SIF detection system, there exist no comparable system to compare performance with.

However, we do possess the subjective knowledge of SiS-id's investigators, and the results of SiS-id's platform expert system. Thus, we propose an exploratory analysis of the naive probabilistic model, using SiS-id's expert system results as a baseline. However, we can only compare the results of the two systems in an explorative way, as they both rely on different assumptions: data-driven systems such as ProbaSIF draws their labels from a data model, whereas expert systems draw their conclusion for rules defined by human experts. However, one can assume that at least part of expert knowledge is constructed from experience, which means the study of past data.

Thus, the first goal of this experiment is to provide a qualitative overview of the differences between the results of a data-driven system and an expert system. This result allows highlighting differences and similarity between the two kinds of systems.

The second goal of this experimentation is to assert the operational efficiency of our system, which is the computational time it takes to compute the probability scores composing the model, along with the inference time needed to assign a label or a score to a suspicious transaction.

5.2.3 Experimental Setup

ProbaSIF has been run on a laptop running Ubuntu 18.04.4 with an Intel Core i7-10510U CPU and a 15,5 GiB memory in order to evaluate its performance. The data used to train the model is the History dataset previously described in Chapter 4 Section 2, consisting of a set of unlabeled transactions between client companies and supplier companies. The data used to test our model is the Audit dataset described in Chapter 4 Section 3.

In order to perform our experimentation, first a case study of a single client company is performed. The goal of limiting the evaluation to a representative client is to be able to investigate in detail the differences between the expert system and the data-driven system. Focusing the experiment on a single client allows us to obtain the granularity needed for this comparison. Thus a client that possesses 251 transactions in the Audit dataset have been selected. The reason to choose this client is that it showed a manageable number of transactions analyzed by SiS-id's expert system, and that it possesses a high number of transactions in the History dataset. These transactions are used to train ProbaSIF, and then the transactions from the Audit dataset are used to compare the results of the rule-based system with the one found by the naive probabilistic model.

In this experiment, we use 2 risk thresholds in order to create 3 distinct labels, so that the results found by the naive probabilistic model are comparable to SiS-id's expert system, described in Chapter 4 . As mentioned earlier, the discretized results (in the form of high, medium and low legitimacy labels) of the naive probabilistic systems relies heavily on the values of the thresholds $\delta 1$ and $\delta 2$. Indeed, a small value of $\delta 1$ leads to fewer transactions labeled with a low legitimacy label, whereas large value of $\delta 2$ leads to almost no transaction labeled with the high legitimacy label.

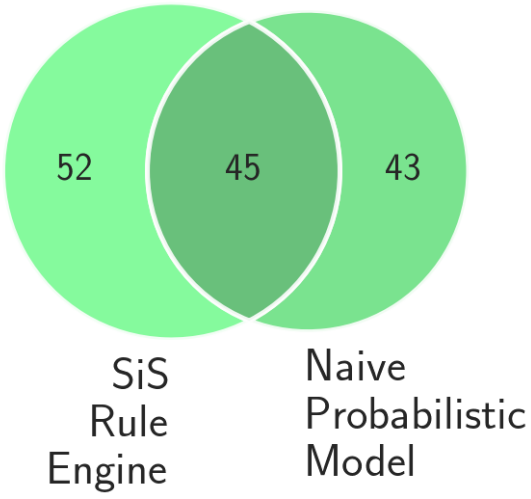

Figure 5.2: Distribution of "high" legitimacy labels - Expert system & Naive probabilistic model.

These risks thresholds have been set to $\delta 1 = 0.5$ and $\delta 2 = 0.9$. These risks thresholds have been determined after discussion with SiS-id's fraud investigation team so that they reflect the behavior and risks taken by the rule engine: $\delta 1 = 0.5$ means that every payment issued from a client to an account of the supplier seen less than half the time by the client is deemed anomalous, whereas $\delta 2 = 0.9$ means that only an account used in the client's payment have been seen at least 90% of the time is considered as legitimate. This configuration should translate into a system that does not assign a high legitimacy score to many transactions, and is severe concerning unknown transactions. The emphasis of the presented work being on the creation of unsupervised model for fraud detection than the determination of correct thresholds for optimal gains, we fix these values. Devising a comprehensive function that maps categorical fraud labels to anomaly values is an interesting topic for future work, but not covered in this thesis.

5.2.4 Results & Discussion

In this section we discuss the results of our experiment and discuss our findings.

High Legitimacy Labels

Figure 5.2 shows the distribution of high legitimacy labels given by the naive probabilistic model and SiS-id's expert system. The results of the two anomaly detection systems are not very consistent, as less than half (45 out of the 97) of the transactions labeled with the high legitimacy label by the expert system have also been given a high legitimacy label by the naive probabilistic model. Table 5.1 shows that out of the 52 diverging labels, the naive probabilistic model attributed 51 low legitimacy labels to the transaction, thus considering them as anomalous. This discrepancy might be caused by the fact that only the knowledge of a single client is used to perform the anomaly detection using the naive probabilistic model. This issue is addressed in the next section. Table 5.1 also shows that the result that were attributed a "high" legitimacy label by the considered system were almost never (4 out of 88) attributed a "low" legitimacy label by the expert system: this means that the expert system does not directly contradict most of the transaction allowed by the considered system, and thus that both systems agrees on the same meaning of "high legitimacy" for these transactions.

Medium Legitimacy Labels

Figure 5.3 shows the distribution of medium legitimacy labels given by the naive probabilistic model and SiS-id's expert system. This label has different meanings for both

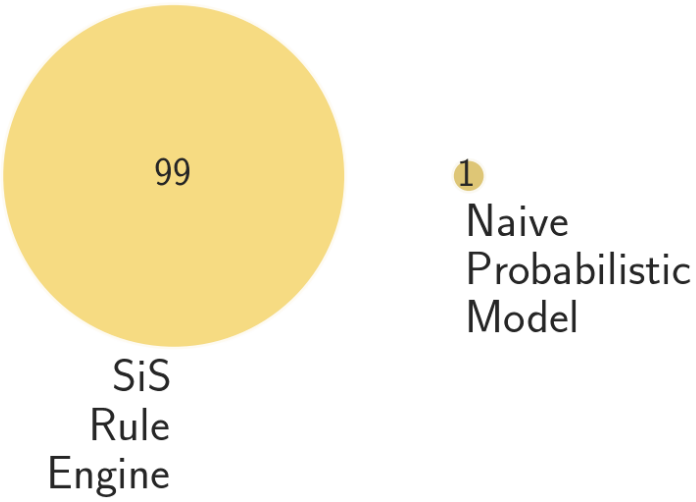

Figure 5.3: Distribution of "medium" legitimacy labels - Expert system & Naive probabilistic model.

Expert System → Naive probabilistic model ↓	High	Medium	Low
High	**45**	39	4
Medium	1	**0**	0
Low	51	60	**51**

Table 5.1: Confusion Matrix - Expert System and Naive Probabilistic Model.

systems: in the case of the expert system, it simply means that the system cannot make an assertion regarding the legitimacy of the transactions because it lacks information. In the case of the naive probabilistic model, it means that the probability of using the account is between the two risk thresholds.

The results of the two anomaly detection systems are thus widely inconsistent. However, the naive probabilistic model shows a tendency towards more assertive than the expert system as only 1 transactions is given an orange label, compared with the 99 transactions who were assigned this label by the expert system. This result highlights the fact that the considered system can provide answers where the expert system does not have enough knowledge to propose a conclusive answer. However this assertiveness highly relies on the risk thresholds determined by the user.

Low Legitimacy Labels

Figure 5.4 shows the distribution of low legitimacy labels given by both SiS-id's expert system and the naive probabilistic model. The results show that most (51 out of 55) of the transactions labeled as not legitimate by the expert system have also been deemed not legitimate by the naive probabilistic model. However, 111 transactions labeled with the "low" legitimacy label by the naive probabilistic model were given other labels by the expert system. Table 5.1 shows that these labeled mostly come from the medium and high legitimacy labels sets previously described.

Operational Efficiency

Finally, Table 5.2 shows the performance of the expert system and the naive probabilistic model in terms of execution time. In order to be consistent with the real use case of SiS-id, we create models for every client found in both the History dataset and Audit dataset, and compute the time taken for the creation of every model. While

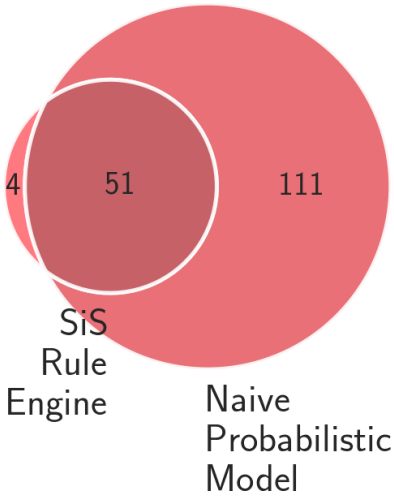

Figure 5.4: Distribution of "low" legitimacy labels - Expert system & Naive probabilistic model.

	Expert System	Naive Probabilistic Model
Training time (62 clients) (s)	-	2
Training time (1 client) (s)	-	0.032
Inference time (s)	3	0.002

Table 5.2: Operational Efficiency - Expert System and Naive Probabilistic Model.

the naive probabilistic model takes 2 seconds to train on the data from the History Dataset, it is not possible to compare it with the expert system. Indeed expert systems use built-in rules, either in the form of hard-coded algorithm or more manageable rule engines. However, the rule engine underlying the expert system has to be regularly updated in order to address evolution in a company's behavior. This update is a very time-consuming task as it requires extensive human input to be performed.

In order to consider the inference time, a focus is made on the inference made by the selected client. When considering the time taken to handle a transaction's labeling, the expert system's performance (3 seconds for one transaction) is very slow. This is explained by the fact that the pre-processing phase of the expert system process relies on call to external APIs to be performed, and thus the answers have to be received by the expert system in order to compute its answer. It is very slow compared to the naive probabilistic performance (0.57 seconds for all 251 transactions, corresponding to 0.002 seconds for one transaction), as the proposed system does not use such external APIs and rely solely on the available data.

5.3 Collaborative Probabilistic Model

While the Naive probabilistic model is suitable to model the narrow view of a client concerning its own transactions with a supplier, it possesses an inherent flaw: it does not take into account the relationship the supplier has with other clients of the ecosystem.

Consider the following scenario: A client C_1 emits payment regularly to a supplier S_1 using the account a_1. S_1 also receives regular, legitimate payments for a client C_2 on the account a_2. Using the naive probabilistic model, if S_1 decides to be paid by C_1 on its account a_2 then the probability α_{a2} is 0 and thus the transaction is considered

as anomalous, and thus potentially fraudulent. This type of error where a legitimate payment is deemed fraudulent is known as "false negative" (w.r.t the legitimacy of the transaction. As a reminder, legitimate transactions are considered "positives" while fraudulent transactions are considered "negative" in the thesis, in order to be consistent with the expert system).

Thus, it is necessary to consider the knowledge of all the clients interacting with a specific supplier, rather than focus on a single point of view, in order to reduce the number of false negatives of the system. The following model proposes an approach to integrate this collaborative knowledge to the anomaly detection system.

5.3.1 Model Description

The model used for the collaborative system is the same as the one describes in the naive system, with the notable difference that rather than use only the subset $T^{C,S}$ of the History dataset containing the transaction involving only company C and company S, the subset T^S containing all the transactions involving S in the History dataset is used. The computation remains the same, so in the end we use the posterior distribution

$$Pr(a_{T^S+1} = Acc_i|\mathbb{A},\boldsymbol{\alpha}) = \frac{n_{Acc_i} + \alpha_i}{H^S_{Acc} + T^S} \tag{5.3}$$

with H^S_{Acc} the set of accounts used to pay supplier S by at least one client in the hisory dataset, and the score function

$$Sc(a,\alpha) = \frac{Pr(a = Acc_i|\mathbb{A},\alpha)}{max(Pr(a = Acc_i|\mathbb{A},\alpha))}$$

for $Acc_i \in 1,\ ...,\ H^S_{Acc}$ in order to pre-compute the anomaly score of a transaction.

Inference Phase

The inference phase remains the same as the one described in the previous section, with the difference that the score is retrieved from $Sc(a_{(T+1)^S},\boldsymbol{\alpha})$ rather than $Sc(a_{(T+1)^{C,S}},\boldsymbol{\alpha})$.

5.3.2 Experimentation Goal

In the remainder of this section, we propose an experimental evaluation of the collaborative probabilistic model. The same comment on the absence of a dataset containing reliably labeled fraudulent and legitimate payment applies. In this experiment we wish to provide an explorative overview of the collaborative probabilistic model's results, in a similar fashion as the previous experiment with the naive probabilistic model.

Finally, in this experimentation, we aim to validate that the hypothesis of including the transaction from several clients instead of a single one indeed leads to an increase in the number of transaction considered legitimate by ProbaSIF. In order to do so, we use the results provided by the expert system as a baseline to compare two probabilistic models.

5.3.3 Experimental Setup

The experimental setup is the same as the one used in the previous section.

5.3.4 Results & Discussion

In this section we present the results of the experiment and discuss our findings.

High Legitimacy Labels

Figure 5.5 shows the distribution of high legitimacy labels given by the collaborative probabilistic model and SiS-id's expert system. The number of similarly labeled trans-

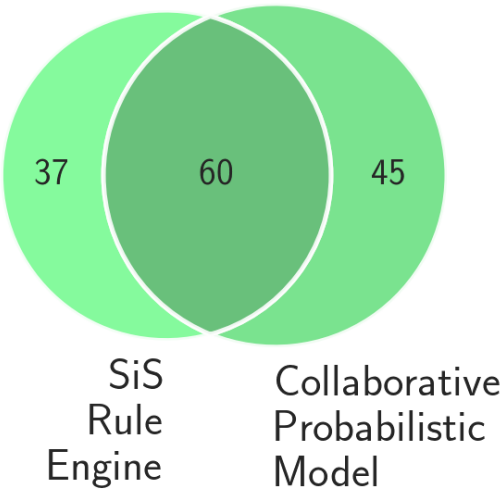

Figure 5.5: Distribution of "high" legitimacy labels - Expert system & collaborative probabilistic model.

actions shows an increase, from 45 to 60. This confirms the hypothesis that adding the knowledge of more client companies is consistent with the expert knowledge used by fraud investigators, and allows to reduce the number of legitimate transactions labeled as fraudulent. However, 37 out of 97 transactions are still labeled differently. Table 5.3 shows that out of the 37 diverging labels, the naive probabilistic model attributed 35 low legitimacy labels to the transaction. This discrepancy might come from the fact that SiS-id's expert system uses knowledge from the underlying fraud detection platform, in the form registered legitimate accounts. This knowledge is not yet available for our models and thus might be the cause of differences in the attribution of high legitimacy labels.

Medium Legitimacy Labels

Figure 5.6 shows the distribution of medium legitimacy labels given by the collaborative probabilistic model and SiS-id's expert system. The two systems gave these labels to different transactions. However, as it was the case with the naive probabilistic model, the collaborative probabilistic model also seems more assertive than the expert system. Furthermore, adding the knowledge of more clients seems to introduce a higher diversity in the probabilities as 5 transactions are given the medium legitimacy label using the collaborative probabilistic model, whereas only 1 was labeled as such by the naive probabilistic model. However, the fact that both naive and collaborative probabilistic models do not yield a high amount of medium legitimacy labels when analyzing transactions doesn't allow for significant conclusions to be drawn.

Low Legitimacy Labels

Figure 5.7 shows the distribution of low legitimacy labels given by both SiS-id's expert system and the collaborative probabilistic model. As expected, there is a decrease in transactions labeled with the low legitimacy labels by the collaborative probabilistic model. Most of the transaction (46 out of 55) still shows a high consistency with the expert model. However, an increase of discrepancy occurs as 2 transactions previously labeled with the low legitimacy label by both the expert system and the naive probabilistic system are labeled with the high legitimacy label by the collaborative probabilistic model (4 to 6 in the upper line, rightmost columen in Table 5.1 and Table 5.3). Raising the ambiguity surrounding this kind of mislabeling is only possible through the use of a dataset containing proven legitimate transactions and fraudulent transactions.

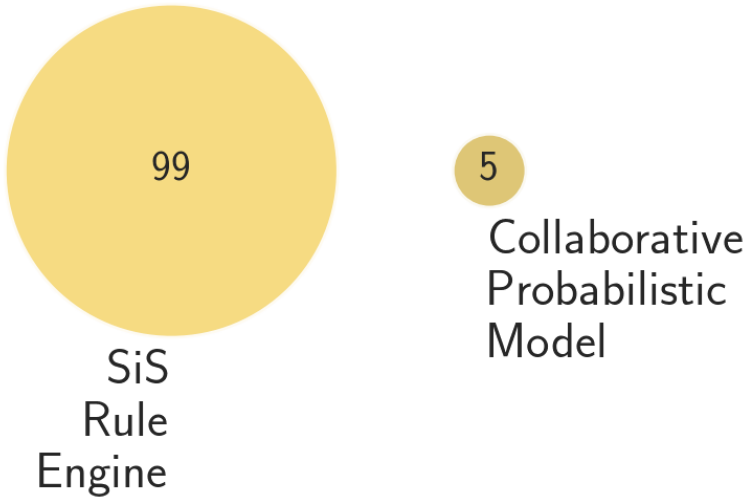

Figure 5.6: Distribution of "medium" legitimacy labels - Expert system & collaborative probabilistic model.

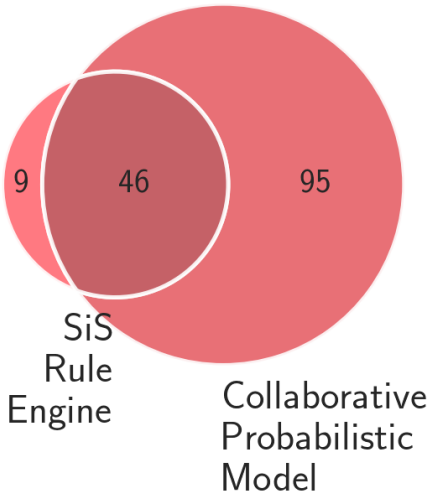

Figure 5.7: Distribution of "low" legitimacy labels - Expert system & Naive probabilistic model.

Expert System → Collaborative Probabilistic Model ↓	High	Medium	Low
High	**60**	39	6
Medium	2	**0**	3
Low	35	60	**46**

Table 5.3: Confusion Matrix - Expert System and Collaborative Probabilistic Model.

	Expert System	Naive Probabilistic Model
Training time (62 clients) (s)	-	2
Training time (1 client) (s)	-	0.032
Inference time (s)	3	0.002

Table 5.4: Operational Efficiency - Expert System and Collaborative Probabilistic Model.

Operational Efficiency

Finally, Table 5.4 shows the performance of the expert system and the collaborative probabilistic model in terms of execution time. The collaborative probabilistic model takes the same amount of time to train (2 seconds for all clients, 0.032 sec each) than the naive probabilistic model. This is due to the fact that while the number of transaction increases when using the collaborative model, the model computation is a simple sum and thus the computation overhead is not significant.

Similarly, the time taken to handle a transaction's labeling, is also the same (0.57 seconds for 251 transactions, 0.002 seconds for one transaction). This illustrates the improvement in operational efficiency of both models.

5.4 Global Results

In this section, we discuss the performance of both the naive probabilistic model and the collaborative probabilistic model when considering the 62 clients of the Audit dataset that possess at least 100 transactions in the History dataset. This analysis allows us to assert both the operational efficiency of both models, along with its global consistency with the results of SiS-id's expert system.

In order to conduct this analysis, we focus on two metrics: the Low Legitimacy Consistency (LLC) score, defined as the number of transactions given a low legitimacy label by both ProbaSIF and the expert system, divided by the number of transactions given a low legitimacy label by the expert system. This metrics allows us to compare the behavior of the two systems regarding the potentially fraudulent cases.

Thus, the second metric used is High Legitimacy Consistency (HLC) metric, defined as the number of transactions given a high legitimacy label by both ProbaSIF and the expert system, divided by the number of transactions given a high legitimacy label by the expert system. This metric is used to assert if ProbaSIF is consistent with the expert system w.r.t legitimate transactions.

Naive Probabilistic Model

In this subsection, we analyze the performance of ProbaSIF when using the naive probabilistic model.

Figure 5.8 shows the Low Label Consistency score for each unique client found in the Audit dataset that contains a set of transactions already evaluated by SiS-id's expert system. We can see that for most of these clients, the LLC of the naive probabilistic model is fairly high, as only 11 out of the 62 considered clients shows a LLC score

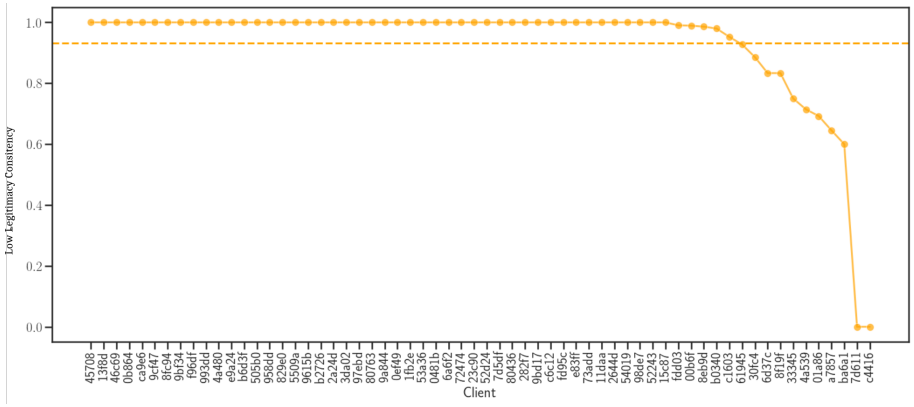

Figure 5.8: Low legitimacy consistency score of the naive probabilistic model for the client of the Audit dataset.

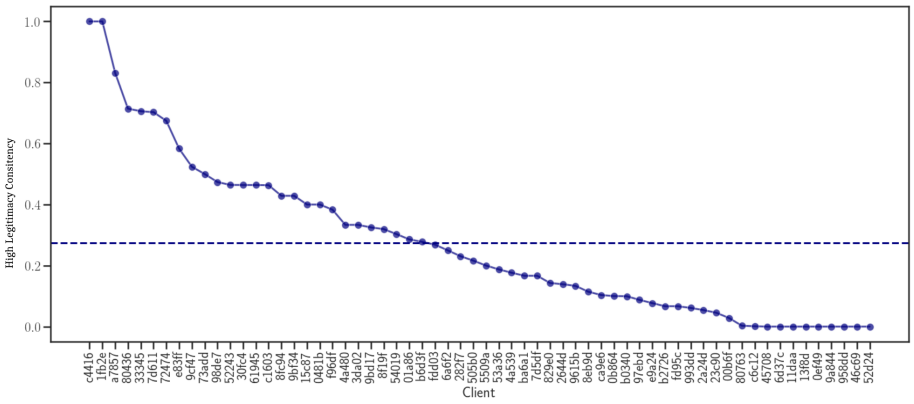

Figure 5.9: High legitimacy consistency of the naive probabilistic model for the client of the Audit dataset.

below the average of 0.932. This means that, when considering the low legitimacy label, the consistency previously witnessed for a single client seems to generalize across the considered dataset. However, 2 of the clients shows a LLC of 0, meaning that the naive probabilistic model failed to attribute a low legitimacy label when at least one transaction was deemed suspicious by the expert system. A thorough analysis of these cases might yield useful insight on the limit of the naive probabilistic model.

Another limit of the naive probabilistic model is illustrated by Figure 5.9, that shows the HLC comparing the transaction given a high legitimacy label by the rule engine and the naive probabilistic model. Most (36 out of 62) of the considered clients fall below the mean of HLC of 0.274. This means that the discrepancy in the attribution of high legitimacy label is frequent throughout the dataset, and that there is room for improvement for this metric if the goal is to improve the consistency between the two models.

Collaborative Probabilistic Model

In this subsection, we analyze the performance of ProbaSIF when using the collaborative probabilistic model.

Figure 5.10 shows the LLC score for each unique client of the Audit dataset. The global LLC mean is reduced from 0.932 to 0.827, thus indicating that using the collaborative probabilistic model alters the capacity of ProbaSIF to assign low legitimacy labels. Furthermore, now 4 of the clients shows a LLC of 0.

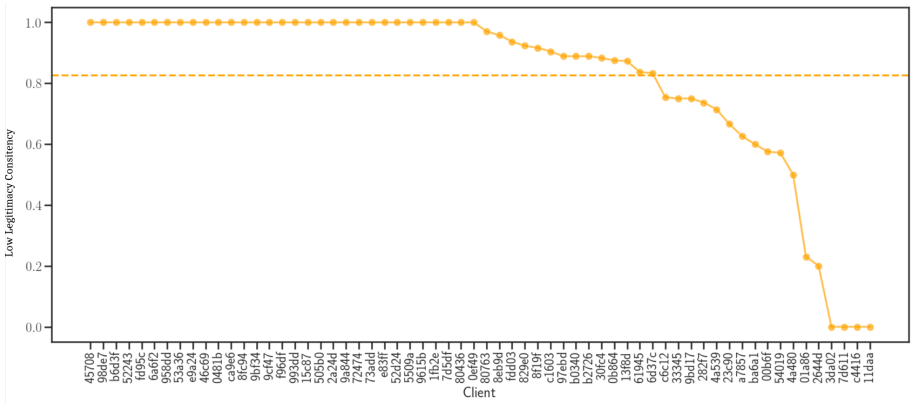

Figure 5.10: Low legitimacy consistency of the collaborative probabilistic model for the client of the Audit dataset.

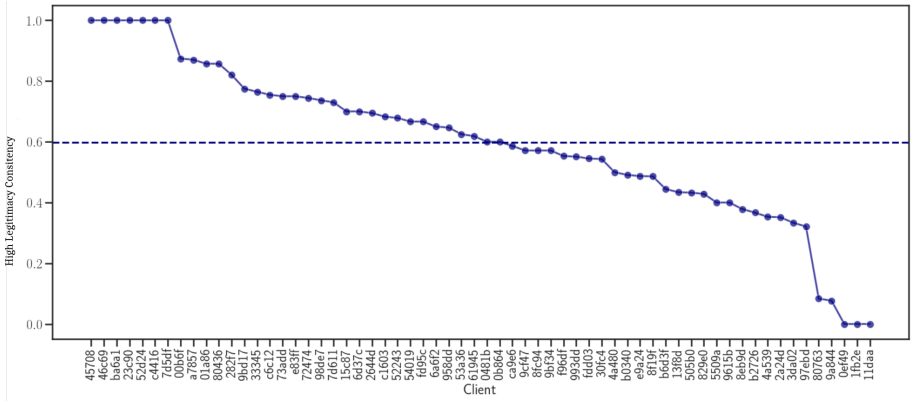

Figure 5.11: High legitimacy consistency of the collaborative probabilistic model for the client of the Audit dataset.

However, this slight decrease in LLC is compensated by the improve in HLC shown by the collaborative probabilistic model. Figure 5.11 shows that the mean HLC goes from 0.274 to 0.598, thus illustrating the improvement in consistency induced by using such a model. Thus, using the naive probabilistic model or the collaborative model depends on the cost of a false positive or a false negative. [BVV15] argues that, in the context of fraud, a false negative (a fraudulent transaction labeled as legitimate) is more costly than a false positive. While the cost of misclassification has to be discussed in the context of SIF, we tend to agree with this statement, as unseen costs such as loss of trust in the supplier and the detection system are tied to a false negative, whereas only costs in terms of investigations are incurred in case of false positive. While these costs might be prohibitive in the case of credit card frauds (high number of payments with small amounts, whose legitimacy are hard to verify), this is not the case for SIF (relatively small number of payments with high amounts).

5.5 Summary

In this chapter, we introduced a SIF detection system called ProbaSIF based on Bayesian Inference that uses the transactions of the History Dataset to compute for each agent the probability to use a specific account to perform a transaction with a supplier, either given a specific client history, or globally. A label is assigned to an unknown transaction based on the probability for the account involved in the transaction to be used to pay the supplier. This work is intended as a first attempt to use purely data-driven sys-

tem in order to detect fraud, and thus focus on the elaboration of a probabilistic model.

We described the two main algorithms composing ProbaSIF, each linked to a specific probability distribution fit by the transaction found in the History Dataset. Firstly, the intra-company analysis focuses on the probability of an account to be used to pay a supplier, knowing that a specific client performed the transaction. Secondly, the inter-company analysis does not take into account the client issuing the transaction and focuses on the accounts used to pay the supplier. This approach aims to model a more general view the supplier's behavior to detect a possible discrepancy. We showed that for each new transaction a probability of usage was computed and from it a legitimacy label was computed using user-defined thresholds. These thresholds were defined using expert knowledge. Their determination and optimization are required in order to further assert the performance of ProbaSIF, but this task strays from the topic of modeling and into the territory of cost analysis, and thus have not been pursued in this thesis. Furthermore, without a dataset containing real information about legitimate and fraudulent transaction, the determination of the optimal thresholds is a difficult task.

We presented the result of ProbaSIF first on a single client to investigate its performance locally, and we then generalized it to the other clients of our B2B ecosystem in order to investigate its global performances.

Results show that locally, ProbaSIF results differs from the expert knowledge w.r.t "medium" legitimacy payments and "high" legitimacy payments. When considering the "medium' legitmacy payments, this discrepancy shows that ProbaSIF is able to disambiguate transactions where the expert system is unable to provide a conclusive answer. Additionally, ProbaSIF seems to be biased towards "low" legitimacy transactions, assigning this label to a significant number of transaction in the use case presented. While this lead to ProbaSIF having a high "recall" w.r.t the results of the expert system, it is at odds with the distribution of fraud in other domains of fraud detection (0.4 % in credit card fraud dataset, as seen in [Luc+19]). Further research is needed, especially using a dataset containing real legitimate and fraudulent transaction, in order to assert the real capacities of ProbaSIF. However, ProbaSIF shows that sharing knowledge between companies have a significant impact on the result of a data-driven model.

6

Supplier Impersonation Fraud Detection using Transaction Networks and Self-Organizing Maps

In this chapter, we present GraphSIF, a Supplier Impersonation Fraud (SIF) detection system that uses a dataset of Business-to-Business (B2B) payments issued between companies and their suppliers to construct a relationship network. We use this network to model the interaction between companies, and to detect anomalous payments that are likely to be fraudulent.

We use the past payments to create a time-evolving behavior sequence summing up the evolution of the graph through time. We then compare the new graph created by adding a suspicious transaction to the behavior sequence and investigate the potential discrepancy it introduces. If this discrepancy is low then the transaction is considered as legitimate, and if the discrepancy is high then the transaction is considered as likely fraudulent. In order to quantify this discrepancy, a Self-Organizing Map (SOM) is trained on the behavior sequence, and a clustering algorithm is used to quantify the similarity of the tested graph with the ones in the behavior sequence.

The contribution of this chapter are the following: a graph-based feature engineering process relying on a bipartite graph constructed from transactions in a B2B context, a classification system that uses Self-Organizing Maps and K-means clustering to investigate the legitimacy of a new transaction, and an comparison of the proposed anomaly detection system with the results of SiS-id's expert system, using data from a real-life B2B ecosystem.

The remaining of the chapter is structured as follows: we first describe in detail the feature engineering process used to compute the graph used by the SIF detection system. We then describe the classification system we use to label unknown transaction. Finally we evaluate the results of our SIF detection system.

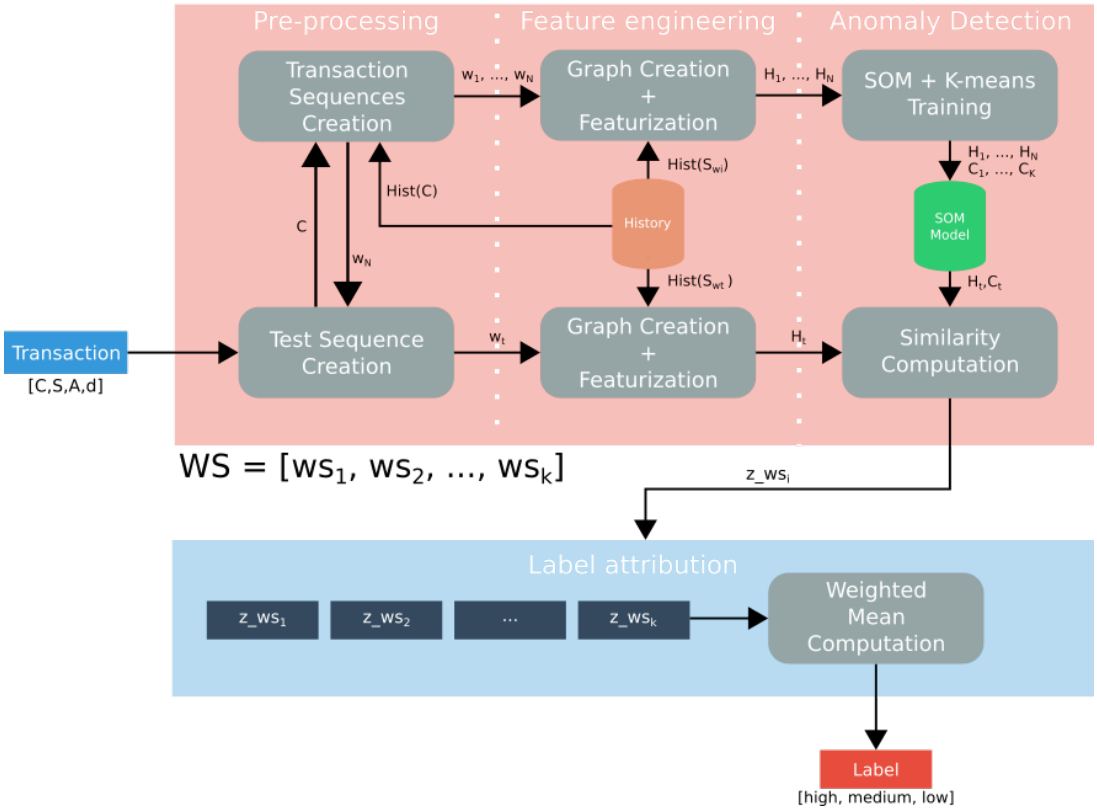

Figure 6.1: GraphSIF overview

6.1 GraphSIF Overview

GraphSIF is a SIF detection system based on anomaly detection that uses the relationship created between the companies interacting in the studied B2B environment in order to determine the legitimacy of a transaction, given the companies that issued the payment.

This system is composed of four phases:

1. A pre-processing phase, where historical transactions are sorted according to their time-frame and the companies that issued them, and grouped in time windows in order to describe a time-evolving sequence composed of several fixed-length windows of transactions, describing the behavior of a specific client. This phase is described in Section 6.2.
2. A feature engineering phase where each of the windows of transactions is transformed into a graph. This graph sums up the interactions that occurs between the client and its supplier during the specified windows. Section 6.3 describe this phase.
3. An anomaly detection phase where a specific transaction (the "suspicious" transaction) is added to the most recent graph, creating a "test graph". This graph's similarity with the ones occurring in the historical sequence is computed. Section 6.4 provides more details about this phase.
4. A label attribution phase where the similarity of the test graph for a set of different sizes of the windows of transactions are aggregated and a legitimacy label derived from the aggregation score. More details about this phase are found in Section 6.5.

Figure 6.2 shows an overview of the process. A transaction t involving the client C, the supplier S and the account A at a date d is given as input to GraphSIF. For a set of window sizes $ws_1, ws_2, ..., ws_k$, the following process is repeated:

First, the identifier of C is used to gather all transactions involving C (T^C) from the History dataset that contains the list of all the transactions occurring between all the companies of the B2B ecosystem. Then, this list of transactions is ordered by date of occurrence and split in N fixed-size windows of size ws_i where $i \in 1,...,k$. The sequence of transactions (dubbed "behavior sequence") is then sent to the next phase. In the same time, the oldest transaction of the most recent window of the sequence (w_N) is removed and t is added to it as its most recent transaction, creating the "test window" w_t. This step allows us to isolate the transaction relevant to the specific client and suppliers potentially victims of the supplier impersonation fraud.

In the next step, the transactions found in each window $w_1,...,w_N$ and w_t are used to create a graph that represents the relationships between C and all of its suppliers. Each window is complemented by a set of transactions from the History dataset $T^{S_{wi}}$, where S_{wi} is the set of supplier involved with C during w_i found in the History Dataset. Similarly, $T^{S_{wt}}$ is used to create the test graph corresponding to w_t. The graphs are then transformed into an histogram that uses the links created between the accounts paid by C and used by its supplier as characterizing features of the graphs. Each payment create a new connection between a supplier node and an account node, or update an existing one. These edges can then be used to define a client C's "payment network" and infer if the use of an account is normal or abnormal. Each of the graphs is converted into its corresponding histogram H_i, thus creating the sequence $H_1,...,H_N$ and H_t the histogram corresponding to the test graph. This step allows us to transform a sequence of transaction into a feature vector representing the relationship between its suppliers, taking into account the historical behavior of the client.

These histograms are then used to assert the similarity of H_t with the histograms of the behavior sequence $H_1,...,H_N$. First, the histograms of the behavior sequences are clustered using the clustering algorithm K-Means ([Jai10]), and then are used to train a Self-Organizing Map (SOM) ([BXD06]). The k centroïds $C_1,...,C_K$ are also located on the SOM. Then, H_t is assigned a cluster using the previously used K-Means algorithm, and its similarity with the other members of the cluster is computed using the z-score metric. This anomaly detection step allows the system to distinguish usual relationship and unusual ones, that are more likely to be fraud attempts.

The z-score computation is repeated for the set of window sizes $ws_1,...,ws_k$, and they are aggregated using a threshold function and a weighted mean. This step is needed to consider the different granularity of each windows, and to produce a label that synthesize all the knowledge provided by the previous analysis.

In the remainder of the chapter, we first describe the different algorithms used at each phase of the system, and discuss the underlying motives behind their design. We then provide an experimental evaluation of the system using the labeled transactions found in the Audit dataset that contains the results of the expert system designed by SiS-id.

6.2 Transactions Pre-Processing

This section details the first phase of the system, where the transactions from the History dataset are pre-processed in order to create local behavior profiles. The goal of this pre-processing is to partition the transaction emitted by a client C in order to detect repeatability in its payments.

6.2.1 Company Local Profile

In this phase of the fraud detection system, all the transactions involving the client C found in the tested transaction $t = [C,A,S,d]$ are gathered from the History dataset.

The History dataset contains all the N_H transactions made by all clients in the studied B2B environment. By isolating only the transaction issued by C, we create a local profile of C. As shown by [BH+01], the use of local profile allows to detect anomalous behavior that would have been deemed legitimate when using a global profile. This local profile is dubbed T^C.

6.2.2 Behavior Sequence

As companies evolve and thus interact differently with suppliers or clients, thus the transactions they issue or receive change as time passes. In order to quantify this evolution, we first order all the transactions in Hist(C) temporally. This gives us an overview of C's interaction with its supplier through time. Then, in order to characterize this behavior, we partition Hist(C) into sets of transactions of size ws, that we call "windows". Using fixed-length windows in order to describe the behavior of a system is a well-known techniques, and has been successfully used in fraud detection systems such as ([QX07]) and ([Mon+13]).

6.2.3 Window creation

In order to create the windows, two kinds of partitioning are possible: by transaction date (from July 1st to August 1st for example), or by transaction rank (10th transaction to 5th transaction, 5th transaction to 1st transaction...). A major issue with partitioning by date is that there is no guarantee that the transactions will be homogeneously divided into the different windows. In the most extreme case, all transactions might occur in a single window, and all the other windows are rendered useless for the system. Thus, creating windows by transaction order allows us to ensure that an equal number of transactions will be found in each windows. This is a strong design decision for the system, as several temporal metrics are lost this way (for example, the number of payments in a given time window can be in itself a feature for fraud detection). Similarly, using the transaction rank instead of transaction date implies a somewhat continuous distribution of payment, as otherwise transactions from very different time period might be aggregated in a single time window.

The number of windows N created from Hist(C) is inversely proportional with the size ws of the windows: $N = \frac{N_H}{ws}$. In the case when N_H is not a multiple of ws, the $N_H \% ws$ oldest transactions in $Hist(C)$ are discarded, where % is the modulo operation. Indeed, the oldest transactions are the less prone to inform us of a fraud in the present.

The size ws of the windows has a major impact on the system. A small window size creates more data points for the system to analyze, at the cost of a decreased variability in the possible payment behavior, and thus a less detailed view of C's behavior. Inversely, a larger window allows for more detailed view of the system, but less input will be provided to the system. Additionally, adding more transaction might add to much noise to the window and thus obfuscate meaningful patterns. Therefore a careful trade-off has to be found for ws. We propose a way to solve this issue in 6.5.

6.2.4 Test Window

In order to assert the legitimacy of a singular transaction, we use the previously partitioned sequence as a basis. The key hypothesis is that a legitimate transaction will not significantly disturb the payment behavior of C, while a fraudulent transaction will be different from the previously recorded behavior. The transaction to be tested is added to the most recent windows of the sequence, whose oldest transaction has been removed, thus simulating the occurrence of the new transaction as the next step in the sequence. Indeed, if only a single transaction was tested against the partitioned

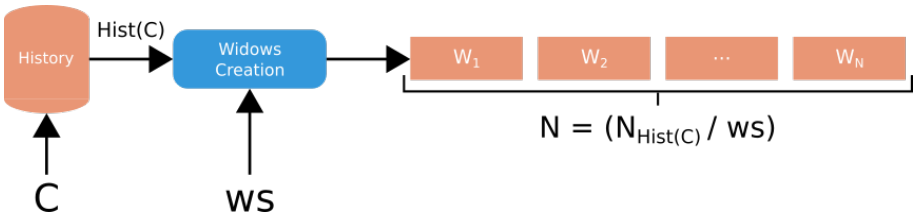

Figure 6.2: Behavior sequence creation

sequence, an anomaly would always be found due to the discrepancy in the number of transactions.

In the next section we detail how the relational data found in partitioned sequence (called "behavior sequence") and the test window is extracted and used to assert the legitimacy of the tested transaction.

6.3 Graph-based Feature Engineering

In this phase, each of the windows created in the pre-processing phase is transformed into a graph. This graph, called "transaction graph", allows the representation of the relationships between companies as a mathematical structure. The graph is composed of companies and accounts represented as vertices (also called "nodes"), while the flow of money between companies creates the edges of the graph. However, in order to use the graphs derived from the behavior sequence as a basis for the anomaly detection system, they need to be converted in a structured data form (commonly referred to as "feature vector") in order to be used. This type of transformation is known as "graph embedding" ([GF18]). We propose a tailored graph embedding approach in order to express a transaction graph as a feature vector called "graph histogram". This approach exploits several properties of the transaction graph in order to construct the embedding. The graph histograms corresponding to the behavior sequence are then used to train our anomaly detection system, while the graph histogram corresponding to the test windows is investigated.

6.3.1 Transaction Graph Creation

In this subsection, we describe the process of creating a transaction graph from a set T of transactions (as a reminder, a transaction t has the following structure: $t = [C_t, A_t, S_t, d_t]$ where C_t is the client issuing the transaction, A_t is the account receiving money from C_t, S_t is the supplier receiving the transaction, and d_t the date when the transaction takes place). A graph $G =< \boldsymbol{V}, \boldsymbol{E} >$ is composed of two sets: a set of vertices $\boldsymbol{V}$ that represents the entity of the targeted system, and a set $\boldsymbol{E}$ of edges that represents the relationship of the entities, and $e \in \boldsymbol{E} =< n_1, n_2 >$ with $n_1, n_2 \in \boldsymbol{V^2}$. Note that a label can be added in both nodes and edges of the graph, in order to add additional information. In the current implementation of the system, we use vertice labels in order to distinguish between company nodes and account nodes. No edge label is used in this implementation. Using the number of transactions, and the time from last transaction implying the two connected nodes, have been considered theoretically but due to lack of time have not been experimentally tested.

Algorithm 3 shows the process that creates a transaction graph $G_{C,T}$ from a set of transaction $\boldsymbol{T}$. An example of output of Algorithm 3 is given in Figure 6.3. The algorithm takes as input a client C and parses $\boldsymbol{T}$ in order to map all of the accounts and suppliers involved in a transaction with C as vertices. For each transaction, two edges are created: one that link C and the account involved in the transaction, and one that link the account with the supplier receiving payment for the transaction. If a vertice representing an account or a supplier is already in the vertice set, it is not duplicated.

Algorithm 3: Transaction Graph Creation

Inputs :
- C: Identifier of a Client company
- $\boldsymbol{T}$: Set of transactions

Outputs :
- $\boldsymbol{V}$: Set of vertices of $G_{C,\boldsymbol{T}}$
- $\boldsymbol{E}$: Set of edges of $G_{C,\boldsymbol{T}}$

Variables : $\boldsymbol{V_c} = C$, $\boldsymbol{V_s} = \varnothing$, $\boldsymbol{V_a} = \varnothing$, $\boldsymbol{E} = \varnothing$

```
foreach t = [C_t, A_t, S_t, d_t] in T do
    if C_t = C then
        V_a.insert(A_t);
        V_s.insert(S_t);
        E.insert((C, A_t));
        E.insert((A_t, S_t));
        T.remove(t);
foreach r = [C_r, A_r, S_r, d_r] in T do
    if S_r in V_s then
        V_a.insert(A_r);
        E.insert((A_r, S_r));
N = V_c ∪ V_s ∪ V_a
```

Similarly, since edges already in the edge set are not duplicated, an occurrence metric is updated in both case in order to prevent the loss of information.

If the algorithm stops at this point, only the payment information related to C is used. This means that if an account is used to pay a supplier S by a client different than C, it will not appear on the graph. In order to add the information provided by other clients, the accounts they use to pay S are appended to the graph, along with edges that link them to S. This addition allows us to make use of the collaborative knowledge of the other clients.

6.3.2 Properties of a Transaction Graph

A transaction graph $G_{C,T}$ shows interesting properties. It is a directed graph, as the edges represent the movement of fund from a client to a supplier through an account. A transaction graph is also a *bipartite graph*. A bipartite graph is defined in ([BVV15]) as a graph whose vertices can be divided into two disjoint and independent sets u and v and such that every edge connects a vertex in u to one in v. The transaction graph satisfies this property as the created edges are only from company to account and from account to company (no account-to-account or company-to-company edges exist in the graph). Table 6.1 shows the representation of the transaction graph T1 (shown in 6.3) as a connectivity matrix: each of the row corresponds to a company vertice, while a column represents an account vertice. The value in the row indicate the if a transaction has been issued involving the specified company and account. Positive ones indicates an inward connection, while negative ones indicate an outward connection.

From the connectivity matrix, it is apparent that the sole vertice representing the client company ($C1$ in the example) plays a central role in the transaction graph. Centrality is an important metric in graph as it informs how a vertice can influence its neighbors. More specifically, the *graph theoretic center* is defined in ([BVV15]) as the vertice with the smallest maximum distance to all other vertices in the network. This vertice is always the company vertice C in the case of a transaction graph $G_{C,T}$. We use this property to create payment patterns in order to characterize transaction graphs.

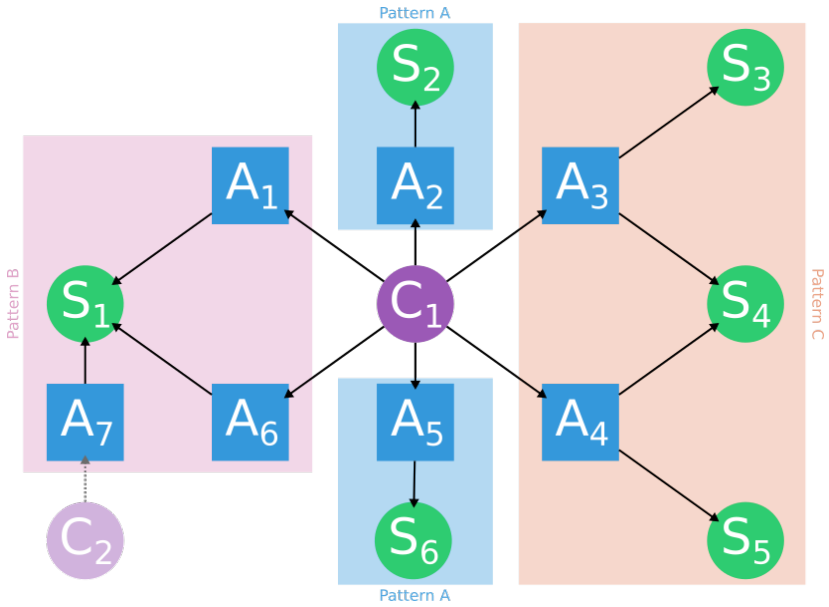

Figure 6.3: Transaction Graph T1. The node C_2 is shown here as an indication for the occurrence of account A_7, but it is not a part of the Transaction graph.

Table 6.1: Connectivity Matrix of Transaction Graph T1. Each column represents an account vertice. Each row represents a company vertice. Number are the number of time a transaction created an edge between the two vertices. The sign of the value indicates the direction of the edge: inward (positive) or outward (negative)
The node C_2 does not appear in the tabular form, as it is not part of the transaction graph.

	A1	A2	A3	A4	A5	A6	A7
C1	-1	-1	-1	-1	-1	-1	0
S1	1	0	0	0	0	1	1
S2	0	1	0	0	0	0	0
S3	0	0	1	0	0	0	0
S4	0	0	1	1	0	0	0
S5	0	0	0	1	0	0	0
S6	0	0	0	0	1	0	0

6.3.3 Payment Patterns

In this subsection, we describe how we use the specific properties of a transaction graph in order to create a set of features capturing the transaction relationship between a client and the accounts used to pay its suppliers. Our approach is akin to the one developed by [AMF10] where a similar featuring process is used to characterize ego-network of specific nodes in the graph. In order to create the features, we first remove the client company's vertice from the graph ($C1$ in Figure 6.3), thus creating a set of D disconnected sub-graphs composed of account vertices linked to supplier vertices (Figure 6.4). Among these D sub-graphs (that we dubbed "payment patterns"), if we only take into account the type of the node ("Supplier" or "Account") and not its label ("S_1" or "A_4"), then it might occur that some these sub-graphs are isomorphic, meaning that they share the exact same structure ([BVV15]). It is the case for example in Figure 6.4 where the sub-graph composed by (S_2,A_2) and (S_6,A_5) are isomorphic.

This set of features can also be translated in the connectivity matrix shown in Table 6.1. Removing the central node means ignoring the first row of the matrix. A

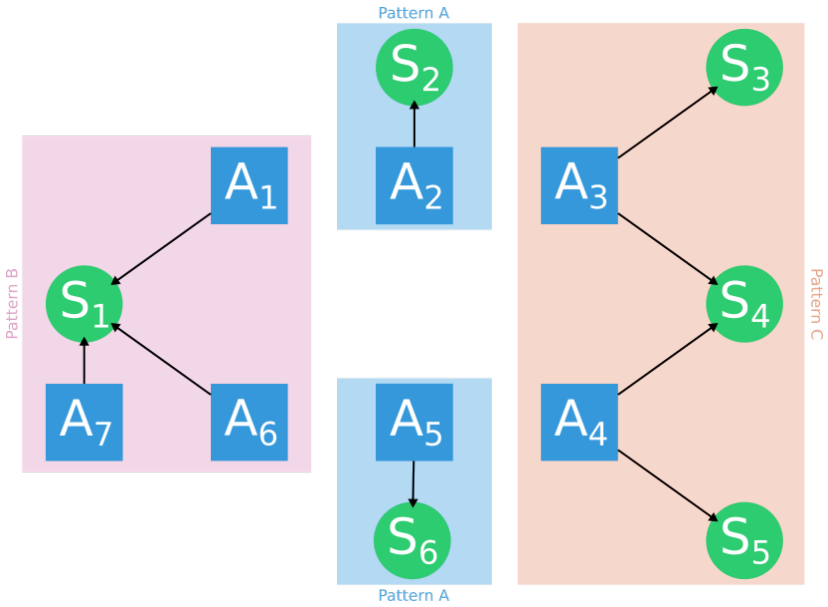

Figure 6.4: Subgraphs ('patterns') characterizing the transaction graph.

connected sub-graph can be connected in two ways: when a supplier is connected to a specific number of accounts (which is the case for S_1), or when an account is connected to a specific number of account (such as A_3). The case of Pattern C is a special one where the pattern satisfies both of these conditions.

Functionally speaking, these connected sub-graphs indicate how the client interacts with its suppliers, and thus shows a "map" of the client's activity in the set of T transactions used to create the transaction graph.

Algorithm 3 shows how the transaction graph is created. This algorithm can be used as is, but the graph created is actually a subgraph of a graph containing all the payments conducted by all the clients found in the History dataset. Creating this global graph, and then narrowing it down to the considered client, is an other way to compute this subgraph.

We then investigate possible number of unique payment patterns that can be created for a specific number of transaction based on Algorithm 3. This number depends on n the number of transactions used to create the graph. Let's consider an example: Let's assume $n = 1$. In this case, only one payment pattern exists, connecting S_1 and A_1 (similar to pattern A in Figure 6.4). If we have $n = 2$, then 3 payment patterns are created: one similar to patter A, that connect S_1 to A_1 twice, or S_2 to A_2 twice, or S_1 to A_1 and S_2 and A_2 separately. Another pattern might be created, where S_1 is connected to both A_1 and A_2. Finally, a last pattern might occur, where A_1 is connected to both S_1 and S_2. The number of payment patterns created corresponds to the number of graph with n vertices having a bipartite connected component [Slo+13]. As [Slo+13], the number of transaction patterns grows very fast (291396 graphs for $n = 10$). This number corresponds to the number of features of an histogram, while n corresponds to the size of a window. This fast growth in the number of features indicates that our data point might be placed in a very sparse high-dimensional space, and thus the distance between them will tend to lose its meaning. This corresponds the to the curse of dimensionality [Tru79].

Table 6.2: Examples of featurized sets of transactions

Transaction set ID	Pattern A	Pattern B	Pattern C	Pattern D
T1	2	1	1	0
T2	3	3	0	1
T3	1	0	0	0

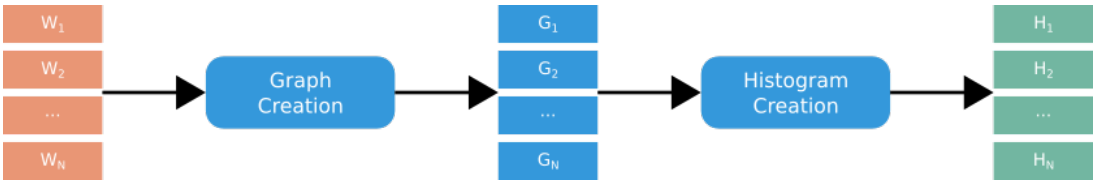

Figure 6.5: Graph histogram sequence creation.First, the transaction windows created in the pre-processing phase are converted into transaction graphs representing the relationships between a client and the accounts it uses to pay its supplier. Then the transaction graphs are in turn transformed into a set of feature vectors composed of the number of sub-graphs found in the transaction graphs.

6.3.4 Feature Set Creation

Once the transaction graph, we transform it into an histogram that represent its composition in a key:value form, so that it can be used to train a model. In order to do so, the occurrences of each payment pattern found in the transaction graph is used as a value, while the key is the payment pattern itself. Table 6.2 shows an example of such a process with T1 representing the graph shown in 6.3 and T2 and T3 representing other graphs. Similarly, the histogram corresponding to the test graph is also created using the same process. These histograms are then used as the basic features of our anomaly detection system.

Figure 6.5 shows an overview of the feature engineering process, proposed as a reminder.

6.4 Anomaly Detection

In this section, we describe the anomaly detection system we use in GraphSIF in order to assert if the test graph created by the tested transaction and the most recent transactions of C is similar to the graphs found in C's behavior sequence created from C's historical transactions.

The anomaly detection system relies on two main building blocks: a parametric clustering algorithm (K-means [Jai10]) that create clusters of transaction graphs represented as histograms according to their similarity, and a single-layer neural network called a Self-Organizing Map [Bul04] that project multi-dimensional features such as the histograms into a two-dimensional space, in order to facilitate the computation of distance between feature vectors and to alleviate the curse of dimensionality [Tru79] that states that the more dimensions, the more difficult it becomes to compute a meaningful distance between two feature vectors.

6.4.1 Overview

Figure 6.6 shows an overview of the anomaly detection process. First, the N histograms created at the end of the feature engineering phase are regrouped into clusters thanks to the K-means algorithm. Each of the histograms are associated with their

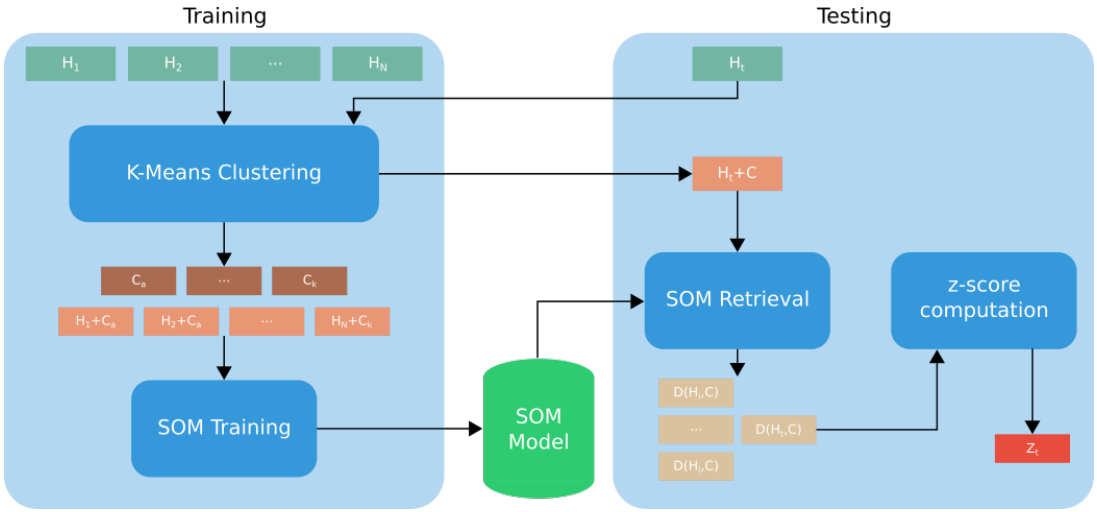

Figure 6.6: Overview of the anomaly detection process.

clusters, and the centroïd of each clusters C_i with $i \in K$ are also computed. K is the number of clusters set as parameter for the K-means algorithm.

Then, the histograms are used to train a Self-Organizing Map (SOM). In order to do so, each of the histogram is fed to the SOM, where an unique neuron (also called "node") is activated. The weights of this node and the eight neighboring ones are then updated in order to match the values of the histogram. The operation is repeated for each of the histograms until every one of them is associated with a node. Several histograms can be associated with the same node. Finally, once the SOM is trained, the centroïd of each cluster is fed to the SOM and the node activated by it is retrieved. This creates the "SOM model" that is composed of the nodes trained with the histograms along with the nodes corresponding to the centroïds.

Once the SOM model created, when an histogram corresponding to a test graph needs to be evaluated, it first goes through the clustering phase undergone by the other histograms in order to be assigned a cluster c. Then, it is fed to the SOM in order to find the node activated by it. The distance between this node n_t and the node activated by the centroïd of the cluster c (n_c) in the SOM (D_t) is retrieved, along with all the distance between the nodes activated by the members of c and n_c. All of these distances are then used to compute the z-score ([Abd07]) of D_t. The z-score is a statistical measure that tells us how many variation away a data point is from the mean. The higher the z-score, the less similar an histogram is from the others.

In the remainder of this section we detail the different algorithms used to obtain the z-score from our input data.

6.4.2 Training

In this subsection, we detail the training of the two models (the K-Means algorithm and Self-Organizing Map)) used in the anomaly detection system. Training the models means that we use the historical histograms created at the end of the graph-based feature engineering phase to fit the models so that they accurately represent the past behavior of the considered client. The training phase is divided in three parts: the histogram clustering where the histograms are assigned a cluster, the SOM training where the weights of the nodes of the SOM are adjusted to match the value of the histograms, and finally the histograms projection where the SOM nodes corresponding to the histograms and centroïds are determined in order to be used in the testing phase.

Algorithm 4: Histograms clustering using K-Means

Input :
- $\boldsymbol{H} = (h_1, ..., h_N)$: Set of histograms (observations)
- $\boldsymbol{C}^{(1)} = (C_1^{(1)}, ..., C_k^{(1)})$: Set of centroids at time 1

Output :
- $\boldsymbol{S_i^{(t)}}$: histograms for each cluster.
- $\boldsymbol{C}^{(t)} = (C_1^{(t)}, ..., C_k^{(t)})$: Set of centroïds at time t

1 $Init(\boldsymbol{C}^{(1)})$;
2 **while** *Not convergence* **do**
3 $S_i^{(t)} = \{h_p : \|h_p - C_i^{(t)}\|^2 \leq \|h_p - C_j^{(t)}\|^2 \; \forall j, 1 \leq j \leq k\}$ for $i \neq j$;
`// assign the histograms to the closest centroid.`
4 $C_i^{(t+1)} = \frac{1}{S_i^{(t)}} \sum_{h_j \in S_i^{(t)}} h_j$;
`// calculate the new position of the centroid as the mean of the observations in the new cluster.`
5 $t = t + 1$

Histograms Clustering

Clustering the histograms is the first step of the training process. It consists in using the K-Means ([Jai10]) algorithm on the set of historical in order to assign them a cluster based on their similarity according to a selected distance metric. K-Means is a well-known clustering algorithm that assign clusters to feature vectors according to their proximity to a centroïd that is the mean of the member of the clusters when the algorithm reach convergence. This proximity is computed according to a distance metric such as the Manhattan distance or the Euclidian distance. We use the Euclidian distance in our current implementation.

Algorithm 4 shows an overview of the algorithm.

When using K-means, it is very important to carefully choose the number of clusters K so that it represents accurately the underlying distribution of the data. Due to time constraints a thorough analysis of the optimal number of parameters could not be performed. In order to do so, a gradient-based optimization could be performed using either a set of synthetic anomalies, or based on the Audit dataset results. A preliminary experimental study conducted on selected test client showed that $K = 3$ seems to yield the best results in terms of stability of the cluster for our test case. This value indicates that the targeted company have three major components in its payment behavior.

The motivation behind the use of a clustering algorithm as a first step in our training process is to partition the local profile of a user in order to create representations of significant transactions graph occurring in the client's history. Figure 6.7 shows an overview of the process. In this toy example, the client's history of transaction have been processed into 5 transactions graphs that we dub H_1, H_2, H_3, H_4 and H_5. In H_1 and H_3, two accounts are respectively used to paid two suppliers, in H_2 and H_3, two accounts are used to paid only one supplier, and in H_5 three accounts are respectively used to pay three suppliers. The Kmeans clustering algorithm is performed on the 5 transactions graphs in their histogram forms, with the parameter k set to 2. 2 centroids (C_1 and C_2) are created. Due to the clustering algorithm, H_1, H_2 and H_3 are assigned the same cluster (let's assume it is C_1). This cluster is thus an approximation of a representative way the client pays its supplier. Similarly, due to their similarity, H_2 and H_3 are assigned the same cluster (let's assume it is C_2). This cluster represents an other way the client performs its transaction. Thus, clustering the graphs into different clusters allows us to extract meaningful representation of frequently occurring trans-

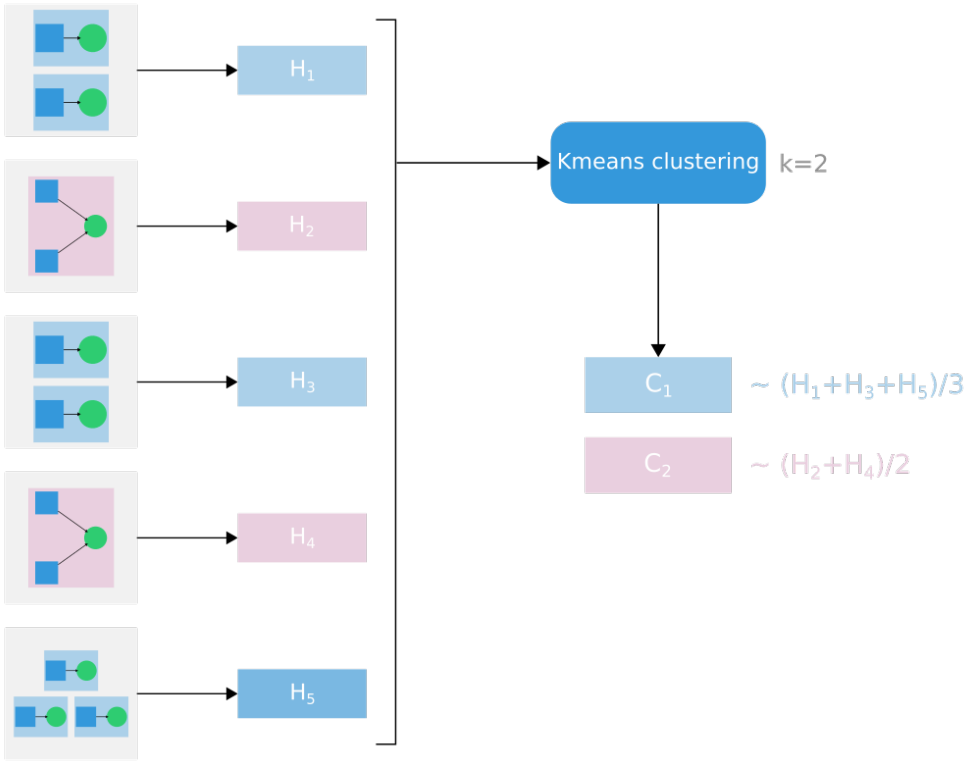

Figure 6.7: Toy example of the transaction graph clustering process. Five transaction patterns H_1, ..., H_5 are assigned clusters using the K-means algorithm. Due to their similarity, H_1, H_3 and H_5 are assigned the C_1 cluster, while H_2 and H_4 are assigned the C_2 cluster. These clusters represent typical transaction graphs found in the client's history.

action graphs, representing the usual behavior of the client. In real life, the transaction patterns are more complex, and thus the clusters become more fuzzy as the client's payment behavior becomes noisier.

This research could be furthered by studying the different clusters found and determining if they relate to a real behavior for the client company (such as the acquisition of materials, taxes payments, and so on). However, in the context of fraud detection, we solely focus on the fact that the clusters can be used as a point of comparison for new transactions graphs.

Alternative clustering algorithms such as the k-medoids algorithm ([PJ09]) or x-means algorithm ([Ish00]) might be investigated in order to optimize the clustering process.

Self-Organizing Map Training

One of the issue with using the transactions patterns described in the feature engineering phase as the dimension for our feature vectors is that we do not have direct control on the dimension of said feature vectors. Furthermore, as discussed earlier the number of possible patterns grows very fast with the number of transactions that are found in the set used to create the underlying transaction graph. This fact leads to two major issues: firstly, the feature vectors representing the algorithm might be projected in a high-dimensional but sparse feature space, thus resulting in artificially high distance due to the curse of dimensionality. Secondly, it is hard for a human to interpret the notion of proximity in such a high-dimensional space. In order to alleviate these two issues, we use a Self-Organizing Map, and more particularly a Kohonen network. ([Koh89],[BXD06],[Bul04]). This type of network aims to organize all of the feature vectors in a two-dimensional plane according to their similarity.

A SOM is an unsupervised neural network based on competitive learning, in the form of a neural network where only one neuron (also called node) is activated at any one time. The specificity of the Kohonen network is that the single computational layer is arranged in rows and columns, and each node is fully connected to all the source nodes in the input layer. In order to train the SOM, after an initialization phase, three steps are performed until convergence: sampling, matching and updating.

In the initialization phase, each of the neurons of the computational layer of the SOM are assigned random activation weights. The values of the weights needs to be reasonably close so that every neuron has a chance to be activated. The number of activation weights of a neuron is the same as the dimension of the feature space.

In the sampling phase, a sample x is drawn from the set of feature vectors X. This sample is randomly chosen so that the ordering of the set does not have any impact on the training process.

In the matching phase, the winning neuron $I(x)$ is found by comparing the weight vector of each neurons and finding the one closest to the values of the input vector. More specifically, the similarity of the vector to a neuron j's weights is computed using a discriminant function $d_j(x) = \sum_{i=1}^{D}(x_i - w_{ji})^2$. Closely related input vectors activate the same winning neuron $I(x)$.

In the updating phase, the winning neuron and its neighbors are updated using the equation

$$\Delta w_{j,i} = \eta(t) T_{j,I(x)}(x_i - w_{ji}). \tag{6.1}$$

In this equation, $T_{j,I(x)}$ symbolizes the topological neighborhood of the winning neu-

rons, and is defined as $T_{j,I(x)} = e^{\frac{-S^2_{j,I(x)}}{2\sigma^2}}$ where σ is the size of the neighborhood of the winning neuron. $\eta(t)$ is the learning rate that dictates how the weights of the winning neuron and its neighbors gets to be updated to a value closest to the input vector. This learning rate is defined as

$$\eta(t) = \eta_0 \, e^{\frac{-t}{\tau}} \tag{6.2}$$

where η_0 is the initial learning rate and τ a parameter for the exponential decay function that decrease the learning rate over time.

The sampling phase, matching phase, and updating phase are repeated until the SOM reach convergence, meaning that no significant modification in the weights occurs.

In our setting, the set of feature vectors X corresponds to the set of histograms H created by the feature engineering process.

Histograms Projection

Once the SOM is trained and convergence is reached, an additional step is performed: each histogram h found in the set of created histograms H is fed to the SOM, and the neurons $n(h)$ activated by the histogram is associated to it. Similarly, each centroïd c of k centroïds created by the clustering phase are also fed to the SOM and the neurons $n(c)$ are associated with the centroïds.

This phase allows us to project the histograms from their high-dimensional feature space to the 2-dimensional space of the SOM nodes, in a way that preserves their similarity. A valuable effect of this projection is that it enables a human expert to read and interpret the created map, and thus allows us to use human knowledge to understand the transactions patterns uncovered.

6.4.3 Testing

In the testing phase, a transaction is given to the anomaly detection system in order to assert its similarity with C's historical transactions. Before being submitted to the anomaly detection system, the test transaction is integrated to the most recent transactions of C and turned into a transaction histogram following the steps of the feature engineering process previously described. The result of this phase is a legitimacy score summing up how distant the transaction is from the historical ones. This phase is divided in 3 steps: the test histogram clustering, then the SOM distance retrieval, and finally the similarity computation. Algorithm 5 sums up the test process.

Test Histogram Clustering

In this step, the test histogram h_t is assigned a cluster based on the centroïds found in the training phase. It is assigned the cluster whose Euclidian distance is the closest, following the equation

$$c_{h_t} = \text{argmin}(c_i : \{\|h_t - C_i\|^2 \leq \|h_t - C_j\|^2 \; \forall j, 1 \leq j \leq k\} \text{ and } i \neq j \tag{6.3}$$

Assigning a cluster to h_t allows us to compare it to the historical transactions closest to it. Furthermore, as the centroïds determined by K-means algorithm represent the mean of all the historigrams of the cluster, it is the representative member of the cluster. Thus, the farthest h_t is from the centroïd, the less similar to the other member of the cluster it is. In other words, the further away h_t is is from the centroïd, then the closer from the edge of the cluster it is, and thus is dissimilar to the other members of the cluster. However, as previously mentioned, the curse of dimensionality might hinder the computation of a meaningful distance in our case. Thus, we use the trained SOM in order to compute the similarity of h_t with the member of its cluster.

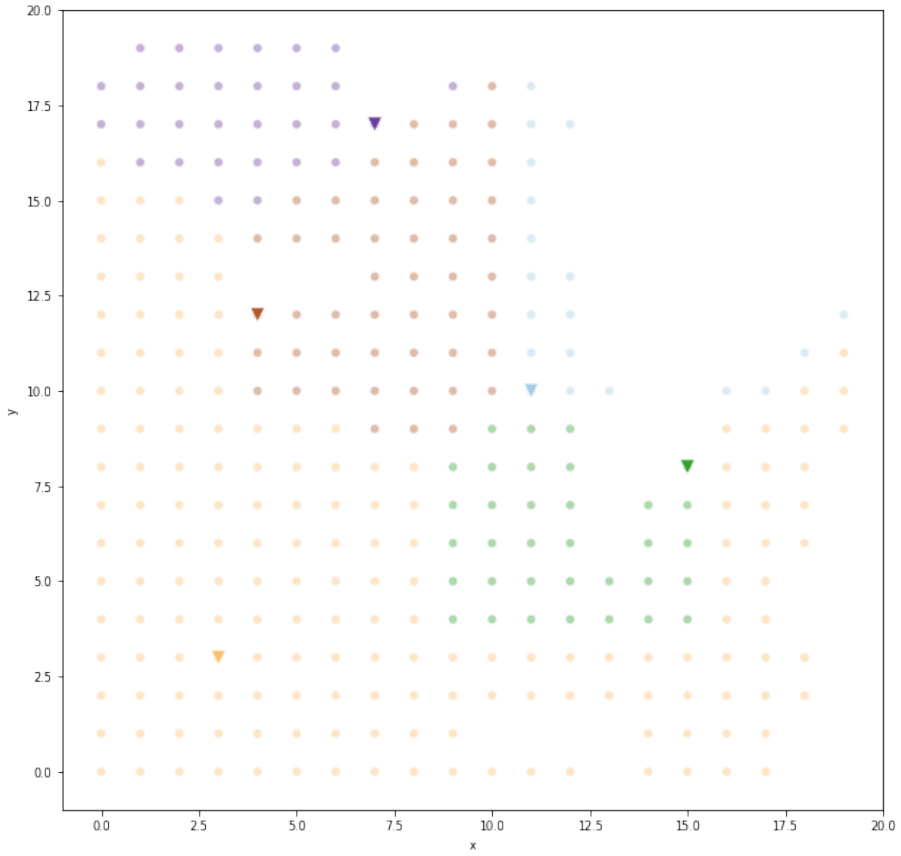

Figure 6.8: Trained SOM. Each point represents a neuron. During the training phase, each of the transaction graphs in its histogram form, along with its assigned cluster, is fed to the SOM and activate one neuron. This figure represents the fully trained SOM where each node is colored by the cluster most found among the transaction that activated it. The triangles represent the nodes activated by the centroids of each cluster. Nodes with no color are nodes that have not been activated by any transaction histogram.

Algorithm 5: Test phase of GraphSIF

Inputs :

- t: Tested transaction.
- w_n: Latest transaction window.
- $C_1, ..., C_k$: Cluster centroids for each clusters.
- $\boldsymbol{D} = \{D(n(h_i), n(c_{h_t}))\ \forall h_i \in S_{h_t}\}$: Distance from the cluster center to each transaction graph in the cluster, for each cluster.

Outputs :

- z_{h_t}: z-score of histogram h_t

$h_t = w_N + t$ `// Test histogram is created`

$c_{h_t} = \text{argmin}(c_i : \{\|h_t - C_i\|^2 \leq \|h_t - C_j\|^2\ \forall j, 1 \leq j \leq k\}$ and $i \neq j$;
`// Test histogram is assigned a cluster`

$D(n(h_t), n(c_{h_t})) = \sqrt{(n(h_t).x^2 - n(c_{h_t}).x^2) + (n(h_t).y^2 - n(c_{h_t}).y^2)}$;
`// SOM distance from cluster centroid is retrieved()`

$\mu_{\boldsymbol{D}} = mean(\boldsymbol{D})$;

$\sigma_{\boldsymbol{D}} = std(\boldsymbol{D})$;

$z_{h_t} = \frac{D(n(h_t), n(c_{h_t})) - \mu_{\boldsymbol{D}}}{\sigma_{\boldsymbol{D}}}$;
`// Similarity computation is performed`

Self-Organizing Map Distance Retrieval

In this phase, we feed h_t to the previously trained SOM and thus retrieve n_{h_t} the neuron activated by h_t, and we compute $D(n_{h_t}, n_{c_{h_t}})$ the Euclidian distance between the neuron activated by h_t and the neuron activated by c_{h_t} that is the centroid of the cluster assigned to h_t. This computation is performed in an off-line fashion prior to the inference phase, and the results stored in a cache in order to speed up the inference process.

Once this distance acquired, it might be tempting to use it directly as a way to determine h_t's legitimacy score, by for example assigning a threshold distance from which h_t would be considered anomalous. However, assigning a value to the threshold might prove a challenge as the distance corresponds to a neuron-to-neuron distance and not a vector-to-vector distance. Thus, it is not clear what the meaning of such a threshold would be. We propose a solution to this issue in Section 6.5.

Similarity Computation

In order to provide a more robust way to assert h_t's similarity, the distances from the other histograms members of the cluster C_{h_t} and its centroïd (that is, $\{D(n(h_i), n(c_{h_t}))\ \forall h_i \in S_{h_t}\} = \boldsymbol{D}$) are also retrieved. Once retrieved, the z-score of $D(n(h_t), n(c_{h_t}))$ with respect to $\boldsymbol{D}$ is computed. The z-score, as described in [Abd07], is a statistical metric that corresponds to how many standard deviation away a data point is from the mean of an ensemble. In our case, it gives us insight about how far away h_t is from the centroïd, with respect to all the other members of the cluster. The higher the z-score is, the more dissimilar h_t is from the other members of the clusters (with respect to the distance from the centroid), and thus the more likely it is to be an anomaly.

6.5 Label Attribution

The previously described training and testing algorithm uses a set of historical histograms to assert the similarity of a test histogram, through the computation of a z-score. However, the computation of these histograms are dependent on the size of the window of transaction *ws* that is used in the pre-processing algorithm. Indeed, the window size directly impact the graph extracted from the window, and thus the histogram describing the graph.

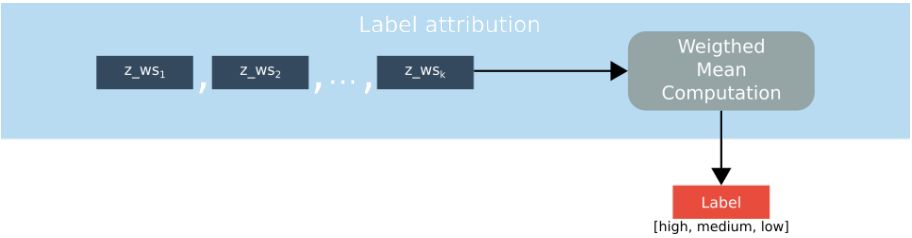

Figure 6.9: Label attribution process.

6.5.1 Impact of windows size

A small window size will create smaller graphs that encompass the short-time behavior of a client C. Furthermore, a small windows size will also provide more behavior histograms to perform the clustering and SOM training, at the expense of a decrease in the complexity of patterns found in a graph, as less transactions means less possible relationships between account and supplier. Lastly, anomaly detection using graphs created from small windows are less stable as adding a single transaction can create a huge topological difference between two graphs.

On the contrary, a large windows size will create larger graphs, that sums up a large number of transaction and thus the long-term behavior of the client. However, as the number of transactions needed to create the graphs will be higher, less data points will be created. As a certain amount of data point (depending on the size of the SOM) is needed for the training algorithm, very high windows size are thus not suitable for the detection system. However, larger transaction graphs means that more complex patterns might be formed, that would have been missed if smaller windows were used. Lastly, using large windows size means that the topological modification following the introduction of a single transaction might not be enough for the anomaly detection system to pick up an anomaly.

Additionally, the size of the windows have a great impact on the performance of our system, in two ways. Firstly, a small window size means that more windows will be created, and thus the graph creation phase has to be run a higher number of times. This task can be expansive in terms of computational power, but can be conducted in a parallel fashion quite easily. Secondly, a large window size means that more transaction are taken into account in the graph, meaning that the algorithm needs to process more data. Furthermore, as described earlier, the number of payment patterns derived from a high number of transaction might become numerous.

In this implementation of our system, we didn't take into account the temporal dimension of the analysis. More precisely, we did not discount the impact of the oldest time windows on the overall computation of the label. In order to do so, a discounting factor could be applied to the windows containing the oldest transactions.

6.5.2 Optimizing windows size

A straightforward way to determine which windows size is more suitable for anomaly detection for a specific client C would be to perform an optimization method such as grid search ([BB12]) or random search ([BB12]). However, these optimization methods rely on the assumption that target labels are available, which is not the case in our use-case. Thus, an alternate method has to be found.

Instead of trying to optimize the size of the window, a possible solution would be to perform the anomaly detection in a range of different windows size, and aggregate the results in a way similar to bagging ([Die00]). This way, the anomaly detection system would be able to draw its conclusion from both small size transaction graphs

and high size transactions graph when assigning the legitimacy score of a transaction.

As a way to perform this aggregation, the following algorithm is proposed. For every size *ws* in $\boldsymbol{W}$ of length $l(\boldsymbol{W})$, the pre-processing phase, feature engineering phase and anomaly detection phase of the anomaly detection system are performed, effectively creating $l(\boldsymbol{W})$ anomaly detection systems, and a z-score is computed for a transaction t from each of them. Then, the following process is applied:

1. The z-scores undergo a discretization process when the score is turned into one of the three legitimacy labels ("high","medium","low"). In order to do so, two risk thresholds ($0 < r_1 < r2 < 1$) are used as parameters for the threshold function. These risk thresholds represent the fraction of the maximal z-score corresponding to each of the legitimacy labels. As a rule of thumb, a z-score of 3 indicates that a value is an outlier with respect to a give data set. Thus, a risk score $r_1 = 0.2$ indicates that if the z-score of a given value is smaller than $0.2 * 3 = 0.6$ then it is considered legitimate by the anomaly detection system. The risk thresholds are defined by the investigation team, as it relies on the costs implied by a false positive (legitimate transaction mislabeled as fraud) or a false negative (fraud mislabeled as legitimate transaction). These costs depends on factors that reside outside of the scope of the anomaly detection system, and thus need to be asserted by a team of experts.
2. Each of the label is assigned a weight (w_h,w_m,w_l) that represent a bonus in the overall legitimacy score. These weight can be parameterized by the investigator in order to control the sensitivity of the system. Usually, $w_h > w_m > 0 > w_l$ so that "high" legitimacy labels pull the score up and "low" legitimacy label decrease the overall legitimacy score.
3. The sum of the $l(\boldsymbol{W})$ weights is computed and normalized so that an overall legitimacy score $0 < L_{\boldsymbol{W}} < 1$ is calculated.

$L_{\boldsymbol{W}}$ can also be discretized using a threshold function if the need to provide labels is found. This threshold function can use as input a list of threshold risks $\delta_1, \delta_2, ..., \delta_n$ corresponding to the operational needs of the fraud detection team. Alternatively, a voting system might be used in order to aggregate the value of each of the anomaly detection systems. This label attribution phase thus alleviates the need to search for an optimal value of *ws* and allows the anomaly detection to be performed on both short-term and long-term transaction behavior of the investigated client *C*.

6.6 Experimental Results

In this section, we discuss the experimental setting used to assert the performance of GraphSIF, and show the experimental results obtained by running GraphSIF on a set of transactions previously labeled by SiS-id expert system. The evaluation process follows the one used in Chapter 5, with a case study featuring a single client, followed by a global analysis.

6.6.1 Experimental Settings

In order to produce these results, GraphSIF has been run on a laptop running Ubuntu 18.04.4 with an Intel Core i7-10510U CPU and a 15,5 GiB memory. The data used to train the model is the History dataset previously described in Chapter 4 Section 2, consisting in a set of unlabeled transactions between client companies and suppliers companies. The data used to test our model is the Audit dataset described in Chapter 4 Section 3.
In order to perform our experimentation, all of the 62 client company found in both the History and the Audit dataset, with more than 100 transactions are selected. The corresponding subset of transaction from the History dataset is used to train GraphSIF, the transactions from the Audit dataset are used to assert the consistency of the results

Parameter	Value
Window size range	[2, 5, 8, 11, 14, 17, 20, 23]
K	3
Z-score thresholds	[0.2, 0.5]
Std max	3
Label weights	[20, 10, 0]
Risk thresholds	[0.8, 0.5]
SOM Parameter	**Value**
Map Size	10x10
Lattice	rectangular
Normalization	var
Initialization	PCA
Neighborhood	Gaussian
Training	batch

Table 6.3: Parameters used in the experience.

with SiS-id's expert system.
The different parameters used in the experiment are described in Table 6.3.

6.6.2 Case Study

In this section we evaluate the performance of GraphSIF on the subset of transaction issued by a single client *C*. In this evaluation, only the transaction issued by *C* in the Audit dataset are considered. This subset contains 1000 test transactions to evaluate the performances and 400 000 historical transactions for the History Dataset to train the model. First, we analyze the consistency of GraphSIF with the expert system, and then we present the operational efficiency of our system.

Consistency

We first consider how close the results of GraphSIF are with the results of the expert system. While these results can not be considered ground truth, as the expert system does not distinguish between fraudulent and invalid transactions, they still provide a baseline for the analysis of GraphSIF results.

High Legitimacy Label

First, we consider the results given by both the expert system and GraphSIF concerning the high legitimacy labels. Figure 6.10 shows a Venn diagram indicating the distribution of the high legitimacy label. The consistency of the two set is not high, only 19 out of the 167 transactions given a "high" legitimacy label by the expert systems are given the same label by GraphSIF. However, this inconsistency might be explained by the fact that the risk thresholds used as parameters of the systems are set to different values that the rule-based system. Furthermore, the expert system relies on knowledge internal to the platform, in the form of a registration of secured suppliers, that is not available for the graph-based model.

Table 6.4 shows the confusion matrix indicating the complete distribution of each label. Most of the high legitimacy labels (128 out of 167) have been labeled as low legitimacy transactions by GraphSIF. This behavior is consistent with the hypothesis of the expert system using additional knowledge to perform its classification. Table 6.4 also shows that, out of the 104 transactions assigned a high legitimacy label by GraphSIF, only 11 where attributed a low legitimacy label by the expert system. This means that GraphSIF mostly does not allow payments that would be deemed fraudulent by the expert system.

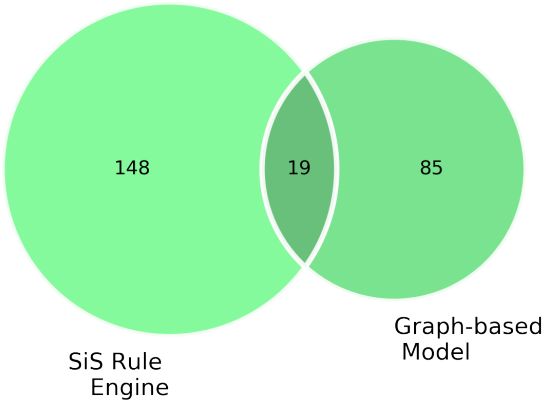

Figure 6.10: Distribution of "high" legitimacy label

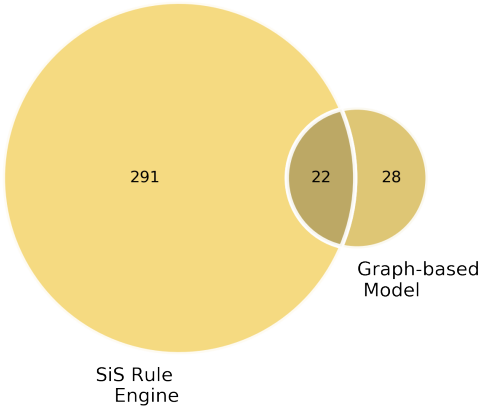

Figure 6.11: Distribution of "medium" legitimacy label

Medium Legitimacy Label

Then, we consider the results given by both the expert system and GraphSIF concerning the medium legitimacy labels. Figure 6.11 shows the distribution of the medium legitimacy label. For the expert system, a medium label indicates a lack of knowledge. Almost a third (313 out of 1000) of the transactions have been assigned this label by the expert system. On the contrary, a medium label doesn't indicate a lack of knowledge for GraphSIF, but rather informs the user that a specific transaction is slightly unusual. Thus the consistency between the two sets is not really expected. However, the low number of transactions labeled with the medium label by GraphSIF (50 out of 1000) seems to indicate a higher assertiveness of the proposed model. Table 6.4 shows that most (217 out of 291) of the medium labels given by the expert system where assigned a low legitimacy label by GraphSIF.

Low Legitimacy Label

Finally, we consider the results given by both the expert system and GraphSIF concerning the low legitimacy labels. Figure 6.12 shows the distribution of the low legitimacy label. There is a high consistency between the expert system and GraphSIF concerning this set of suspicious transactions, as 501 out of 520 transactions where given the low legitimacy label by both of the detection systems. The 345 transaction assigned a low legitimacy transaction by GraphSIF and not the expert system come from the two other sets. Furthermore, the last row of Table 6.4 shows that only a handful of transaction labeled as low by the expert system have been given an other label by GraphSIF.

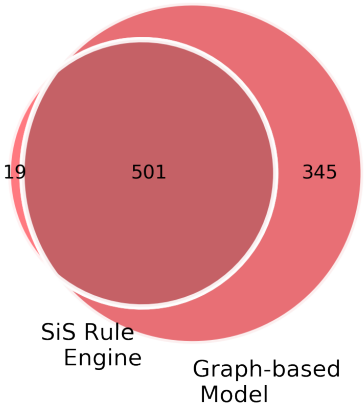

Figure 6.12: Distribution of "low" legitimacy label

Table 6.4: Confusion Matrix - SiS Rule Engine's and GraphSIF

Expert system → GraphSIF ↓	High	Medium	Low
High	**19**	74	11
Medium	20	**22**	8
Low	128	217	**501**

It is possible that the 11 transaction given a low legitimacy label by the rule engine could use knowledge not available for GraphSIF, such as when a supplier closes an account.

Operational Efficiency

In this section, we consider the operational efficiency of GraphSIF. GraphSIF, as opposed to the rule engine used by the expert system, requires a pre-processing phrase and a training phase to be completed in an off-line fashion. Then a transaction can be assigned a label.

Pre-processing time

Figure 6.13 shows the time taken by GraphSIF to pre-process the transaction. The results are separated by windows size, as this parameter has a huge impact on the quantity of windows created, and the amount of transactions in each window.

The pre-processing time seems to vary exponentially between 500 ms and 2500 ms depending on the windows size. A shorter windows size leads to higher pre-processing time, probably due to the fact that more windows are created.

In the case of the expert system, a pre-processing phase is also performed, requiring calls to internal and external APIs. This pre-processing phase can take up to 3 seconds, which is the same order of magnitude as GraphSIF's pre-processing time, due to the need of adding the pre-processing time of all the windows. However, GraphSIF relies entirely on local data, whereas SiS-id's expert system makes use of several external API's, and thus its pre-processing time is dependent on the state of the network, and availability of the APIs.

Training time

Once the pre-processing phase is complete, the training phase can begin. Figure 6.14 shows the time taken to perform the training phase. We can see that this time il also correlated with the windows size, and varies from 28 seconds to 25 seconds as the

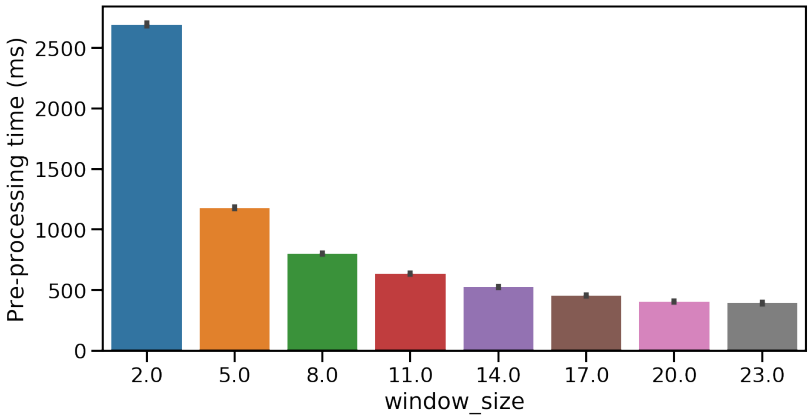

Figure 6.13: Pre-processing time for each window size.

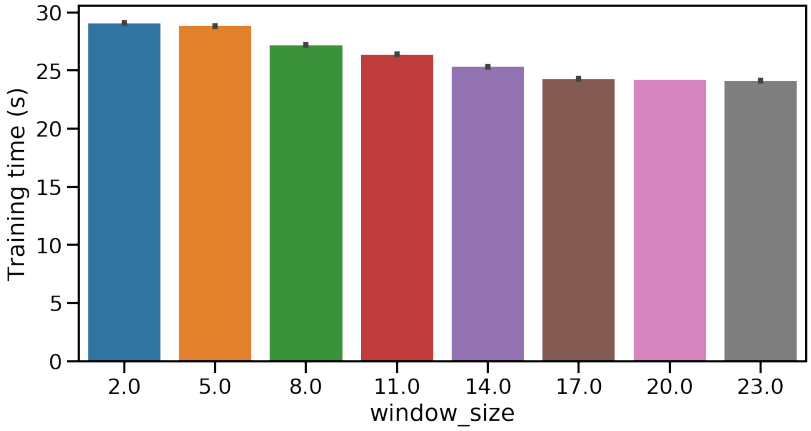

Figure 6.14: Training time for each window size.

windows size becomes larger. Two reasons might impact the decrease in training time: the number of windows used to train the SOM is reduced and thus less data is available for training, or, as the windows grow larger, more distinctive patterns emerges and thus the training is more easily performed. This training phase can be performed in an offline fashion and does not impact the inference time of the system, thus such values were deemed acceptable.

In the case of the expert system, there is no training phase, but however the rule engine has to be updated in order to adapt to new fraud cases. This process is long and costly, as it relies heavily on human input. GraphSIF data-driven evolution is thus an improvement.

Test time

Figure 6.15 shows the time taken to perform the classification of each transaction in the test set. While a slight increased is visible for a windows size of 2, the test time is mostly uniform and close to 300 ms. The expert system classification time is closer to 3 seconds, as the pre-processing has to be performed for each tested transaction. Thus GraphSIF is faster than the expert system. The value of 300 ms in average is moderately fast considering that the detection is performed multiple time over several window sizes, and possess a graph processing component. Furthermore, as opposed to credit card fraud transaction where fraud detection must be conducted in a near-instantaneous fashion, SIF detection has a larger granularity as the opportunity to cancel a fraudulent B2B exists.

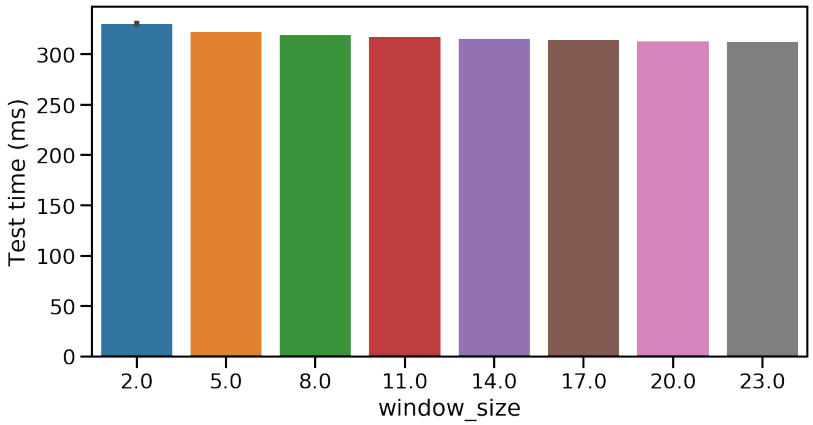

Figure 6.15: Testing time for each window size.

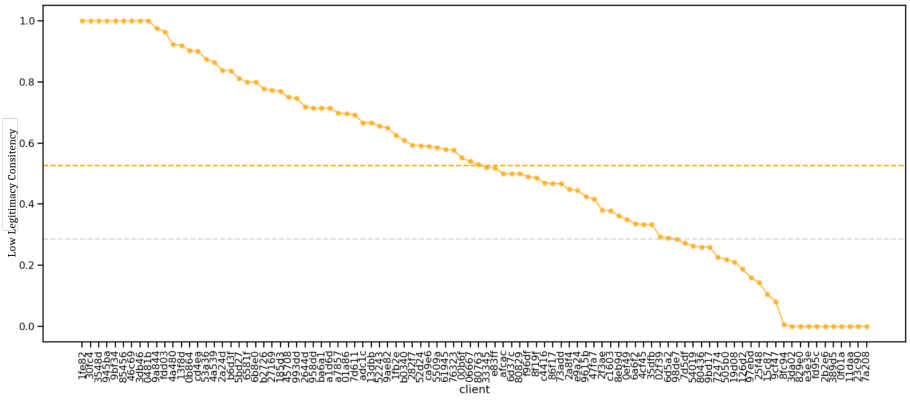

Figure 6.16: Low legitimacy consistency score for each client in the Audit Dataset.

6.6.3 Global Results

In this section we focus on GraphSIF's behavior with the other clients of the Audit dataset, in terms of consistency and operational efficiency.

Global Consistency

In this section we use the Low Legitimacy Consistency (LLC) score and High Legitimacy Consistency score (HLC) previously described in Chapter 5. LLC corresponds roughly to True Positive Rate (assuming the fraudulent transaction are considered the positives) while HLC corresponds to True Negative Rate (in the same setting).

Figure 6.16 shows the low legitimacy consistency score for each clients of the Audit dataset. It shows how consistent GraphSIF is with the expert system regarding suspicious transactions. The dotted orange line shows the mean LLC over all the clients (0.53), while the grey dotted line shows the mean LLC of a random classification machine (0.29). We can see that while there is still room for improvement, GraphSIF shows promising results in terms of consistency.

Furthermore, Figure 6.16 allows us to pinpoint a set of client on which GraphSIF is not performative (rightmost ones). These clients are a promising starting point for further improvements.

Figure 6.17 shows the high legitimacy consistency score for each client in the Audit dataset. The HLC in our setting is fairly low (0.24), even lower than the one produced by randomly guessing a label for the transactions (0.27). This highlight the

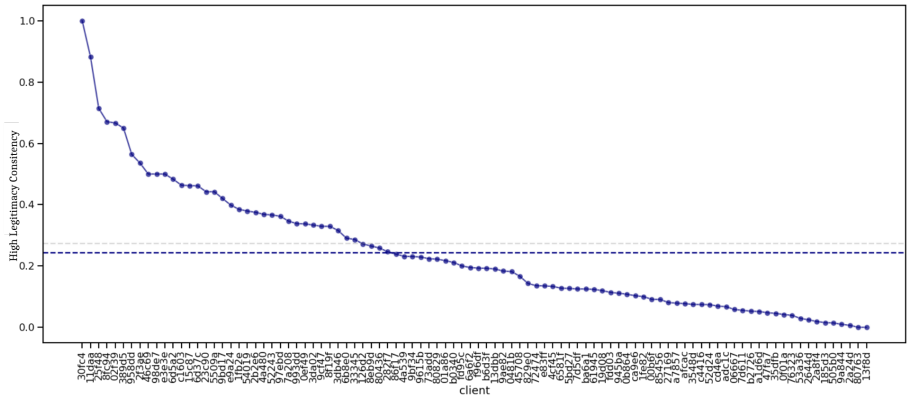

Figure 6.17: High legitimacy consistency score for each client in the Audit Dataset.

fact that GraphSIF tends to assign a low number high legitimacy labels overall.

However, in the context of fraud detection, it is more costly to assign a high legitimacy label to a fraudulent transaction than to assign a low legitimacy label to a legitimate transaction. Thus, while improving the high legitimacy consistency score is important for the future, this discrepancy does not have a tremendous impact on GraphSIF usage.

6.6.4 Discussion

While results shows that GraphSIF is able to provide consistent results in terms of consistency with the expert system, the most pressing issue is the lack of ground truth about actual legitimate and fraudulent transaction. While SiS-id's expert system provide an equivalent to expert knowledge regarding the labeled transaction, this system is prone to error, most notably because fraud is dynamic and often changes faster than the rules of an expert system. Furthermore, comparing GraphSIF graph-based approach an SiS-id expert system is difficult, as they both rely on different type of knowledge (expert-based vs data-driven).

However, in the absence of a dataset containing trusted labels, comparing our systems with SiS-id's expert system is the only way to propose an evaluation of their performance. We presented the result of our anomaly detection system on a single client to investigate its performance locally, and shows that the results, while consistent in terms of low legitimacy payments, differs from the expert system when it comes to high legitimacy transactions. This might be explained by the fact that the process behind the two system is very different: while the expert system relies on expert knowledge about the payment and might even use external knowledge, GraphSIF only consider the relations between companies and accounts in order to perform its analysis. However, GraphSIF shows consistency with the expert system regarding the low legitimacy labels, and its training and testing time are lower than the expert system. As the two systems rely on different features to perform their analysis, it might be suitable to use both GraphSIF and SiS-id's expert system in a complementary fashion. An open question would then be, how to combine the insights of these systems in a single answer ? A way to do so would be to integrate the result of GraphSIF as a new feature in a meta-analysis, containing also the features proposed by the expert system.

We presented the use of a neural network in the form of a Self-Organizing Map. Recent advances in the field of neural networks proposes a number of network showing promising results in a wide array of different fields. More specifically, Recurrent Neural Networks (RNN) sems to be especially suited for the task of SIF, as they are

tailored to address time-aware issues such as fraud detection. However, the lack of ground truth for SIF makes the training process of this kind of supervised neural networks almost impossible. However, recent advances in unsupervised networks such as Generative Adversarial Network have been made and might prove interesting to study.

In this work, GraphSIF only uses the relationship between one client and its suppliers to perform its fraud detection. However, an interesting research direction would be to consider a graph modeling the relationships between all the clients and all the suppliers of the platform, to form a graph modeling all the ecosystem rather than just the egonet centered on a specific client. Then, some graph-based metrics such as centrality or betweeness could be use as new features for SIF detection.

6.7 Summary

In this chapter, we introduced GraphSIF, a novel feature-engineering process that creates a feature vector based on the relationship between a client company and the accounts it used to pay its supplier company, providing a new tool to describe the underlying payment behavior involved in their interaction.

We then used the temporal information contained in the historical payments to create a behavior sequence composed of the transaction emitted by a client aggregated in several windows. We showed how to use this behavior sequence to create a data model based on Self-Organizing Maps representing the payment behavior of a client company through time. We then used this data model to infer the legitimacy of new payments using the K-means clustering algorithm, along with an aggregation algorithm allowing us to combine the results obtained for different window sizes in a global score.

We then presented the results of our system on a set of payments and compared them with the results provided by SiS-id's expert system. While GraphSIF shows consistency with the expert system when the low legitimacy transactions are considered, the high level legitimacy transactions are not labeled consistently between the two algorithms. GraphSIF also shows inferior training and testing time and relies solely on data-driven knowledge in the construction of its model. The only human input needed in GraphSIF is the determination of the risk threshold.

7

Conclusion

As a conclusion, we sum up the process undergone in the work of this thesis to proposes a data-driven Supplier Impersonation Fraud (SIF) detection system. In Chapter 1, we introduced the notion of Supplier Impersonation Fraud and underlined the motivation of using a data-driven approach to build SIF detection systems. In Chapter 2, we described in details the B2B ecosystem where the Supplier Impersonation Fraud takes place, and provided impersonation scenarios based on real-life cases. By analyzing these scenario we determined that the key to a successful SIF was the tampering of the supplier's transaction. In Chapter 3, we presented the State of the Art system in data-driven unsupervised fraud detection, along with the techniques used to build such systems. In Chapter 4, SiS' expert system and the datasets used in SIF detection were presented and discussed. In Chapter 5, we introduced our first contribution in the form of ProbaSIF, a data-driven SIF detection system based on Bayesian Models that performs SIF detection. Its results were compared with SiS-id's expert system and show a high consistency in term of potentially fraudulent transaction, while being purely a data-driven model relying on almost no human input. In Chapter 6 we presented GraphSIF, a SIF detection system which uses both the contextual and temporal features found in the history of transactions of a client to construct a new feature vector based on graph analysis. This vector is then used to build a company's payment behavior and suspicious transactions are compared with this behavior in order to assert their fraudulence.

7.1 Summary of our contributions

In this section we sum up the major contribution of this thesis: an overview of the SIF issue, ProbaSIF, and GraphSIF.

7.1.1 Supplier Impersonation Fraud

We first provided a comprehensive framework to describe BtoB transaction and Supplier Impersonation Fraud (SIF) by describing and analyzing three real-life scenarios where SIF takes place. These scenarios where provided by SiS-id, a company specialized in SIF detection. We then proposed a formal description of a B2B transaction, and isolated the bank account used in the payment issued from a client company to a supplier company as a distinctive feature for fraud. We argued that tampering with the

bank account is mandatory for a fraudster to perceive the benefits of its fraud.

We also provided a description of a SIF automated detection system and their challenges: unbalanced class representation, quick concept drift and noisy dataset. We proposed a description of the most recent tools and techniques used to perform fraud detection in fields related to supplier impersonation fraud, as to the best of our knowledge there exist no public work in the literature that focuses solely on SIF.

7.1.2 Probability-Based SIF Detection System

We proposed a new SIF detection system, named ProbaSIF, based on probability theory and Bayesian inference. This system extract the bank accounts used in historical transactions from the dataset and use them to assert the legitimacy of new transactions according the client and supplier involved. We use a Multinomial-Dirichlet distribution to model the probability of occurrence of an account, and use a user-defined threshold function to assign a legitimacy label depending on the probability value of the account used in the targeted transaction.

Two different probabilistic models have been considered for ProbaSIF: in the first one, only the historical transactions issued by the client involved in the targeted transaction are considered to build the model. In the second one, all the historical transactions received by the supplier involved in the targeted transaction are considered. Our results shows that using all the transactions received by the supplier seems to lead to more consistency with the expert system considering the low legitimacy labels.

We also provided an experimental evaluation of ProbaSIF running both models, and compared it with SiS rule-based system. The goals were to analyse its computational performance and to assert the consistency of ProbaSIF with the expert system in an explorative fashion. Results shows that while some discrepancy exists when considering "medium" legitimacy labels, the findings of the two models seem consistent when low legitimacy labels. However, consistency in the high legitimacy label is not important, meaning that a number of transactions considered as legitimate by the expert system were considered as fraudulent in by the two models.

7.1.3 Graph-Based SIF Detection System

Then, we investigated how the historical transactions of client companies could be use to detect fraudulent transaction. We proposed a supplier fraud impersonation detection system called GraphSIF, that relies on the relationship created by different companies and bank account, to assert the legitimacy of the transaction. GraphSIF is constituted of three different processes: firstly, a feature engineering process, then an unsupervised modeled based on Self-Organizing Maps and K-means clustering algorithm, and finally a decision process based on z-score and label aggregation.

The feature engineering phase is based on a bipartite graph modeling companies and back account as nodes, and linking them through the transactions they are involved in. Several graphs representing the interactions of a client with its suppliers are created, based on the aggregation of transaction in a bounded window. These windows are used to model the payment history of this client through time. These graphs are then transformed into feature vectors through the use of pattern-based embedding. These feature vectors are then used to train a Self-Organizing Map, as the number of dimension of the feature vectors can become high.

We then use K-means to compute clusters from the SOM, thus completing the training phase by creating an unsupervised model regrouping similar transaction into clusters interpretable in the SOM.

Then, in order to assert if a transaction is legitimate or fraudulent, first a new feature vector is computed by adding it to the most recent window. Then, a cluster is assigned to this feature vector, and the z-score between the node of the SOM attributed to the feature vector and the node attributed to the centroid of the cluster is computed. If this z-score is high, the transaction is considered as anomalous, otherwise it is considered as legitimate. This process is repeated for several windows sizes.

Experimental results shows that the window size chosen for the behavior sequence is a parameter with high impact, as a small window will lead to an increase in the computation time. Furthermore, the size of the window define the maximum number of payment patterns created. In order to provide a comprehensive result we designed an aggregation algortihm that relies on the analysis of different windows size in order to return a cohesive result.

GraphSIF was tested using the real data provided by SiS-id and its results were compared to the results of SiS-id's expert system in an exploratory fashion. Results shows that while relying on a very different assumption than the expert system, GraphSIF provides remarkably consistents answer with regards to low legitimacy transactions. Furthermore, the results found for our representative client seems to extend for a large number of other companies. However several test cases yield poor results and require further investigation.

7.2 Perspectives & Future Work

While this thesis provided a first attempt to propose data-driven systems in order to detect Supplier Impersonation Frauds, our work unveiled a number of open questions that are suitable research directions in the future. In this section we focus on two main directions: leveraging the issue of sharing sensible data by using Privacy-Preserving Machine Learning (PPML) algorithm allowing the detection to be performed on encrypted data, and modeling the B2B ecosystem as a global graph of exchanges between the companies in order to add more context to the fraud detection process.

7.2.1 Confidential Fraud Detection System

One of the major issue when conducting SIF detection is that the transactions between a company and its suppliers are often considered as sensitive data by the company. This stems from the fact that client/supplier relationships in a B2B ecosystem are the result of long term negotiations and a competitor or a fraudster obtaining such knowledge might then exploit it to perform economical attacks or impersonations.

Thus, companies often feel entitled to protect such information and requires a high number of guarantees before sharing the data with a third party, even for critical tasks such as fraud detection. While the protection of sensitive data have been a dynamic topic is science for a long time ([MH04]), most of the research focused on data storage protection, or data transmission protection, thus leaving a necessary step of the data analysis part without protection: the data analysis itself. However, recent research in Privacy Preserving Machine Learning (PPML) shows promising results in preserving sensitive data in the training and utilization of a data-driven systems ([JC14]). Such system, combined with adequate private storage and communication protocols, could ensure to a company that their sensitive data is protected at every step of their usage, and thus encourages more companies to share their data in the collaborative effort needed for Supplier Impersonation Fraud.

PPML can be roughly divided in two sets of techniques: anonymization techniques such as k-anonymity ([Swe02]) and differential privacy ([Ryf+18]), that aims to alter the data so that the sensitive data can not be linked to the source of the data. The

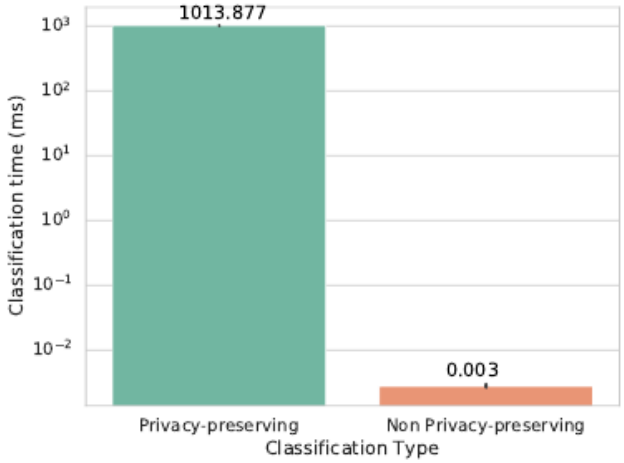

Figure 7.1: Latency in classification time induced by CIPHERMED. ([Bos+15]).

second type of PPML techniques are obfuscation techniques such as homorphic encryption ([Veu10]) that aims to encrypt the sensitive data so that their real value can only be known by the company and not by any of the other parties. These techniques borrow heavily from the field of Secure Multi-Party Computation ([GAC18]).

The challenges in introducing PPML to a SIF detection system are twofolds: first, the selection of a suitable privacy preserving technique adapted to the task of fraud detection needs to be carefully analyzed as, to the best of our knowledge, there is no attempt to use PPML for fraud detection for now. Furthermore, both anonymization and obfuscation techniques introduce a cost in the system, impacting the utility (accuracy, precision) of the system in the case of anonymization techniques, and the operational efficiency (time taken to train the model) in the case of obfuscation techniques. Figure 7.1 shows the overhead in classification time on a decision tree algorithm introduced by the Ciphermed obfuscation algorithm ([Bos+15]). An exploratory work ([Can+18]) shows that the trade-off in terms of computation time is correlated with the complexity of the underlying data model as shown in Figure 7.2.

In order to investigate PPML for SIF detection, an exploratory approach of homomorphic encryption [Pai99] used with classical models such as decision trees and Support Vector Machines have been proposed in [Can+18]. In this work, an homomorphic encryption scheme is used to obfuscate the values used to create the model, thus ensuring maximum privacy of the data. However, using such a scheme add to the computational complexity of the system. The execution time is directly related to the complexity of the underlying machine learning model and the size of the encryption key used in the encryption algorithm, and thus these two parameters could be taken into account in the optimization process of the system.

Once a suitable privacy-preserving technique is selected and the trade-offs are analyzed in details, PPML could be a major advance in SIF as it would incite more companies to openly share their data and thus, allows for the construction of more accurate models.

7.2.2 B2B Exchange Network Model

In GraphSIF, we use a bipartite graph to model the payment issued from a client company to its supplier company as a network linking companies and bank accounts used to receive payment. Such a network could be expanded by considering every transactions available in the B2B ecosystem, rather than only the ones corresponding to a specific client. Such a graph would thus model the payment relationships at the level

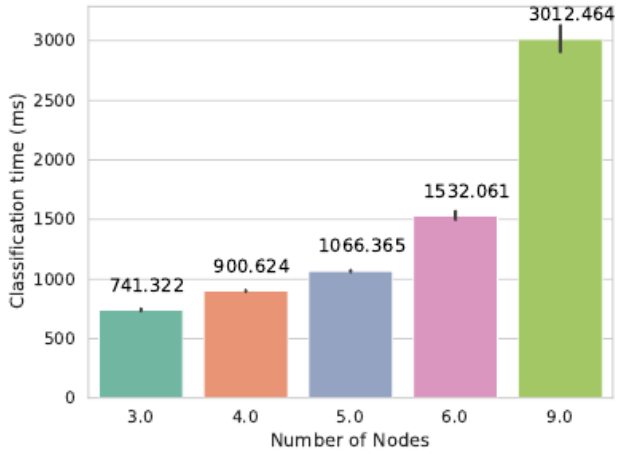

Figure 7.2: Impact of the Decision Tree complexity on the classification time in by CIPHERMED. ([Bos+15]).

of the whole ecosystem, rather than focus on the client's immediate neighborhood.

Figure 7.3 shows an example of a global B2B payment network built using the transaction data provided by SiS-id. This network contains as many nodes as there are unique clients, suppliers and accounts in the dataset (a total of 386 963 at the time of writing), and more than more than two millions edges. Such a network could be used for various analysis: first, as of today, no model of such B2B payment network have been proposed. The characterization of such a graph using latest graph theory tools (degree analysis, connectivity, community analysis) might yield useful insights in terms of company interactions and payment behavior. Such insight might be valuable for economical research and finance. However due to its size, suitable Big Data techniques (such as data distributed techniques such as map-reduce algorithm to parallelize graph creation and query, or the use of distributed database implementation specifically tailored for graph such as Neo4j) have te be explored in order to handle graphs of large size without computational issues.

A more complex model, such as one taking into account the number and time-delta between the transaction occurring in the B2B system, might be more suitable to represent the B2B ecosystem considered. This model, akin to a well-known large graphs such as Facebook's social network, could yield a number of interesting new features to enrich the companies and the transactions they perform, as they will accurately describe the context in which such transaction takes place. Furthermore, the study of the B2B graph itself, for example by investigating its similarity with well-known theoretic or real graphs, or by discovering meaningful subgraphs, yield in itself a lot of perspectives.

As for SIF, using such a global network of payment might be used in two ways: first, using the neighborhood of suppliers, victims of impersonation, might allow for an early warning of potential future victims (client as well as suppliers). Fraud attempts, projected on such a network, might underline parts of the network where fraud might be more likely to occurs, and using label propagation algorithms such as the one underlined in [ZG02], the probability of a supplier's to fall prey to SIF might be computed. Furthermore, graph-based features such as centrality, betweeness and degree ([BVV15]) can be used to enrich the anomaly detection process, by integrating them into the feature vector in the pre-processing phase, as they represent the context in which the transaction occurs.

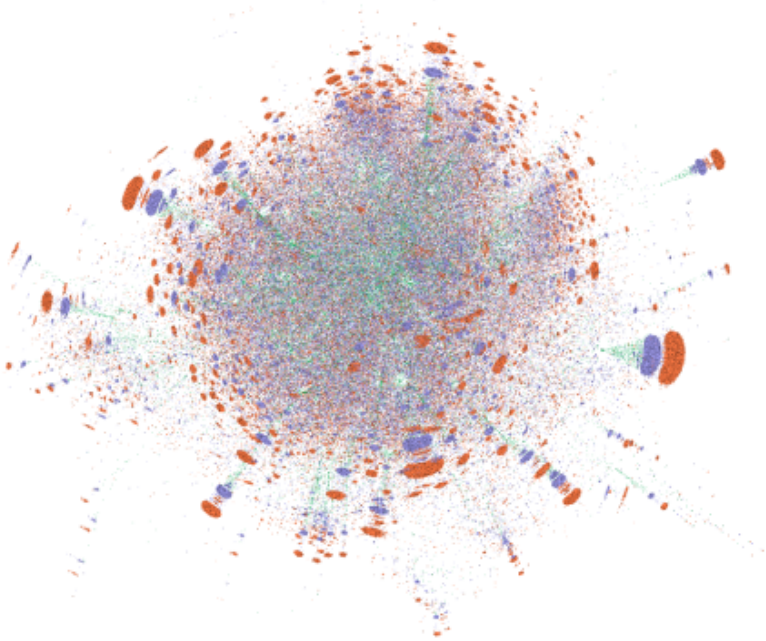

Figure 7.3: Example of global B2B payment network. Client companies are represented are green nodes, accounts as blue nodes and supplier companies as orange nodes.

Bibliography

[Abd+08] Michel Abdalla et al. “Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions”. In: *Journal of Cryptology* 21.3 (2008), pp. 350–391. ISSN: 09332790. DOI: 10.1007/s00145-007-9006-6.

[AMZ16] Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. “Fraud detection system: A survey”. In: *Journal of Network and Computer Applications* 68 (2016), pp. 90–113. ISSN: 10958592. DOI: 10.1016/j.jnca.2016.04.007.

[Abd07] Hervé Abdi. “Z-scores”. In: *Encyclopedia of measurement and statistics* 3 (2007), pp. 1055–1058.

[AHK01] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. “On the surprising behavior of distance metrics in high dimensional space”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 1973 (2001), pp. 420–434. ISSN: 16113349. DOI: 10.1007/3-540-44503-x_27.

[AIG19] AIG. *Impersonation Fraud Claims Scenarios*. \url{https://www.aig.com/content/dam/aig/america-canada/us/documents/business/management-liability/impersonation-fraud-claims-scenarios-brochure.pdf}. 2019.

[AMF10] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. “OddBall: Spotting anomalies in weighted graphs”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6119 LNAI.PART 2 (2010), pp. 410–421. ISSN: 03029743. DOI: 10.1007/978-3-642-13672-6_40. URL: http://www.cs.cmu.edu/%7B~%7Dlakoglu/pubs/OddBall%7B%5C_%7Dcameraready.pdf%7B%5C%%7D5Cnpapers://63650f89-9a27-4a9d-91f7-fca3c5048b3e/Paper/p1946.

[Apr17] Apruve. *7 Key Differences Between B2B Transactions and Consumer Purchases*. \url{https://blog.apruve.com/7-key-differences-between-b2b-transactions-and-consumer-purchases}. 2017.

[BVV15] Bart Baesens, Veronique Van Vlasselaer, and Wouter Verbeke. *Fraud analytics using descriptive, predictive, and social network techniques a guide to data science for fraud detection*. John Wiley & Sons, 2015.

[BGC17] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*. Vol. 1. MIT press Massachusetts, USA: 2017.

[BB12] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization". In: *Journal of machine learning research* 13.Feb (2012), pp. 281–305.

[BJ+06] David M Blei, Michael I Jordan, et al. "Variational inference for Dirichlet process mixtures". In: *Bayesian analysis* 1.1 (2006), pp. 121–143.

[BH+01] Richard J Bolton, David J Hand, et al. "Unsupervised profiling methods for fraud detection". In: *Credit scoring and credit control VII* (2001), pp. 235–255.

[Bol+02] Richard J. Bolton et al. "Statistical Fraud Detection: A Review". In: *Statistical Science* 17.3 (2002), pp. 235–255. ISSN: 08834237. DOI: 10.1214/ss/1042727940.

[Bos+15] Raphael Bost et al. "Machine Learning Classification over Encrypted Data". In: *Ndss '15* February (2015), pp. 1–31. ISSN: 03029743. DOI: 10.14722/ndss.2015.23241. URL: http://eprint.iacr.org/2014/331.pdf.

[Bro+02] Patrick L Brockett et al. "Fraud classification using principal component analysis of RIDITs". In: *Journal of Risk and insurance* 69.3 (2002), pp. 341–371.

[BXD06] Patrick L. Brockett, Xiaohua Xia, and Richard A. Derrig. "Using Kohonen's Self-Organizing Feature Map to Uncover Automobile Bodily Injury Claims Fraud". In: *The Journal of Risk and Insurance* 65.2 (2006), p. 245. ISSN: 00224367. DOI: 10.2307/253535.

[BCM+00] Stephen P Brooks, Edward A Catchpole, Byron JT Morgan, et al. "Bayesian animal survival estimation". In: *Statistical Science* 15.4 (2000), pp. 357–376.

[BSO09] Suratna Budalakoti, Ashok N. Srivastava, and Matthew E. Otey. "Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety". In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 39.1 (2009), pp. 101–113. ISSN: 10946977. DOI: 10.1109/TSMCC.2008.2007248.

[Bul04] John A. Bullinaria. "Self Organizing Maps: Fundamentals, Introduction to Neural Networks : Lecture 16". In: (2004), pp. 1–15.

[Bus19] BusinessDictionary. *financial transaction*. http://www.businessdictionary.com/definition/finan transaction.html. 2019.

[Can+18] Rémi Canillas et al. "Exploratory Study of Privacy Preserving Fraud Detection". In: 2018, pp. 25–31. ISBN: 9781450360166. DOI: 10.1145/3284028.3284032.

[Can+20] Rémi Canillas et al. "GraphSIF: analyzing flow of payments in a Business-to-Business network to detect supplier impersonation". In: *Applied Network Science* 5.1 (2020). ISSN: 23648228. DOI: 10.1007/s41109-020-00283-1.

[Can+a] Rémi Canillas et al. "Supplier Impersonation Fraud Detection in Business-To-Business Transaction Networks using Self-Organizing Maps". In: (), pp. 1–12.

[Can+b] Rémi Canillas et al. "Supplier Impersonation Fraud Detection using Self-Organizing Maps". In: ().

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly detection: A survey”. In: *ACM computing surveys (CSUR)* 41.3 (2009), p. 15.

[CMK08] Varun Chandola, Varun Mithal, and Vipin Kumar. “Comparative evaluation of anomaly detection techniques for sequence data”. In: *Proceedings - IEEE International Conference on Data Mining, ICDM* (2008), pp. 743–748. ISSN: 15504786. DOI: 10.1109/ICDM.2008.151.

[CC14] Jau Shien Chang and Wen Hsi Chang. “Analysis of fraudulent behavior strategies in online auctions for detecting latent fraudsters”. In: *Electronic Commerce Research and Applications* 13.2 (2014), pp. 79–97. ISSN: 15674223. DOI: 10.1016/j.elerap.2013.10.004. URL: http://dx.doi.org/10.1016/j.elerap.2013.10.004.

[CSN09] Aaron Clauset, Cosma Rohilla Shalizi, and M. E.J. Newman. “Power-law distributions in empirical data”. In: *SIAM Review* 51.4 (2009), pp. 661–703. ISSN: 00361445. DOI: 10.1137/070710111. arXiv: 0706.1062.

[Dal05] A. I. Dale. “Thomas bayes, an essay towards solving a problem in the doctrine of chances (1764)”. In: *Landmark Writings in Western Mathematics 1640-1940* 53.April 2003 (2005), pp. 199–207. DOI: 10.1016/B978-044450871-3/50096-6.

[Den+02] D. G.T. Denison et al. “Bayesian partition modelling”. In: *Computational Statistics and Data Analysis* 38.4 (2002), pp. 475–485. ISSN: 01679473. DOI: 10.1016/S0167-9473(01)00073-1.

[DFC19] Euler-Hermes DFCG. *Barometre Euler Hermes-DFCG 2019*. https: //www.eulerhermes.fr/actualites/ etude-fraude-2019.html. 2019.

[Die00] Thomas G Dietterich. “Ensemble methods in machine learning”. In: *International workshop on multiple classifier systems*. Springer. 2000, pp. 1–15.

[Dim20] Dimensions. *Dimension*. 2020. (Visited on 09/11/2020).

[Dom+16] Rémi Domingues et al. “An Application of Unsupervised Fraud Detection to Passenger Name Records”. In: *Proceedings - 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN-W 2016*. 2016, pp. 54–59. ISBN: 9781467388917. DOI: 10.1109/DSN-W.2016.21.

[Dom+18] Rémi Domingues et al. “A comparative evaluation of outlier detection algorithms: Experiments and analyses”. In: *Pattern Recognition* 74 (2018), pp. 406–421. ISSN: 00313203. DOI: 10.1016/j.patcog.2017.09.037.

[DH+73] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*. Vol. 3. Wiley New York, 1973.

[GHS+19] Mohamed Ahmed Galal, Wessam M Hussein, Mahmoud MA Sayed, et al. “Satellite battery fault detection using Naive Bayesian classifier”. In: *2019 IEEE Aerospace Conference*. IEEE. 2019, pp. 1–11.

[GS13] Andrew Gelman and Cosma Rohilla Shalizi. “Philosophy and the practice of Bayesian statistics”. In: *British Journal of Mathematical and Statistical Psychology* 66.1 (2013), pp. 8–38.

[Gey92] Charles J Geyer. “Practical markov chain monte carlo”. In: *Statistical science* (1992), pp. 473–483.

[Gia+20] Gabriele Gianini et al. “Managing a pool of rules for credit card fraud detection by a Game Theory based approach”. In: *Future Generation Computer Systems* 102 (2020), pp. 549–561. ISSN: 0167739X. DOI: 10.1016/j.future.2019.08.028. URL: https://doi.org/10.1016/j.future.2019.08.028.

[GAC18] Francisco Javier González-Serrano, Adrián Amor-Martín, and Jorge Casamayón-Antón. "Supervised machine learning using encrypted training data". In: *International Journal of Information Security* 17.4 (2018), pp. 365–377. ISSN: 16155270. DOI: 10.1007/s10207-017-0381-1. URL: http://link.springer.com/10.1007/s10207-017-0381-1.

[GF18] Palash Goyal and Emilio Ferrara. "Graph embedding techniques, applications, and performance: A survey". In: *Knowledge-Based Systems* 151 (2018), pp. 78–94. ISSN: 09507051. DOI: 10.1016/j.knosys.2018.03.022. arXiv: 1705.02801.

[Hoo+16] Bryan Hooi et al. "BIRDNEST: Bayesian inference for ratings-fraud detection". In: *16th SIAM International Conference on Data Mining 2016, SDM 2016* (2016), pp. 495–503. arXiv: arXiv:1511.06030v2.

[Inv17] Federal Bureau of Investigation. *Business E-mail Compromise, E-mail Account Compromise, the 5 Billion Dollar Scam.* \url{https://www.ic3.gov/media/2017/17050 2017.

[Ish00] Tsunenori Ishioka. "Extended K-means with an efficient estimation of the number of clusters". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 1983 (2000), pp. 17–22. ISSN: 16113349. DOI: 10.1007/3-540-44491-2_3.

[Jai10] Anil K Jain. "Data clustering: 50 years beyond K-means". In: *Pattern recognition letters* 31.8 (2010), pp. 651–666.

[JC14] Sachin Janbandhu and S M Chaware. "Survey on Data Mining with Privacy Preservation". In: 5.4 (2014), pp. 5279–5283.

[JK77] Norman Lloyd Johnson and Samuel Kotz. "Urn models and their application; an approach to modern discrete probability theory". In: (1977).

[JKB97] Norman Lloyd Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Discrete multivariate distributions*. Vol. 165. Wiley New York, 1997.

[Kap19] Dean Kaplan. *Advice From One B2B Collection Agency on How to Get Paid.* \url{https://articles.bplans.com/advice-from-one-b2b-collection-agency-on-how-to-get-paid/}. 2019.

[KW95] Robert E Kass and Larry Wasserman. "A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion". In: *Journal of the american statistical association* 90.431 (1995), pp. 928–934.

[KS12] Yoonseong Kim and So Young Sohn. "Stock fraud detection using peer group analysis". In: *Expert Systems with Applications* 39.10 (2012), pp. 8986–8992. ISSN: 09574174. DOI: 10.1016/j.eswa.2012.02.025. URL: http://dx.doi.org/10.1016/j.eswa.2012.02.025.

[KY04] Bryan Klimt and Yiming Yang. "The enron corpus: A new dataset for email classification research". In: *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)* 3201 (2004), pp. 217–226. ISSN: 03029743.

[Koh89] Teuvo Kohonen. "Self-learning musical grammar, or 'associative memory of the second kind'". In: (1989), pp. 1–5. DOI: 10.1109/ijcnn.1989.118552.

[KL18] John K. Kruschke and Torrin M. Liddell. "Bayesian data analysis for newcomers". In: *Psychonomic Bulletin and Review* 25.1 (2018), pp. 155–177. ISSN: 15315320. DOI: 10.3758/s13423-017-1272-1.

[LW+02] Andy Liaw, Matthew Wiener, et al. "Classification and regression by randomForest". In: *R news* 2.3 (2002), pp. 18–22.

[Lil16] Gary L. Lilien. “The B2B Knowledge Gap”. In: *International Journal of Research in Marketing* 33.3 (2016), pp. 543–556. ISSN: 01678116. DOI: 10.1016/j.ijresmar.2016.01.003.

[Liu11] Chuanhai Liu. *Maximum Likelihood Estimation From Incomplete Data via Em-Type Algorithms Bt - Advanced Medical Statistics*. Vol. 39. 1. 2011, pp. 1051–1071. ISBN: 978-981-02-4799-7. URL: http://www.worldscientific.com/doi/abs/10.1142/9789812388759_0028%5Cnpapers3://publication/doi/10.1142/9789812388759_0028.

[Luc+19] Yvan Lucas et al. “Multiple perspectives HMM-based feature engineering for credit card fraud detection”. In: *Proceedings of the ACM Symposium on Applied Computing* Part F147772 (2019), pp. 1359–1361. DOI: 10.1145/3297280.3297586. arXiv: 1905.06247.

[Min03] Thomas P. Minka. “Bayesian inference, entropy, and the multinomial distribution”. In: *Technical Report.* 2003.original 1998 (2003), pp. 1–11. URL: http://research.microsoft.com/en-us/um/people/minka/papers/minka-multinomial.pdf?ref=herseybedava.info&origin=publication_detail.

[MMA13] Mohd Izhan Mohd Yusoff, Ibrahim Mohamed, and Mohd Rizam Abu Bakar. “Improved expectation maximization algorithm for gaussian mixed model using the kernel method”. In: *Mathematical Problems in Engineering* 2013 (2013). ISSN: 1024123X. DOI: 10.1155/2013/757240.

[Mon+13] Misael Mongiovì et al. “NetSpot: Spotting significant anomalous regions on dynamic networks”. In: *Proceedings of the 2013 SIAM International Conference on Data Mining, SDM 2013* 2 (2013), pp. 28–36.

[MH04] Gregory Hagan Moulton and Felix Hamilton. *System and method for data protection with multidimensional parity*. US Patent 6,826,711. Nov. 2004.

[NWY19] Xuetong Niu, Li Wang, and Xulei Yang. “A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised”. In: (2019). arXiv: 1904.10604. URL: http://arxiv.org/abs/1904.10604.

[Pai99] Pascal Paillier. “Public-key cryptosystems based on composite degree residuosity classes”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 1592. 1999, pp. 223–238. ISBN: 3540658890. DOI: 10.1007/3-540-48910-X_16.

[PJ09] Hae-Sang Park and Chi-Hyuck Jun. “A simple and fast algorithm for K-medoids clustering”. In: *Expert systems with applications* 36.2 (2009), pp. 3336–3341.

[QX07] Quan Qian and Mingjun Xin. *Hidden Markov Model for System Call Anomaly Detection*. 2007. ISBN: 9783540715481. URL: http://link.springer.com/chapter/10.1007/978-3-540-71549-8%7B%5C_%7D17.

[Rab89] Lawrence R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. 1989. DOI: 10.1109/5.18626. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=18626.

[Ris+01] Irina Rish et al. “An empirical study of the naive Bayes classifier”. In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3. 22. 2001, pp. 41–46.

[RBL19] Ethan Roberts, Bruce A. Bassett, and Michelle Lochner. “Bayesian Anomaly Detection and Classification”. In: (2019). arXiv: 1902.08627. URL: http://arxiv.org/abs/1902.08627.

[Ryf+18] Theo Ryffel et al. *A generic framework for privacy preserving deep learning*. Tech. rep. 2018, pp. 1–5. arXiv: 1811.04017. URL: http://arxiv.org/abs/1811.04017.

[Slo+13] Neil JA Sloane et al. "The on-line encyclopedia of integer sequences". In: *Ann. Math. Inform* 41 (2013), pp. 219–234.

[Str19] Black Stratus. *5 Impacts a Data Breach Has on Your Business*. https://www.blackstratus.com/5 impacts-a-data-breach-has-on-your-business/. 2019.

[Swe02] Latanya Sweeney. "K-Anonymity: A Model for Protecting Privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570. ISSN: 0218-4885. DOI: 10.1142/S0218488502001648. arXiv: 10(5),2002;555-570. URL: http://www.worldscientific.com/doi/abs/10.1142/S0218488502001648.

[Tar+95] Lionel Tarassenko et al. "Novelty detection for the identification of masses in mammograms". In: (1995).

[Tru79] G. V. Trunk. "A Problem of Dimensionality: A Simple Example". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.3 (July 1979), pp. 306–307. ISSN: 1939-3539. DOI: 10.1109/TPAMI.1979.4766926.

[Van+14] Rens Van de Schoot et al. "A Gentle Introduction to Bayesian Analysis: Applications to Developmental Research". In: *Child Development* 85.3 (2014), pp. 842–860. ISSN: 14678624. DOI: 10.1111/cdev.12169.

[Van+15] Véronique Van Vlasselaer et al. "APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions". In: *Decision Support Systems* 75 (2015), pp. 38–48. ISSN: 01679236. DOI: 10.1016/j.dss.2015.04.013. URL: http://dx.doi.org/10.1016/j.dss.2015.04.013.

[Veu10] Thijs Veugen. "Comparing Encrypted Data". In: *Unpublished* (2010). URL: http://insy.ewi.tudelft.nl/sites/default/files/Comparing%20encrypted%20data.pdf.

[WM03] Takashi Washio and Hiroshi Motoda. "State of the art of graph-based data mining". In: *ACM SIGKDD Explorations Newsletter* 5.1 (2003), p. 59. ISSN: 19310145. DOI: 10.1145/959242.959249. arXiv: 1401.6981. URL: http://portal.acm.org/citation.cfm?doid=959242.959249.

[ZS06] Vladimir Zaslavsky and Anna Strizhak. "Credit card fraud detection using self-organizing maps". In: *Information and Security* 18 (2006), p. 48.

[ZG02] Xiaojin Zhur and Zoubin Ghahramanirh. "Learning from labeled and unlabeled data with label propagation". In: (2002).

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : CANILLAS DATE de SOUTENANCE :

Prénoms : Rémi

TITRE : Privacy and Security in a B2B environment: Focus on Supplier Impersonation Fraud Detection using Data Analysis

NATURE : Doctorat Numéro d'ordre : 2020LYSEI118

Ecole doctorale : ED512 Infomaths

Spécialité : Informatique

RESUME : La fraude au fournisseur (Supplier Impersonation Fraud, SIF) est un type de fraude se produisant dans un contexte Business-to-Business (B2B), où des entreprises et des commerces interagissent entre eux, plutôt qu'avec le consommateur. Une fraude au fournisseur est effectuée lorsqu'une entreprise (fournisseur) proposant des biens ou des services à une autre entreprise (client) a son identité usurpée par un fraudeur. Dans cette thèse, nous proposons, d'utiliser les techniques et outils récents en matière d'apprentissage machine (Machine Learning) afin de résoudre à ces différent points, en élaborant des systèmes de détection de fraudes se basant sur l'analyse de données.
Deux systèmes de détection de fraudes basés sur l'analyse de données sont proposés: ProbaSIF et GraphSIF. Ces deux systèmes se composent d'abord d'une phase d'entraînement où les transactions historiques sont utilisées pour calculer un modèle de données, puis d'une phase de test où la légitimité de chaque transaction considérée est déterminée.
ProbaSIF est un système de détection de fraudes au fournisseur qui se base sur un modèle bayésien (Dirichlet-Multinomial). ProbaSIF utilise la probabilité d'un compte en banque à être utilisé dans une transaction future d'une entreprise pour déterminer sa fiabilité.
GraphSIF, le second système de détection de fraude au fournisseur que nous proposons, a pour but d'analyser les propriétés relationnelles créées par l'échange de transactions entre une entreprise et ses fournisseurs. À cette fin, une séquence de différents graphes compilant tous les liens créés entre l'entreprise, ses fournisseurs, et les comptes en banque utilisés pour payer ces fournisseur, appelé séquence de comportement, est générée. Une transaction est catégorisée en l'ajoutant au graphe le plus récent de la séquence et en analysant les motifs formés, et en les comparant à ceux précédemment trouvés dans la séquence de comportement.
Ces deux systèmes sont comparés avec un jeu de données réelles afin d'examiner leur performances.

MOTS-CLÉS : Machine Learning, Fraud detection, Bayesian Inference, Supplier Impersonation Fraud, Graph mining, Unsupervised Machine Learning, Anomaly detection, Self-Organizing Maps

Laboratoire(s) de recherche : Liris

Directeur de thèse: Lionel BRUNIE

Président de jury :

Composition du jury :
- Mme Claudia RONCANCIO (claudia.roncancio@imag.fr), professeur (ENSIMAG Grenoble), rapporteur
- M. Gabriele GIANINI (gabriele.gianini@unimi.it), assistant professeur (UNIMI Milan), rapporteur
- M. Lionel BRUNIE (lionel.brunie@insa-lyon.fr), professeur (INSA-LYON Lyon), directeur de thèse
- M. Omar HASAN (omar.hasan@insa-lyon.fr), maître de conférence (INSA-LYON Lyon), co-directeur de thèse
- Mme Sara BOUCHENAK (sara.bouchenak@insa-lyon.fr), professeur (INSA-LYON Lyon), examninateur
- M. Laurent SARRAT (laurent.sarrat@sisnet.fr), CTO (SiS-id Lyon), examinateur
- M. Stelvio CIMATO (stelvio.cimato@unimi.it), assistant professeur (UNIMI Milan), examinateur
- M. Béchara AL BOUNA (bechara.albouna@upa.edu.lb), professeur (Antonine University Liban), examinateur