Advancing Anomaly Detection in Tabular Data : A Case-Study on Credit Card Fraud Identification

Hugo Thimonier

université PARIS-SACLAY

Advancing Anomaly Detection in Tabular Data: A Case-Study on Credit Card Fraud Identification

Faire Progresser la Détection d'Anomalies sur les Données Tabulaires : Étude de cas sur l'Identification des Fraudes au Paiement par Carte Bancaire

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, ED Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique
Graduate School : Informatique et sciences du numérique. Référent : CentraleSupélec

Thèse préparée dans l'unité de recherche **LISN (Université Paris-Saclay, CNRS)**, sous la direction de **Bich-Liên Doan**, Professeure à CentraleSupélec, le co-encadrement de **Fabrice Popineau**, Professeur à CentraleSupélec, et le co-encadrement d'**Arpad Rimmel**, Professeur Assistant à CentraleSupélec.

Thèse soutenue à Paris-Saclay, le 30 Septembre 2024, par

Hugo THIMONIER

Composition du jury

Membres du jury avec voix délibérative

Louise Travé-Massuyès Directrice de recherche, LAAS-CNRS & ANITI	Présidente
Alain Celisse Professeur des Universités, Université Paris 1 Panthéon-Sorbonne	Rapporteur & Examinateur
Marius Kloft Professor, RPTU Kaiserslautern-Landau	Rapporteur & Examinateur
Mazen Alamir Directeur de recherche, CNRS & Université de Grenoble	Examinateur
Gaël Varoquaux Directeur de recherche, Inria & Université Paris-Saclay	Examinateur

THESE DE DOCTORAT
NNT : 2024UPASG046

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Faire Progresser la Détection d'Anomalies sur les Données Tabulaires : Étude de cas sur l'Identification des Fraudes au Paiement par Carte Bancaire

Mots clés: Apprentissage Machine, Apprentissage Profond, Déséquilibre de Classes, Détection d'Anomalies, Données Tabulaires

Résumé: Les progrès récents dans le domaine de l'apprentissage automatique ont permis aux banques de s'appuyer sur ce type d'algorithmes pour renforcer leurs systèmes de détection de fraudes aux paiements par carte bancaire. Néanmoins, utiliser ces méthodes pour identifier les fraudes reste un défi en raison (i) du déséquilibre de classe et (ii) du changement de distribution. Les méthodes de détection d'anomalies (AD) apparaissent comme une solution potentielle en ce qu'elles sont insensibles à ces deux caractéristiques. Nous avons donc proposé deux nouvelles méthodes d'AD pour les données tabulaires. Puis, nous avons testé différentes méthodes d'AD sur un jeu de données de paiements par carte bancaire, et avons comparé leurs performances à celles des Gradient Boosted Decision Trees (GBDT). Nous observons un écart de performance significatif en faveur des GBDT, suggérant que les méthodes d'AD nécessitent davantage d'investigations pour concurrencer les GBDT pour la détection de fraudes.

Title: Advancing Anomaly Detection in Tabular Data: A Case-Study on Credit Card Fraud Identification

Keywords: Machine Learning, Deep Learning, Imbalanced Learning, Anomaly Detection, Tabular Data

Abstract: Recent advances in the field of Machine Learning has enabled banks to rely on this class of algorithms to build or augment their detection systems. Nevertheless, applying machine learning methods to identify frauds still remains challenging due to (i) the inherent imbalance in the available datasets and (ii) the possibility of distribution shift. Weakly-supervised anomaly detection (AD) methods appear as a possible solution as they should be robust to both challenges. In this work, we propose two novel weakly-supervised AD methods targeted for tabular data. We then test AD methods on a private online credit card payment dataset and compare their performance to Gradient Boosted Decision Trees (GBDT). We observe a significant performance gap between GBDT and AD methods, in favor of GBDT. Our experiments supports the idea that although promising, weakly-supervised AD method need further improvements to compete with GBDT for the task of fraud detection.

Contents

List of Figures

List of Tables

List of Algorithms

Hic sunt ~~dracones~~ fraudes.

Remerciements

Dans le cadre de mes travaux de recherche, je souhaiterais tout d'abord remercier chacun des membres du jury qui m'ont fait l'honneur d'évaluer mon travail. Je remercie également mes encadrants pour leur accompagnement et leur soutien tout au long de cette aventure. Je tiens aussi à remercier Lusis et la Chaire d'Intelligence Artificielle Lusis - CentraleSupélec "*Artificial intelligence applied to credit card fraud detection and automated trading*", ainsi qu'une grande banque française sans qui ce travail n'aurait pas pu voir le jour.

Lusis a proposé dans le cadre de sa chaire avec CentraleSupélec de travailler sur le problème de la détection de fraude aux paiements par carte bancaire. C'est un problème auquel tout le monde est confronté, mais dont le grand public sait peu de choses. Étant initié aux techniques d'apprentissage automatique, la perspective d'avoir accès à de grands jeux de données fournis par une grande banque française a piqué ma curiosité et je me suis embarqué dans ce sujet. L'expertise de Lusis à la fois sur les données de fraude et sur les techniques qui leur sont applicables m'a permis de faire progresser le volet applicatif particulièrement ardu de ce problème. L'ensemble des expériences réalisées au cours de ces 3 années de thèse ont pu l'être grâces aux ressources du "Mésocentre du Moulon » (http://mesocentre.centralesupelec.fr/) et de l'IDRIS via l'allocation GENCI n°101424 sur le supercalculateur Jean Zay du CNRS.

"Un chemin droit ne mène jamais qu'au but" - André Gide.

S'il n'est pas dans mes habitudes de m'épancher, ces 3 années (et quelques) de thèse, avec leur lot de joies et de déceptions, m'auront permis de discerner le bon grain de l'ivraie. *Laissez croître ensemble l'un et l'autre jusqu'à la moisson, et, à l'époque de la moisson, je dirai aux moissonneurs : Arrachez d'abord l'ivraie, et liez-la en gerbes pour la brûler, mais amassez le blé dans mon grenier.*[1]

Mes premiers remerciements ne peuvent aller qu'à ma famille, père, mère, frère et soeur[2], dont j'avais pu dire à la suite d'un voyage au Danemark, qu'il était heureux qu'on ne choisisse pas sa famille car tout le monde aurait voulu la mienne. Je ne crois pas que ces quelques lignes soient le lieu pour raconter des souvenirs personnels, néanmoins, je me permettrai ces quelques mots. Merci à mes parents qui m'ont transmis ces valeurs de travail et d'abnégation, et m'ont donné les moyens de poursuivre mes (courtes) études jusqu'à leur terme. Je pense que tout enfant rêverait d'être entouré, épaulé et aimé comme je le suis. Dix ans à Paris, mais mes racines me ramènent toujours à une certaine rue de Rouen.

Mon grand frère, ma petite soeur; Ma petite soeur, mon grand frère, impossible de vous dissocier tant notre fratrie ne s'est jamais distendue depuis la fin de nos adolescences respectives. Merci pour tout : les moments simples, l'aide, les discussions, les disputes, les voyages (j'ai un peu plus apprécié Dublin qu'Edimbourg, allez savoir pourquoi). C'est en observant les autres qu'on se rend compte qu'une famille et fratrie soudées comme la notre n'est pas (plus?) la norme. Nombre de souvenirs me viennent en mémoire nous concernant, mais je crois que les meilleurs moments à vos côtés sont aussi les plus simples: ces soirées d'été sur la terrasse à Busseaut, ces après-midi à jouer aux cartes le dimanche, ces balades dominicales étant plus jeunes. Pour tout ça, merci, merci et merci encore.

Merci à ma grand-mère qui m'aura inspiré autant qu'elle m'aura aimé, un grand esprit, un grand coeur, une rigueur et une modestie à toute épreuve. Merci pour ces mardis et jeudis rue Villebois-Mareuil, pour les framboises, moins pour le poisson, pour les glaces, les livres et le goût de la lecture. Merci à mon grand-père pour tous les moments de complicité, que ce soit autour d'un bon plat, de sport, tennis et football[3], ou de périgrinations à Amplepuis entrecoupées de "Bonjour Monsieur Thimonier" tous les dix mètres. Merci à mes oncles et tantes pour tous les moments passés avec eux, ces discussions animées, ces jeux endiablés, à Busseaut, à Amplepuis, au Ski, à Longvic, avenue Victor-Hugo, dans le Jura, à Royan, à Ré. Merci pour ces tartes aux prunes à Saint-Georges de Didonne et pour ces tartines au beurre et au nesquik à Busseaut.

Merci à mes cousins. Ce voyage à Rome restera parmi mes meilleurs souvenirs de ces dernières années. Attention à la trotinette électrique tout de même. L'équipe désorganisée reste encore à réunir.

Merci à toi pour ces réunions régulières dans le quartier latin, au Luco autour d'une partie d'échecs, dans un café pour philosopher, ou aux arènes pour jouer à la pétanque. De ce barbeau charcuté, en passant par ces cabanes construites ou parfois démolies, ces matchs de foots, ce jogging sous 1000°, Busseaut nous aura forgé et continuera de le faire.

Merci à toi pour ce voyage à Dublin, même si tu as triché au tarot (ou bien c'est mon frère?), et pour ces souvenirs de plage à Royan, à Amplepuis, à Dijon, à Busseaut (merci de t'être déplacé à chaque fois).

Merci à toi pour les discussions autour du cinéma, de la musique, de la photo, pour les cafés et discussions toujours profondes (pas la soeur de son frère pour rien) à Bordeaux quand on a pu s'y retrouver.

Après la famille de sang, vient la famille de coeur. En premier lieu je ne peux que remercier ma moitié qui m'aura accompagné jusqu'ici. De Paris XIII, en passant rue des Saint-Pères, puis Rue Simon Dereure et enfin *the place to be* Perreire. Depuis notre rencontre dans une certaine prairie, en passant par Prague, Milan, Busseaut, Dijon, et l'extérieur français, il n'y a pas un moment que je ne souhaiterais revivre. C'est une phrase assez pompeuse, mais les mots de Jean d'O sur les roses et les épines raisonnent parfaitement pour décrire ces années. Beaucoup de roses, très peu d'épines.

Les amis[4]. J'aurai un mot pour chacun, dans lesquels j'espère vous vous reconnaîtrez.

De l'ENS, en passant par l'ENSAE, la Nouvelle-Orléans, les soirées busseautines, le run après le déluge, les pétanques, ton téléphone dans les toilettes, le DM que tu m'as gentiment corrigé (non t'auras pas le fichier

[1] Matthieu, 13, 30.
[2] Random ordering.
[3] prononcé comme "bal" de bal populaire.
[4] Random ordering

`.tex`), le caprice après les trois (quatre?) As à la PBC, ou cette soirée mémorable chez Didoche pour travailler le projet Python. Merci Doc pour tous ces moments qui en annoncent d'autres. *SANTEEEE MARION!*

De l'ENS, à la professionnal school of AOEII and jeux de sociétés, en passant par le bâteau sur le lac du bourget ou le lac d'Hourtin, Bruxelles, les concerts (certains plus importants que d'autres *m'a bien compris*), les discussions philosophiques parfois sérieuses et toutes les autres carabistouilles qui ont fait cette amitié : La deuxième famille comme tu dis. Merci Doc (j'anticipe) pour tout ça, et pour la suite.

De l'ENS, à la maison de l'éco boulevard de l'hopital, en passant par Berlin, Zadar, le Plan B, l'Avenue René Coty et ses nombreux excès, un des plus beaux fous rires de ma vie (🍅), les deutchs dans le bistrot en attendant le cours de notre prof de micro préférée, le fait d'être là depuis lundi, le supercoin, et pour Bruxelles prochainement j'espère. Qu'est ce qu'on est bien avec *nos petites chaussures* : un grand merci Doc de la part de crispy noodles.

De l'ENS, en passant par la porte d'Orléans ou Issy-les-Moulineaux capitale du crime, merci pour ces moments d'inatention toujours fabuleux (évite de me *mute* pendant ma soutenance stp), pour ton abonnement malencontreux à un hebdomadaire douteux, ces tennis, ces runs, ces souvenirs autour du foot, en premier lieu un certain ASM-City, nos discussions parfois profondes qui mettent souvent en lumière nos nombreux accords et rares désaccords. Un litron de Délirium Octave? *T'emballe pas Nénesse*.

De PSE, aux quais de Seine où on a parlé pour la première de l'AJA, les nombreuses soirées DM à Convention, les Punk IPAs, les nombreuses parties de Fifa, scratching sur mes vinyles que j'ai ruinés, soirée Concrete, à la PBC, détours dans des frips, le selfie avec un chien à Bagnolet en uniforme chemise BDP-wayfarer-style de malade, les ronflements des enfers à Berlin, le festival plus que mémorable dans les tréfonds de la Normandie avec notre tente 18 places, "allo allo", et la veste déchirée aux abords de la gare, quel fou rire. Merci Doc pour tous ces moments, et pour tous ceux à venir, de la part de crispy noodles.

Du club de tennis de Fontaine-lès-Dijon où l'on s'est rencontré sans le savoir, à la Rustine où notre amitié a commencé, en passant par mon garage, la maison fontainoise appartenant à une star montante du cinéma africain, Montigny-Montfort (à lire avec un accent bourguignon), Busseaut, Prague (cette ville belle et vivante), ce festival en Normandie, Lille, le Jura, probablement d'autres destinations halieutiques à venir (mais sans le coach), merci pour ces soirées à swinguer sur de la funk/disco, pour ces runs, ces tennis, la découverte de la pêche, ces fous rires, ces gdb après un détour par RosMartinos del gatos avec les patatas fritas. Attention en vélo tout de même, tu m'avais fait tomber en rentrant de la péniche. *Chistoooole!*

De Foncine-le-Haut, où on a pu manger dans un étoilé michelin où les *gens sont biens*, au saint chalet, à Busseaut, Plombières, Velars, aux abords de la Seine, de l'Ain, de la Saine, de la Coquin'z, et probablement d'autres rivières encore, merci Doc pour ces discussions de chercheur à chercheur, ces moments d'insécurité dans la voiture du bébouh, pour mon album plein de photos de TES poissons, merci pour ça et pour la suite. Dis à tes collègues de faire un effort sur les trains stp. *Ouais, contact ouais.*

Du magistère des enfers où un certain D. A. nous aura permis de nous rencontrer, au stade de Chambly (rip ta cheville), en passant par la gazette de Cassel, *la putain de BU* en mangeant une tartiflette pas cuite, les soirées infinies à jouer à Fifa avec 8 cartons rouges, le Parc des Princes en parcage où je n'ai jamais vu que des défaites, le penalty re-tiré par Ibra qui aura brisé nos rêves de chambreurs fous (j'ai encore les copies de page écran que je re-regarde régulièrement), l'histoire de la douche qui aura traumatisé des générations entières de forcalquiérens, merci pour tout ça et tout le reste. *Allez le foot!*

Du magistère des enfers, en passant par le café du croissant (big up à Jaurès), Nantes capitale du crime (pas une blague cette fois ci), à ce circuit de karting où tu auras chock au dernier moment, cette baffe mémorable, ces discussions autour de l'Histoire, de la numismatie, du foot, des ultras, des échecs (de la dérouillée que je t'ai mise la dernière fois), de la dérouillée qu'on aura pris à la Beaujoire (content d'avoir fait le déplacement), en passant par les matchs chez ton frangins, merci! *Ecoutez ça ne fait rien, je vous garde quand même*.

Du magistère des enfers, en passant par Busseaut, Castelbrando, malheureusement pas Milan, Paris XVIIlème (*j'ai mis un TSR 18ème*), le XIIIème, le château où on aura appris l'esperanto et bu un poil trop d'absinthe, la PBC ou le brewberry, nos discussions foots endiablées : merci! (Et allez l'OL).

D'une ligue MPG qui nous amena à nous retrouver à Vezelay pour péleriner en direction de Saint-Jacques de Compostelle, voyage pendant lequel on aura connu de sacrées aventures, du portail d'énergie à la soupe d'ortie soupoudrée de commentaire complotistes, en passant par le tractage refusé à Issoudun, l'attaque par le doberman, la maison des enfers comme premier refuge, la rencontre avec le pélerin parisien, la discussion avec notre hôte dans la caravane ou encore l'arrivée à Bourges qui aura confirmé notre vision du monde. Pour

nos discussions historiques, nos débats politiques, philosophiques, et tout le reste : merci.

Pour le reste de la team MPG qui est devenue tout sauf MPG, merci pour le milliard de running gags qui perdurent maintenant depuis n années : incroyable, il en a rien à foutre, $12,5\%$, instants magiques, le gif de Yannis, le maquillage de vache et j'en passe. Une amitié née grâce à un petit bonhomme tout poilu, et je l'en remercie.

De Saint-Ouen et Clichy, en passant par le luco et tout un tas de bars parisiens, un stage qui aura fait naître une réelle amitié : de manager à pote il n'y a qu'un pas, de manager à ami il y en a 1000 et je crois qu'on les a fait. De discussions sur la F1, le jeu d'échec, notre vision de la vie, du travail, tous nos points d'accord ne pouvait que mener à une réelle amitié. *Amitié en 4, trait au blanc*.

De l'école maternelle Montchapet, en passant par le club de foot de Fontaine-lès-Dijon, le collège Roupnel, la rue Adolphe Dietrich (*Mais où est-il?*), Beauséjour (*chauds chauds les marrons*), 25 ans d'une amitié qui n'a jamais failli. De guitar hero, au cocavelo, en passant par les sessions photo, les sirops place de la lib, les semaines à Busseaut, les bricolages plus qu'approximatifs de biclous, on aura bien rigolé. Merci pour tout ça! *J'rap avec mon backpack, putain d'sac à dos*.

De Roupnel, au Beausej', en passant par l'Univers, le Saint Nic', le garage, de la bave de saint Balou, au pic dans le torse, en passant par le Lounge et ses shots épicés, les tentatives de te mettre à la course à pied à la Combe à la Serpent qui ont mis ton cardio à rude épreuve, les discussions autours de la musique, ta lettre Adopi pour avoir téléchargé 20000Go de musique des Beatles, que de souvenirs. Merci pour tout ça, et pour ce qui s'annonce! *Dis moi tes Ben Simon, ils font les mêmes pour homme ou pas?*

Du Lycée International Charles de Gaulle, en passant par des débats en tous genres, surtout musicaux, sur Facebook (on était des fanboys, on peut se le dire maintenant, p.s. Eminem>Cudi), les semaines à Busseaut, notamment celle où tu m'as rasé le crâne avant ma rentrée à l'ENS, les nombreuses soirées de médecins (désolé pour la chute dans les escaliers de la triloc), les après-midi à ne pas faire grand chose, à refaire le monde et fifater en même temps, les week-ends à Marseille qui m'ont parfois donné des bouffées d'air pendant cette thèse parfois intense – je rentrais fatigué physiquement mais reposé mentalement – merci pour tout ça et pour ton amitié. Hâte de faire la course dans les calanques avec nos Porsches respectives. *The heart of a Lion*.

De nos vacances en Crête qui ont fait naître une réelle amitié, en passant par le Lycée International Charles de Gaulle, le garage (*C'est une menace?*), l'appart des parents (rip l'enduit sur le mur). Certaines amitiés, comme la notre, sont comme ces plantes qui n'ont pas besoin d'être arrosées tous les jours, mais qui ont plutôt besoin, de temps à autres, d'une bonne grosse dose d'eau d'un coup. Merci pour ça, et pour la suite! Le bonjour au *Pape des Escargots*!

Du lycée Internationale Charles de Gaulle où tu lançais des boules de neige, – *Gamin va!* – en passant par l'ile d'Yeu (meilleures vacances, *qu'on m'arrête à la sortie si ce que je dis n'est pas vrai*), le Morvan et sa cornemuse, Busseaut, la Paris Beer Week, Marseille bébé, que de souvenirs : cette daurade au barbecue, ces soirées *trivial pursuit*, ces truites loupées et parfois sorties, ces moments après la pêche qui sont souvent plus beaux encore que la pêche elle-même (il me revient en mémoire une mousse à Montmoyen après avoir sorti 18 sevens), ces nombreux débats qui n'en étaient pas puisque que nous étions toujours d'accords, et ces moments *yoyoyo*. Merci pour tout ça, et pour ce qui vient. *Attention Jocelyn, il peut y avoir toutes formes de blessures, de brûlures!*

De l'école primaire Montchapet et ton jeu de *Questions pour un Champion*, de l'*imbecile*[5] sorti si naturellement en cours d'anglais, en passant par le téléphone explosé d'énervement au skatepark, ou les nombreuses conversations en visio à refaire la vie en jouant à Fifa, les après-midi piscine du côté d'Etaule, la soirée avec le glaçon Mickey, le match Thimonier vs Th. pendant la sortie vélo de CM2 (on vous a explosé), que de souvenirs aussi : merci pour tout ça! *The best offer* c'est sans doute le lys et non pas le compas et l'équerre.

Du lycée International Charles de Gaulle, mais surtout au lycée Eiffel où la prépa nous aura permis de nouer une amitié solide dont naquirent de mémorables souvenirs. Sans mon binôme de prépa, ca aurait été quand même vachement moins marrant : la PPC sur Charlie the Unicorn (sans doute le plus gros fou rire de ma vie), en passant par les imitations douteuses de certains de nos profs (*inverse* au risque tu connais), 512 - Banque, *n'impotekomen*, et toutes les soirées dégustations de bières qu'on aura pu faire, et ces soirées parisiennes trop peu nombreuses finalement. Hâte de venir te voir dans ta contrée à la feuille d'érable. Merci pour tout!

A ma famille de sang, et de coeur, mille mercis !

[5] à lire avec un accent anglais

Chapter 1

Introduction

1.1 Credit Card Fraud Detection

The credit card's history spans over a century and involves various derivatives of what is now designated by the term *credit card*. The first ever credit card is often considered the Diners Club Card, which appeared in the 1950's and was introduced by Franck McNamara. The story goes that Franck McNamara had forgotten his wallet while dining out in a restaurant with his wife and faced embarrassment, which he promised himself never to live again. He created a Diners Club card, allowing customers to dine without cash.

Later on, American Express and BankAmericard were introduced by American Express and Bank of America, respectively, which extended the concept of credit beyond just dining. These first credit cards highly resemble what we currently have as credit cards in the 21^{st} century. Indeed, after the introduction of magnetic stripes and electronic authorization systems in the 1970s, transactions were made faster, facilitating the spread of credit cards among households. Later, in the 1980s and 1990s, the deregulation of the banking industry led to a surge in competition that contributed to significant innovations in the credit card market, such as airline miles or cashback. Finally, in the 21^{st} century, the rise of the internet and e-commerce modified how people consumed and used credit cards by significantly increasing online transactions. Questions regarding fraud and its detection began to appear as previous paradigms only enabled fraud to be committed in real life. At the same time, online transactions made committing fraud easier thanks to social engineering or information theft. Hence, banks introduced in the 2000s chip cards, also known as EMV cards, aiming at enhancing security and reducing fraud as much as possible.

Frauds[1] not only induce a significant financial impact that makes detecting them necessary but are also related to organized crime. Indeed, while lone fraudsters exist, they do not represent the majority of fraudsters and represent a neglectable share of the induced cost. On the contrary, extensive and well-organized groups have multiplied and represent the main threat to financial institutions. Those groups are often linked to other criminal activities and partially finance these activities through fraud on credit card payments. In that sense, reducing the share of fraud in the financial system by detecting them has a financial, political, and social impact.

[1] Throughout this manuscript, the term *fraud* will refer to credit card fraud.

1.1.1 Rule-based Models

Nevertheless, a significant surge in online fraud accompanied the growing number of online transactions and led banks to create fraud detection systems. The first systems consisted of human supervision flagging suspicious payments, but quickly, banks turned to algorithms to automate such detection. Until recently, banks and financial institutions that aimed at detecting fraud in credit card payments relied on simple yet relatively effective approaches such as rule-based models. These types of models appeared as particularly appealing for this task since they satisfy two objectives associated with *efficiently* detecting frauds in credit card payments: (i) being interpretable, while (ii) keeping the *false positive rate* low.

Interpretability Typical rule-based models are considered interpretable since they can be represented by simple binary trees with logical relations on features based on expert knowledge. A simple rule to flag fraud may resemble a logical statement as (amount > some amount) $\wedge$ (time > some time of the day) $\wedge$ (country == some country) then payment is a fraud. By construction, such rules allow explicitly identifying what caused a payment to be rejected and flagged as fraudulent. This is often described as *interpretability* because the model produces human-understandable information that can account for its prediction. This is in opposition to *black box* models that make predictions without the possibility of formulating the computation that led to its prediction in the form of a logical statement that a human can understand.

False positive rate Fraud detection is a binary classification application in which one infers from the characteristics of a payment whether this payment is fraudulent or honest. In inference, once an algorithm classifies a sample as belonging to the positive class (fraud) or the negative class (legit), one can face four possible outcomes:

(a) True positive (TP): The sample classified as a fraud is, indeed, a fraud.

(b) False positive (FP): The sample classified as a fraud is, in fact, legit.

(c) True negative (TN): The sample classified as legit is indeed legit.

(d) False negative (FN): The sample classified as legit is, in fact, a fraud.

From the point of view of a bank, the misclassification of a payment has a very different impact depending on whether it is a *false positive* or a *false negative*. The former impacts its reputation as a whole, while the latter has a more concrete financial impact. Indeed, when failing to flag actual frauds (false negative cases), a customer will suffer financial loss and the practical cost of changing credit cards. Nevertheless, the financial cost is the responsibility of the banks since they are obligated to reimburse (sometimes with a franchise) the amount customers lose due to fraudulent transactions. Thus, failing to flag fraud leads to a significant financial loss to the banks. Parallel to this, when flagging a legit payment by the cardholder as a fraud (false positive case) and thus canceling the payment, the bank may face discontent from the customer due to its transaction being canceled.

Financial institutions have often set the minimization of the false positive rate as an objective to minimize customer dissatisfaction. One can consider rule-based models well suited for such an objective since they flag frauds based on rules humans have constructed. To construct these rules, banks solely rely on past observations and what they have explicitly identified as fraudulent payments.

Thus, these models can never detect fraudulent behaviors that were never seen before. Hence, these rule-based models tend to only partially satisfy the performance objective regarding fraud detection that banks have set.

Performance Overall, as mentioned in the previous paragraph, these rule-based models are precise but unable to flag frauds humans have not already seen and formalized to create the rule used in the models. Moreover, banks often set performance objectives for their fraud detection models in terms of two metrics: precision and recall. Let us define both metrics as functions of TP, FP, TN, and FN.

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{recall} = \frac{\text{FP}}{\text{FP} + \text{FN}} \tag{1}$$

The ideal fraud detection system obtains both a high precision and a high recall. However, given the difficulty of the task, banks are often faced with an arbitrage:

- High precision but low recall: Most payments flagged as frauds are indeed frauds, but a small share of the total frauds is detected.
- High recall but low precision: a high share of the total frauds are detected but at the cost of having a lot of legit payments wrongly flagged as frauds.

The standard approach is achieving a precision level above a threshold, precision $\geq \tau$, while maximizing the recall. Formally, the bank aims at finding the detection system f in the set of considered systems $\mathscr{F}$, using the following program

$$\begin{aligned} \max_{f \in \mathscr{F}} \quad & \text{recall}(f) \\ \text{s.t.} \quad & \text{precision}(f) \geq \tau, \end{aligned} \tag{2}$$

where the bank defines τ according to its objectives.

Given this objective and the weak capacity of rule-based models to achieve a high recall, banks have recently turned to machine learning models to try to reach their fraud detection objectives. Formally, this consists in choosing a different set of considered systems $\mathscr{F}$ to choose f as described in equation (2).

1.1.2 Machine Learning for Fraud Detection

Machine Learning (ML) has become ubiquitous in various applications. Indeed, since its earliest models including the perceptron (Rosenblatt, 1958) or the nearest neighbors algorithm (Fix & Hodges, 1989), growing interest and an expanding research community have allowed machine learning models to become stronger on a broad range of tasks such as medical image segmentation (You et al., 2022), detecting solar panels on images (Kasmi et al., 2022), generating text (Touvron et al., 2023) or images (Rombach et al., 2022) and many more.

ML is a branch of artificial intelligence that seeks to build algorithms that learn from data and make predictions or decisions without being explicitly programmed. At its core, machine learning aims to uncover patterns within datasets, enabling systems to improve their performance over time. There are several approaches to machine learning, with supervised learning being one of the most common. In supervised learning, the algorithm learns from labeled data, where each input is associated with a corresponding output. In other words, labeled data corresponds to a set of examples

composed of a sample's characteristics and its corresponding class, called the label, e.g., a picture of a dog and its label: `dog`. Supervised learning allows the model to predict new, unseen data based on prior training. On the other hand, unsupervised learning involves extracting patterns from unlabeled data, seeking to find hidden structures or relationships within the dataset without explicit guidance. Unsupervised learning is often used for clustering, where each sample is associated with other samples from the dataset based on a measure of similarity. Self-supervised learning is a branch of unsupervised learning where the model creates its supervision signal from the input data. The word "*self*" in self-supervised learning originates from the fact that this approach involves constructing artificial labels from possibly unlabeled data or simply ignoring the existing label. A typical example named masked reconstruction consists in predicting missing input parts from the unmasked counterpart. Self-supervised learning is often used in a *pre-training* phase to foster the performance of a supervised classifier by transforming a sample from the original dataspace to obtain a representation that facilitates separability between classes.

ML can be applied to different data modalities. One usually considers two main data categories: structured data and unstructured data. On the one hand, *structured data* is characterized by its organization into a predefined format, making it easily searchable and analyzable. An example of structured data is tabular data, commonly stored in databases or spreadsheets. Each entry in a table is organized into rows and columns, where columns represent variables or features, and rows represent individual records or instances. This format facilitates efficient data management, querying, and analysis due to its explicit schema and the consistency of data types across the dataset. *Unstructured data*, on the other hand, lacks a predefined structure and is often heterogeneous in nature. This category includes text documents, images, audio files, and videos. Unlike structured data, unstructured data does not fit neatly into rows and columns.

ML applied to fraud detection would fall in the category of supervised learning, in which a *classifier* will learn from a labeled dataset composed of legitimate and fraudulent payments. A supervised classifier is a decision maker that "decides" based on past examples, the class to which a sample belongs, given its characteristics. In the fraud detection application, a classifier would learn to identify patterns from observing the characteristics of payments, *e.g.* its amount, location, currency, time, and its associated label, *i.e.* whether a payment is legit or a fraud, and formulate a decision on whether a payment is legit.

However, using ML methods for fraud detection is far from trivial as this application displays class imbalance and distribution shift. First, the ideal setting for supervised ML algorithms to learn from a dataset involves a **balanced setting**. In the case of binary classification, this involves a setting where both classes have more or less an equal number of samples in the dataset used to train the algorithm. Minor discrepancies between the cardinality of each class can be mitigated using preprocessing techniques such as oversampling or undersampling, as later detailed in section 2. However, when a severe imbalance exists between the two classes, these methods struggle to improve classification performance significantly. This statement is particularly true for structured tabular data, on which imbalance-correction techniques are challenging to apply. Unfortunately, fraud detection involves a highly skewed dataset that is comprised of a negative class (legit payments) that vastly outnumbers the positive class. This situation is often referred to as *extreme imbalance*. Indeed, real-life credit card payment datasets often involve legit payments representing more than 99% of the samples in the dataset. This represents the first challenge involved with applying ML models for fraud

detection. Second, one of the core hypotheses underlying ML is that the distribution that generated the dataset from which the model learns also generates the samples on which the model will make its predictions in inference. In the case of fraud detection, this simply implies that samples of legit payment and fraud behaviors remain constant over time concerning the behaviors observed in the training set used to train the ML models. This hypothesis is often invalidated in real-life scenarios, as discussed in section 1.2 and in chapter 5.

Hence, applying standard supervised machine learning for fraud detection applications appears particularly challenging and may require investigating ad-hoc techniques or specific classes of ML algorithms. In the next section, we formally discuss the concepts introduced in this section.

1.2 Theoretical overview

This section briefly overviews supervised machine learning and then discusses how fraud detection deviates from the standard setting.

1.2.1 Machine learning in the standard setting

Depending on the data available and the target objective, Machine Learning (ML) algorithms can be categorized into several subbranches, such as supervised learning or unsupervised learning. Supervised learning involves a training set composed of samples $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ is the features of samples i and $y_i \in \mathcal{Y}$ its label which designates to which class the sample belongs. The goal of supervised learning is to estimate a mapping from the features space to the label space; in other words, estimate the class of a sample from its features. Unsupervised learning involves a training set containing unlabeled samples to characterize the underlying data distribution.

Formally, supervised classification consists in estimating a function $f : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X} \subseteq \mathbb{R}^d$ describes a d-dimensional feature space, and $\mathcal{Y}$ is the label space (*e.g.* $\mathcal{Y} = \{0, 1\}$ in the binary classification case). As mentioned in the previous paragraph, one usually disposes of a training set composed of labeled samples,

$$\mathcal{D}_{train} = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^{n}, \tag{3}$$

to estimate the function f. The key hypothesis behind this type of approach is that the samples contained in the training set are *i.i.d.* (independent and identically distributed) samples from an unknown distribution P,

$$(\mathbf{x}_i, y_i) \overset{iid}{\sim} \mathrm{P}. \tag{4}$$

A machine learning algorithm outputs the "*best*" classifier f within an estimator set $\mathcal{F}(\mathcal{X}, \mathcal{Y})$.

Let $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ designate the loss function, $\ell(y, y')$ measures the loss incurred by predicting y' when the true label is y. The quality of a classifier is generally quantified by its risk

$$\mathrm{R}_{\mathrm{P}}(f) = \mathbb{E}_{\mathrm{P}}[\ell(y, f(\mathbf{x}))]. \tag{5}$$

The "*best*" classifier is then defined as the classifier that minimizes the risk over all classifiers in $\mathcal{F}(\mathcal{X}, \mathcal{Y})$. Formally,

$$f^* = \arg \min_{f \in \mathcal{F}(\mathcal{X}, \mathcal{Y})} \mathrm{R}_{\mathrm{P}}(f), \tag{6}$$

where the best classifier f^* is often called the Bayes classifier or the oracle. Since $\mathrm{R_P}(f)$ depends on the unknown data distribution P, it is intractable, and so is f^*. However, $\mathrm{R_P}(f)$ can be estimated by its empirical equivalent expressed as,

$$\hat{\mathrm{R}}_n(f) = \frac{1}{n} \sum_{(\mathbf{x}_i, y_i) \in \mathscr{D}_{train}} \ell(y_i, f(\mathbf{x}_i)).$$

One then chooses as a classifier the function $\hat{f}_n \in \mathscr{F}(\mathscr{X}, \mathscr{Y})$ through Empirical Risk Minimization (ERM),

$$\hat{f}_n = \arg \min_{f \in \mathscr{F}(\mathscr{X}, \mathscr{Y})} \hat{\mathrm{R}}_n(f), \tag{7}$$

This general setting can be applied to a broad range of tasks. However, as discussed in section 1.1.2, fraud detection stands out from standard applications and displays two main characteristics that make matters more complex: imbalanced classes and distribution shift.

1.2.2 Imbalanced Learning

The case of fraud detection on credit card payment is often considered to be a very challenging task (Wei et al., 2013) since it displays an extreme imbalance between classes (Krawczyk, 2016): the majority class often represents more than 99% of the dataset.

ERM, as defined in equation (7), performs well in situations where the average performance is an appropriate surrogate for the objective task. However, in applications like fraud detection, where the training set displays imbalanced classes, ERM usually performs poorly. Indeed, the estimator defined in equation (7) will favor samples in the majority class at the expense of the minority class. This is especially detrimental when false-positive samples need to be avoided, as in the case of fraud detection.

Scholars have investigated the consequence of imbalanced learning on many canonical classifiers and have shown that most standard classifiers need to be adapted to imbalanced settings. For instance, the limitations of standard machine learning techniques when confronted with imbalanced datasets are examined comprehensively by Yanmin et al., 2011. The study also sheds light on the struggles faced by backpropagation algorithms in converging within imbalanced setups, as the dominant majority class can overwhelm the gradient vector used for weight updates in neural networks as supported by (Carvajal et al., 2004; Japkowicz & Stephen, 2002). More specifically, (Akbani et al., 2004; Wu & Chang, 2003) demonstrate that SVMs can suffer significant performance deterioration in highly skewed class distributions. Similarly, Estabrooks et al., 2004; Weiss and Provost, 2001 reveal the inefficiency of standard machine learning algorithms like random forests on severely imbalanced datasets. Additionally, Zhang and Mani, 2003 highlight the failure of k-nearest neighbors for skewed class distributions, as the sparsity of minority class samples makes their correct classification challenging.

Given this diagnostic, researchers have investigated techniques to tackle the problem of imbalanced learning. According to Krawczyk, 2016, there are three main approaches to addressing the problem of learning from imbalanced data for supervised tasks:

- *Data-level methods*: modify the dataset to balance distributions.
- *Algorithm-level methods*: methods that modify an existing algorithm to adapt it to the skewed

distribution setting.

- *Hybrid methods*: methods that combine both previous approaches.

We further discuss these methods in section 2. Nevertheless, the literature has shown that most methods are proposed for a general setting and struggle with structured data.

1.2.3 Distribution Shift

A central hypothesis for machine learning approaches to work is that the distribution between the observed samples in the training set and those observed in the test set are identical. Indeed, when this is the case $\hat{\mathrm{R}}_n(f)$ displays interesting statistical properties. Under the hypothesis that $\mathbb{E}_{\mathrm{P}}[\ell(y, f(\mathbf{x}))^2] < +\infty$, then the central limit theorem and strong law of large numbers give that

$$\hat{\mathrm{R}}_n(f) \xrightarrow[n\to+\infty]{a.s.} \mathrm{R}_{\mathrm{P}}(f), \qquad \sqrt{n}\left\{\hat{\mathrm{R}}_n(f) - \mathrm{R}_{\mathrm{P}}(f)\right\} \xrightarrow[n\to+\infty]{\mathscr{D}} \mathcal{N}(0, \mathbb{V}[\ell(y, f(\mathbf{x})]).$$

However, when the distribution of the samples in the training set differs from the general distribution, those properties no longer hold. For instance, if equation (4) still holds for the training set, but the general distribution of data $\tilde{\mathrm{P}}$ differs from P, then there are no guarantees regarding the empirical risk $\hat{\mathrm{R}}_n(f)$ and its capacity to estimate the risk of a classifier correctly. This can happen in the case of a distribution shift, which describes a sudden or progressive shift in the data distribution generating the samples.

The literature (Moreno-Torres et al., 2012; Quiñonero-Candela, 2009) has revealed different types of distribution shift, among which the most important are covariate shift, prior probability shift, and concept shift. Let us denote $\mathrm{P}(\mathbf{x}, y) = p(\mathbf{x} \mid y)p(y) = p(y \mid \mathbf{x})p(\mathbf{x})$ and $\tilde{\mathrm{P}}(\mathbf{x}, y) = \tilde{p}(\mathbf{x} \mid y)\tilde{p}(y) = \tilde{p}(y \mid \mathbf{x})\tilde{p}(\mathbf{x})$, the training set distribution and general distribution respectively. The hypothesis for machine learning to generalize outside the training set is that $\tilde{\mathrm{P}} = \mathrm{P}$.

When **covariate shift** occurs, the distribution of x changes between the training and test sets. In other words, covariate shift designates a situation where $p(\mathbf{x}) \neq \tilde{p}(\mathbf{x})$ while the conditional distribution of y given $\mathbf{x}$ remains unchanged, $p(y \mid \mathbf{x}) = \tilde{p}(y \mid \mathbf{x})$. This can happen when the training and test sets samples are drawn from different distributions. In the case of credit card fraud detection, this could happen due to changes in customer behavior, economic factors, or technological advancements. For instance, imagine that during the training phase, most credit card transactions were made using traditional payment methods. However, during the testing phase, there was a significant increase in mobile payment transactions. As a result, the input features (e.g., transaction time, transaction amount, location) may have different distributions, leading to a covariate shift.

Prior **probability shift**, on the contrary, occurs when the distribution of the label y is different between the training set and the test set, $p(y) \neq \tilde{p}(y)$, but the conditional distribution of $\mathbf{x}$ given y remains the same, $p(\mathbf{x} \mid y) = \tilde{p}(\mathbf{x} \mid y)$. This can happen when the distribution of the label changes between the training and test set; in the case of credit card payment, a prior probability shift might change the proportion of fraud between the training and test set. Let us say that during the training phase, the prevalence of credit card fraud was relatively low. However, during the testing phase, a sudden surge in fraudulent activities occurs due to a data breach at a major retailer. This results in a shift in the prior probabilities of fraud and non-fraud cases.

Finally, **concept shift** describes two types of situations. First, it encompasses the cases where

the distribution of the features remains the same, $p(\mathbf{x}) = \tilde{p}(\mathbf{x})$ but the conditional distribution of y given $\mathbf{x}$ differs between the training set and test set, $p(y \mid x) \neq \tilde{p}(y \mid \mathbf{x})$. Second, it also includes its dual where the label distribution stays the same $p(y) = \tilde{p}(y)$ but the conditional distribution of $\mathbf{x}$ given y has changed, $p(\mathbf{x} \mid y) \neq \tilde{p}(\mathbf{x} \mid y)$. In the context of frauds, concept shift could translate into new types of frauds appearing in the test phase, which were not included in the training phase. Concept shift is the most detrimental distribution shift for supervised machine learning models since it directly involves a modification of the relation between the input features and the target variable. However, if not appropriately addressed, all three types will likely negatively affect machine learning models' performance.

Given this description, desirable machine learning models need to be robust to distribution shift. The literature of domain generalization introduced in Blanchard et al., 2011 focuses on producing algorithms that generalize well out-of-distribution and are thus robust to distribution shift. Zhou et al., 2022 extensively discuss the literature on domain generalization and categorize domain generalization methods into several groups. However, most methods in the domain generalization field have focused on computer vision problems and cannot be adapted to structured data and fraud detection.

Overall, standard remedies to imbalanced learning and distribution shifts prove less effective on tabular data. Moreover, combining both methods multiplies the possibilities of imprecision and mistakes. Therefore, this calls for methods that are robust to both imbalanced classes and distribution shifts. A possible class of algorithms that may match these requirements is anomaly detection algorithms.

1.3 Anomaly Detection

Anomaly detection is a class of ML algorithms encompassing weakly-supervised and unsupervised methods. In the case of the former, one disposes of the label and indirectly uses it in the training process by building a training set solely composed of samples belonging to the *normal* class[2]. While weakly-supervised approaches can be used in situations where the imbalance is too severe for standard supervised approaches to work, unsupervised approaches are usually confined to applications that consist in removing samples that may hinder models' performance on a particular task, *e.g.* mislabeled samples or outliers. In the context of fraud detection, since we dispose of a large dataset of credit card payments with labels, we will focus on *weakly* or semi-supervised anomaly detection.

Weakly-supervised anomaly detection methods differ from standard supervised approaches because they only use labels *indirectly*. Indeed, weakly-supervised approaches consist in training a classifier using a dataset $\mathscr{D}_{train}$ as defined in equation (3), using both sample features $\mathbf{x}_i$ and labels y_i. Moreover, $\mathscr{D}_{train}$ contains samples that belong to each class considered in $\mathscr{Y}$. On the contrary, weakly-supervised anomaly detection methods use the label to build a training set solely composed of a single class, referred to as the *normal* class. In the case of binary classification, the *normal* class is the majority class, *e.g.* the legit payments in the case of fraud detection, and the training set can then be constructed as

$$\mathscr{D}^{\text{AD}}_{train} = \{\mathbf{x}_i : y_i = 0\}_{i=1}^{n}. \tag{8}$$

This anomaly detection framework aims at characterizing the normal distribution, $p(\mathbf{x} \mid y = 0)$. In

[2]Throughout this manuscript, the term *normal* relates to the notion of normality.

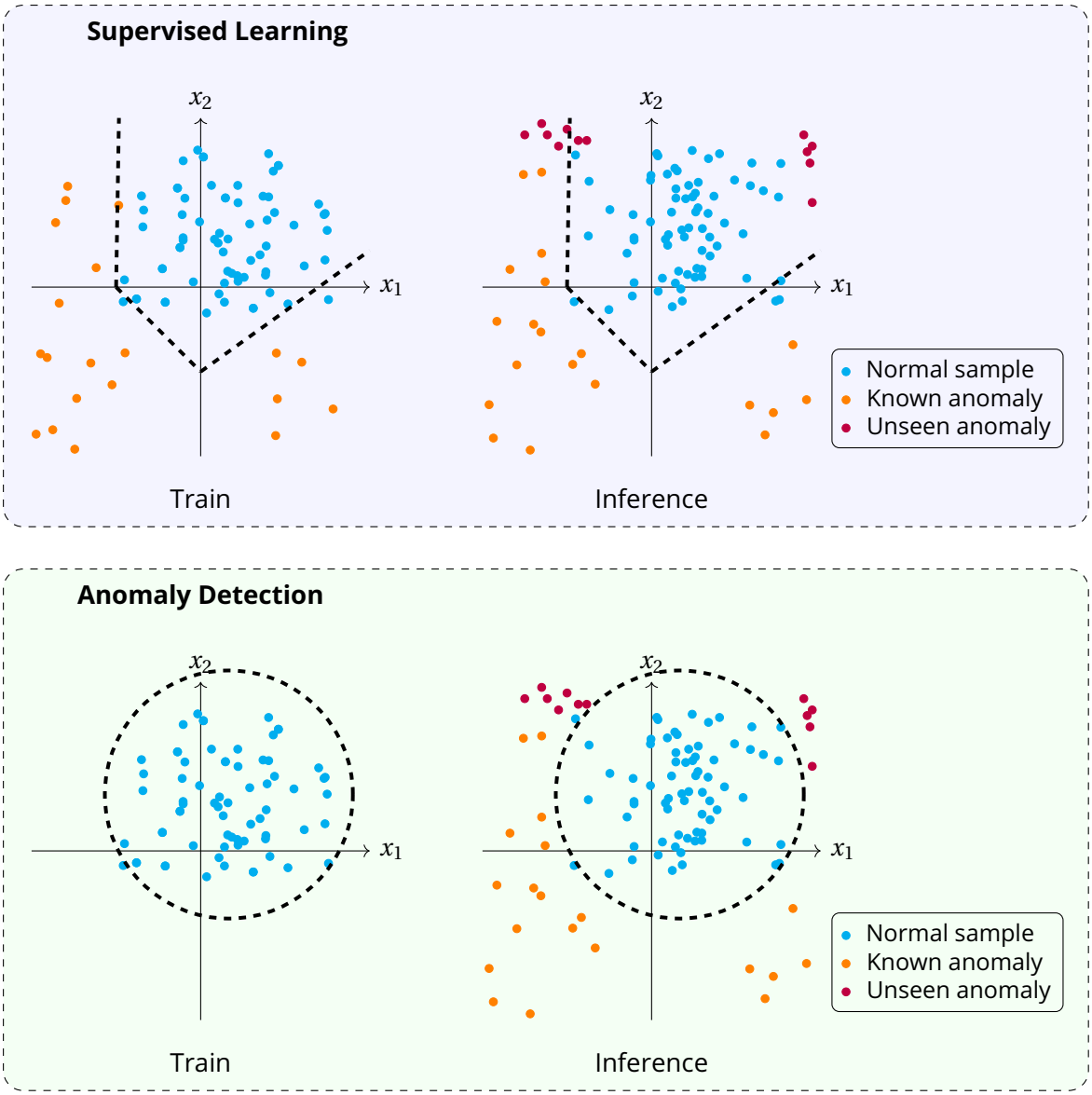


Figure 1.1: Difference between standard supervised learning and weakly supervised anomaly detection. While supervised learning methods use both classes to learn their decision frontier, anomaly detection models only use the *normal* class (●). Suppose in inference, a *new* type of anomaly appears in a region of the data space that did not contain anomalies during training (e.g., *unseen anomalies* (●) at the top right corner of the inference figures). In that case, supervised models will struggle to detect them. On the contrary, anomaly detection models should be able to detect such anomalies.

inference, the characterization determines whether a sample belongs to the normal distribution or should be seen as an anomaly.

As such, the anomaly detection setup presents several advantages over standard supervised approaches for the fraud detection application. Indeed, since the training set is solely composed of normal samples, the inherent imbalanced characteristic of the dataset no longer causes any problems in the learning phase. Moreover, weakly-supervised anomaly detection models should also be robust to concept shift for a similar reason. They should be able to detect new appearing frauds that standard supervised models would struggle to identify. Indeed, since anomalies are defined in opposition to the *normal* distribution, as long as anomalies do not perfectly match the normal distribution,

they should be flagged as *not* belonging to this distribution. Such phenomena are illustrated in figure 1.1.

One usually categorizes weakly-supervised anomaly detection algorithms into four classes or methods: one-class classification, density estimation, reconstruction-based, and self-supervised methods. We extensively discuss all four categories in section 2; however, let us briefly describe each category's underlying concepts.

The term **one-class classification** was coined in Moya and Hush, 1996 and describes identifying anomalies without directly estimating the *normal* density. One-class classification (OCC) involves discriminative models that estimate a decision boundary directly. Some approaches have been widely adopted on various tasks, such as OCSVM (Schölkopf et al., 1999) and SVDD (Tax & Duin, 2004). These kernel-based approaches were then extended in more recent work, such as Deep-SVDD (Ruff et al., 2018) or Deep Semi-Supervised SVDD (Ruff et al., 2020).

Anomaly detection algorithms based on **density estimation** is a straightforward approach that consists in estimating the normal distribution $p(\mathbf{x} \mid y = 0)$ and flagging as anomalies samples that lie in low probability region under this estimation distribution.

Reconstruction-based methods consist in training a model to reconstruct samples from the *normal* class. In inference, the capacity of the trained model to reconstruct a sample serves to estimate whether the sample belongs to the *normal* class or the anomaly class. Indeed, the higher the reconstruction error of a sample, the less likely the sample is to belong to the distribution seen by the model during training. The most standard approaches include training models like autoencoders, Variational Autoencoders (VAE) (Kingma & Welling, 2014) or GANs (Goodfellow et al., 2014) such as ANOGAN (Schlegl et al., 2017).

Finally, **self-supervised** approaches to anomaly detection use pretext tasks to identify anomalies. This method class includes transformation-based approaches (Bergman & Hoshen, 2020; Qiu et al., 2021) or contrastive approaches (Shenkar & Wolf, 2022).

1.4 Research questions

Most general AD methods have shown strong performance for unstructured data but struggle with structured tabular data. Indeed, the best-performing methods (Bergman & Hoshen, 2020; Qiu et al., 2021; Shenkar & Wolf, 2022) are tailored for this particular data type.

A similar observation was also made regarding the recent literature on deep learning for tabular data. Indeed, while methods like deep learning have been widely adapted to all sorts of machine learning tasks for unstructured data, it still lags behind gradient-boosted decision trees (GBDT) on most classification tasks on tabular data (Grinsztajn et al., 2022; Shwartz-Ziv & Armon, 2021). Nevertheless, recent literature includes work that put forward deep architectures tailored to account for the particular tabular data structure. In particular, works like (Gorishniy et al., 2023; Kossen et al., 2021; Somepalli et al., 2021) emphasize the relevance of simultaneously considering feature-feature and sample-sample relations to foster deep models' performance.

Given this terminology, we observe that the anomaly detection literature for tabular data lacks approaches relying on sample-sample dependencies: **I - Are sample-sample dependencies relevant to identifying anomalies within a dataset?**

Similarly, to our knowledge, no AD method in the literature combines both **feature-feature** and **sample-sample** dependencies: **II - Can we leverage efficiently both dependencies to detect anomalies on tabular data ?**

Finally, although *apriori* anomaly detection appears as a pertinent approach for fraud detection: **III - Are anomaly detection methods effective in identifying frauds on a real-life credit-card payment dataset?**

In chapter 3, we propose a novel AD method based on **influence measures** and relying on sample-sample dependencies to try and assess whether AD methods for tabular data using relations between samples can effectively identify anomaly samples (I). In short, the influence of a sample $\mathbf{x}$ on another sample $\mathbf{z}$ measures the extent to which a sample $\mathbf{x}$ contributed to decreasing the loss of $\mathbf{z}$ in the course of training. A positive influence implies that sample $\mathbf{x}$ helped decrease the loss of sample $\mathbf{z}$, while a negative influence implies that the loss of sample $\mathbf{z}$ increased due to $\mathbf{x}$. The proposed approach, `TracInAD`, relies on the hypothesis that normal-to-normal influence relations differ from the normal-to-anomaly influence relations. `TracInAD` consists in (i) training a deep anomaly detection approach on *normal* samples, and (ii) measuring the average influence of a subsample of the training set on the samples of interest: normal samples should have on average a lower influence on anomalies (see figure 3.1). We empirically observe that this hypothesis proves true and permits satisfactory anomaly detection performance on tabular data.

In chapter 4, we propose two novel AD methods for tabular data relying on transformer architectures (Kossen et al., 2021; Vaswani et al., 2017) that are the first AD methods to combine both feature-feature and sample-sample dependencies (I, II). We first propose NPT-AD, a method that *internally* leverages inter-sample dependencies by relying on Non-Parametric Transformers (NPTs) (Kossen et al., 2021). NPT enables accounting for inter-sample dependencies through its Attention Between Datapoints mechanism (detailed in section 4.4.2). Second, we propose a similar method to NPT-AD but based on vanilla transformers (Vaswani et al., 2017) augmented by *external* retrieval modules to enable attending to inter-sample dependencies. Both approaches rely on the self-supervised task of mask reconstruction: we train a model to reconstruct masked features of normal samples and use in inference the reconstruction ability of the trained model to measure *normality*. A higher reconstruction error should indicate anomalousness. Both approaches offer strong performance, especially NPT-AD, which obtains state-of-the-art anomaly detection performance on a large benchmark of 31 tabular datasets. These results emphasize that combining both types of dependencies indeed permits enhanced anomaly detection performance on tabular data.

Finally, in chapter 5, we test anomaly detection methods on a real-life credit-card payment dataset and compare their performance to gradient-boosted decision trees to assess whether AD methods are indeed relevant to detect frauds (III). Our experiments indicate that anomaly detection methods perform poorly on our private real-life credit card dataset compared to gradient-boosted decision trees. We also compare the performance of AD methods on our dataset and their performance on an open-source fraud detection dataset (Pozzolo et al., 2015). We observe that the tested AD methods obtain significantly stronger performance on this second dataset and discuss reasons that may account for such discrepancies. Overall, our experiments support the idea that anomaly detection methods should be further improved to compete with GDBT for fraud detection.

1.5 Publications

Our work to be presented in this manuscript gave birth to the following publications:

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024a). Beyond individual input for deep anomaly detection on tabular data. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, & F. Berkenkamp (Editors), *Proceedings of the 41st international conference on machine learning* (Pages 48097–48123). PMLR. (Cited on pages 78, 81, 82).

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024b). Retrieval augmented deep anomaly detection for tabular data. *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA*. https://doi.org/https://doi.org/10.1145/3627673.3679559

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2022). TracInAD: Measuring influence for anomaly detection. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–6. https://doi.org/10.1109/IJCNN55064.2022.9892058 (cited on pages 48, 78)

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2023). Comparative evaluation of anomaly detection methods for fraud detection in online credit card payments.

The chapters of this manuscript consist of extended and enriched versions of material already published. This manuscript is self-contained, hence it is unnecessary to read the associated publications to understand its content.

1.6 Code

The code to reproduce the experiments discussed in this manuscript is available through the following links:

Chapter 3: https://github.com/hugothimonier/TracInAD

Chapter 4: https://github.com/hugothimonier/Retrieval-Augmented-Deep-Anomaly-Detection-for-Tabular-Data

Chapter 4: https://github.com/hugothimonier/NPT-AD

Chapter 2

Related Works

In this section, we discuss topics related to our research questions. We present a summary of the literature on imbalanced learning in section 2.1. Then, we discuss the recent advancements in the literature related to anomaly detection in 2.2. Finally, in section 2.3, we present an overview of the literature on learning from tabular data using deep models.

2.1 Imbalanced Learning

According to Krawczyk, 2016, there are three main approaches to addressing the problem of learning from imbalanced data.

- *Data-level methods*: modify the dataset to balance distributions.
- *Algorithm-level methods*: methods that modify an existing algorithm to adapt it to the skewed distribution setting.
- *Hybrid methods*: methods that combine both previous approaches.

2.1.1 Data-Level methods

Data-level methods mainly modify the training set to foster a more accurate training of classifiers. Two approaches are usually considered: oversampling and under-sampling. The former designates the process of generating new samples for the minority class to reduce the imbalance between the majority and minority class. The latter consists in removing samples from the majority class to achieve a similar objective.

As discussed by He and Garcia, 2009, under-sampling encompasses random under-sampling methods or more refined *informed* methods. Random oversampling consists in randomly duplicating samples from the minority class, while random under-sampling consists in randomly dropping samples belonging to the majority class. Informed methods include Easy Ensemble and BalanceCascade (Liu et al., 2009) or K-nearest neighbors-based methods (Zhang & Mani, 2003), e.g. NearMiss. For the case of image datasets, oversampling can be achieved using Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) while oversampling can become less trivial for structured data spaces, e.g., tabular datasets. Strategies such as SMOTE (Synthetic Minority Oversampling Technique) (Chawla et al., 2002) were proposed as a data augmentation technique to increase the cardinality of the minority

class. In a nutshell, SMOTE generates artificial samples from the minority class based on the feature space similarities between samples. Formally, consider a sample from the minority class $x_i \in \mathscr{X}$ and its k nearest neighbors in $\mathscr{X}$ belonging to the minority class. A synthetic sample $\hat{x}_i$ is created from randomly selecting one of the k nearest neighbours of x_i, denoted x_k, as well as a scalar $\lambda \in [0,1]$ such that

$$\hat{x}_i = (1-\lambda)x_i + \lambda x_k$$

Other methods augmenting SMOTE also exist in the literature, such as Borderline-SMOTE (Hui Han, 2005) or Modified Synthetic Minority Over-sampling Technique (MSMOTE) (Hu et al., 2009). Bordeline-SMOTE identifies instances of the minority class that are located near the border between the minority and majority classes. These instances are considered more informative for the learning process because they are more likely to be misclassified by the classifier. For each borderline instance, Borderline-SMOTE selects one or more of its nearest minority class neighbors. These neighbors serve as the basis for generating synthetic samples. Unlike SMOTE, which connects the minority instance to a random neighbor, Borderline-SMOTE only connects the instance to its nearest neighbors that belong to the minority class. MSMOTE works similarly to Bordeline-SMOTE since it focuses on the minority class by considering the density of its distribution and then focuses on the minority class close to the border between the majority and minority classes. For each borderline instance, MSMOTE selects its k-nearest neighbors from the minority class, considering the density distribution. MSMOTE assigns different weights to the neighbors based on their density. Samples with lower density are given higher weights to emphasize their importance in generating synthetic samples.

While the listed methods are often suited for unstructured data, especially image instances, those approaches may need to be revised for structured tabular data. Hence, for oversampling, one may consider recent approaches proposed in the tabular data generation literature to augment the minority class's cardinality. For instance, Kotelnikov et al., 2023 propose TabDDPM, which models tabular data using diffusion models. Their approach accounts for the heterogeneity of tabular data and proposes a different approach to generate features depending on its type: categorical or numerical. It relies on the standard Gaussian diffusion processes for numerical features, while categorical and binary features involve multinomial diffusion. Other works, such as TabSyn (Zhang et al., 2024), have considered using diffusion models in the latent space to alleviate the issue of differentiating the diffusion process between categorical and numerical features. Other works include tabular generation using VAE (Kingma & Welling, 2014) such as TVAE (Xu et al., 2019) or GOGGLE (Liu et al., 2023), relying on GAN (Goodfellow et al., 2014) such as CTGAN (Xu et al., 2019), CTABGAN (Zhao et al., 2021) or using LLMs (Borisov et al., 2023).

2.1.2 Algorithm-level Methods

This class of methods involves altering existing classifiers to alleviate their bias towards the majority class. The main branch is known as *cost-sensitive learning*: the learner is modified by introducing a varying cost of misclassification between each considered group in the hope that it will annihilate the bias towards the majority class. Through increasing the cost of misclassifying a sample belonging to the minority class, the importance of the less represented samples is boosted. As discussed in Krawczyk, 2016, setting the suitable cost matrix in real-world applications is often non-trivial and requires precise expert knowledge. However, this approach disposes of appealing theoretical foundations since

the literature has shown clear connections between imbalanced learning and cost-sensitive learning (Chawla et al., 2004).

Ensembling methods to alleviate the issue of imbalanced classes have also been widely used, as discussed in Galar et al., 2012: bagging, boosting, and methods combining the two. Ensemble methodology aims at improving a single classifier's performance by combining several classifiers to obtain a new classifier that outperforms each classifier taken independently. First, bagging, introduced in Breiman, 1996, consists in training several classifiers on bootstrapped datasets from the original dataset. In inference, a weighted average or majority vote is applied to the classifiers' predictions to form the prediction of the bagging classifier. Second, boosting introduced in the seminal work of (Schapire, 1990) consists in training several models on the *same* dataset (contrary to bagging) iteratively. At each iteration, one sets a goal to classify the misclassified samples at the previous iteration correctly; this allows the algorithm to focus on the most challenging examples. Moreover, weights are assigned to each classifier after each iteration to account for how well the weak classifier performed: the better, the higher the weight in the future predictions. Examples of such algorithms include AdaBoost (Freund & Schapire, 1997) or XGBoost (Chen & Guestrin, 2016) and LightGBM (Guolin et al., 2017).

2.1.3 Hybrid Methods

Hybrid methods simply consist in combining both data-level and algorithm-level methods to improve a model's performance. This approach includes SMOTEBoost (Chawla et al., 2003), which combines SMOTE for oversampling the minority class with boosting to give more weight to difficult-to-classify instances. This also includes methods like Random Under-Sampling Boosting (RUSBoost) (Seiffert et al., 2009), which consists in training a sequence of weak learners iteratively, focusing more on the minority class. In each iteration, RUSBoost applies random undersampling to balance the dataset and trains a weak learner on this balanced dataset. This learner typically tries to focus on the minority class, as after each iteration, RUSBoost updates the weights of misclassified instances, putting more emphasis on the misclassified samples. Another notable approach close to RUSBoost is Adaptative Synthetic Sampling (ADASYN) (He et al., 2008): A variant of SMOTE that generates more synthetic examples for minority class instances that are harder to learn.

Other work includes EasyEnsemble (Liu, Wu, et al., 2008), which focuses on creating several balanced subsamples containing an equal number of samples from the majority class and all samples from the minority class. Then, one trains weak learners on each of these subsamples. Once all base learners are trained, EasyEnsemble combines them into an ensemble. This ensemble typically takes a simple form, such as averaging the predictions for classification tasks or taking a weighted average based on the performance of each base learner. Liu, Wu, et al., 2008 also propose BalanceCascade, which involves an iterative approach that focuses on progressively reducing the imbalance in the dataset. BalanceCascade starts by randomly under-sampling the majority class to create an initial balanced subset on which a weak learner is trained. This weak learner is then evaluated on the *entire* dataset, and correctly identified samples from the majority class are removed for the next iteration. The next iteration repeats a similar process where, this time, the dataset used to create a balanced subsample excludes the correctly identified samples from the previous iteration.

2.2 Anomaly Detection

Anomaly detection is a class of algorithms that can be applied to all data modalities such as tabular data (Bergman & Hoshen, 2020; Qiu et al., 2021; Shenkar & Wolf, 2022), images (Golan & El-Yaniv, 2018; Kim et al., 2020; Schlegl et al., 2017) or time series (Dai & Chen, 2022; Krönert et al., 2024; Shen et al., 2020). This section will focus on the literature of anomaly detection for images and tabular data, as most methods proposed for images are also applicable to tabular data. On the contrary, most methods tailored for time-series cannot be directly adapted to other data modalities such as tabular data. As discussed in section 1, anomaly detection is comprised of two classes of algorithms: unsupervised and semi-supervised anomaly detection. This section briefly defines both approaches and presents a series of works relevant to our topic. We cover both unsupervised and semi-supervised algorithms, with a strong focus on semi-supervised methods since they constitute the class of algorithms of interest. We start by presenting shallow anomaly detection methods in section 2.2.1, and then we delve into deep anomaly detection approaches in section 2.2.2.

Weakly-supervised Anomaly Detection In section 1.3 we give a detailed characterization of semi-supervised anomaly detection algorithms. In short, semi-supervised anomaly detection consists in learning to caracterise the *normal* distribution from a dataset $\mathcal{D}^{\text{AD}}_{train}$ solely composed of samples generated by this *normal* distribution, as detailed in equation (8). This class of AD algorithm is considered weakly supervised since it supposes that one disposes of a labeled dataset that allows to construct a dataset only composed of samples belonging to the *normal* class. These algorithms are often used for classification tasks that display strong imbalances between classes, which prevent standard supervised algorithms from offering sufficient performances.

Unsupervised Anomaly Detection Unsupervised AD methods designate the more challenging tasks of identifying anomalies in a dataset without disposing of a label. In other words, this class of algorithms aims at constructing a decision function without disposing of the labels. Therefore, these methods aim to identify the most effective way to characterize anomalies within a dataset, e.g., low-probability samples under an estimated distribution. In this approach, the training set is also the dataset from which one wishes to exclude anomalies. Unsupervised AD is usually confined to applications that remove outliers or mislabeled samples from a dataset.

In the rest of the section, we discuss existing methods found in the literature, starting with shallow methods (i.e., non-deep-learning-based methods). Then, we present deep methods that have gained growing attention recently.

2.2.1 Shallow Anomaly Detection Methods

A broad range of shallow anomaly detection methods exist, such as one-class classification approaches, approaches based on density estimation, probabilistic models, and cluster-based methods.

Density Estimation and Probabilistic Models

The most straightforward approach to anomaly detection involves computing the Mahalanobis distance from a test sample to the training data mean as first proposed in Laurikkala et al., 2000 for med-

ical datasets. There also exist more elaborate methods that include kernel density estimators (KDE) (Parzen, 1962), histogram estimators, or mixture models (Roberts & Tarassenko, 1994a) to model the *normal* data distribution. This type of approach is mainly suited for low-dimensional data and displays limitations when it comes to higher dimensional problems since non-parametric estimation often suffers from the curse of dimensionality (Ruff et al., 2021): estimators require a sample size exponential in the dimension.

Li et al., 2020 propose an efficient anomaly detection method, COPOD, based on copulas, which model multivariate data distributions. Formally, a d-variate copula $\mathrm{C}:[0,1]^d \to [0,1]$ is the cumulative distribution function (CDF) of a random vector $(\mathrm{U}_1, \mathrm{U}_2, \ldots, \mathrm{U}_d)$ with $\mathrm{U}_j \sim \mathscr{U}(0,1)$ for $j = 1, \ldots, d$ such that:

$$\mathrm{C}_{\mathrm{U}}(\mathbf{u}) = \mathbb{P}(\mathrm{U}_1 \leq u_1, \ldots, \mathrm{U}_d \leq u_d) \tag{1}$$

where $\mathbb{P}(\mathrm{U}_1 \leq u_j) = u_j$ for $j = 1, \ldots, d$ and $u_j \in [0,1]$. Via inverse sampling, uniform distributions can be transformed into any desired distribution,

$$\mathrm{X}_j := \mathrm{F}_j^{-1}(\mathrm{U}_j) \sim \mathrm{F}_j \tag{2}$$

Based on this, Sklar's Theorem (Sklar, 1959) stipulates that copulas can serve to describe the joint distribution $\mathrm{F}(\mathbf{x}_1, \ldots, \mathbf{x}_d)$ of any random variable $(\mathrm{X}_1, \ldots, \mathrm{X}_d)$ using only the marginals $\mathrm{F}_1, \ldots, \mathrm{F}_d$ as

$$\mathrm{F}(\mathbf{x}) = \mathrm{C}(\mathrm{F}_1(\mathbf{x}_1), \ldots, \mathrm{F}_d(\mathbf{x}_d))$$

Thus, considering samples $\mathbf{x} \in \mathscr{X} \subseteq \mathbb{R}^d$, one can model each feature of the samples in $\mathscr{D}_{train} = \{\mathbf{x}_i\}_{i=1}^n$ separately and then link them together to estimate the joint distribution. In their paper, the authors propose to use empirical copulas that fit empirical cumulative distribution functions $\hat{\mathrm{F}}(x)$ to model the dataset distribution.

$$\hat{\mathrm{F}}(\mathbf{x}) = \mathbb{P}((-\infty, \mathbf{x}]) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{\mathrm{X}_i \leq \mathbf{x}\},$$

where $\mathbf{1}\{\cdot\}$ is the indactor function. By leveraging the relation given in equation (2), one obtains the empirical copula observations and thus the empirical equivalent of equation (1). Based on this, the proposed algorithm, COPOD, is a three-staged method. First, compute the empirical CDF on the dataset of interest. Second, the empirical CDFs produce the empirical copulas and compute the tail probabilities. In such a framework, low probability points are then marked as anomalies.

One-Class Classification

One-class classification can be defined as a binary classifier that learns to separate the normal class from the anomaly class using exclusively normal samples in the training phase. The objective is two-fold: (i) flag all anomalies and (ii) keep the false positive rate as low as possible.

Formally, Ruff et al., 2021 propose to express the following objectives using the classification risk. Considering a data space $\mathscr{X}$ and a binary label space $\mathscr{Y} = \{0,1\}$, where $y = 0$ denotes *normal* data samples while $y = 1$ denotes anomalous points. Let us denote the normal distribution $p(\mathbf{x} \mid y = 0)$, and the anomalous distribution $p(\mathbf{x} \mid y = 1)$. Let $\ell : \mathbb{R} \times \mathscr{Y} \to \mathbb{R}$ be a classification loss and $f : \mathscr{X} \to \mathbb{R}$ a real-valued measurable score function taken from $\mathscr{F}$ the set of score functions. Minimizing the

classification risk of f under the loss ℓ formalizes the two objectives detailed above as such

$$\min_{f\in\mathscr{F}} \mathrm{R}(f) = \min_{f\in\mathscr{F}} \underbrace{\mathbb{E}_{\mathrm{X}\sim\mathbb{P}^+}[\ell(f(\mathrm{X}),0)]}_{\text{(A)}} + \underbrace{\mathbb{E}_{\mathrm{X}\sim\mathbb{P}^-}[\ell(f(\mathrm{X}),1)]}_{\text{(B)}}. \tag{3}$$

Minimizing (B) in equation (3) is equivalent to minimizing the miss rate, *i.e.* trying to achieve objective (i), while minimizing (A) aims at achieving objective (ii). Minimizing the empirical risk in the unsupervised approach writes

$$\min_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(f(\mathbf{x}_i),0) + \mathscr{R} \tag{4}$$

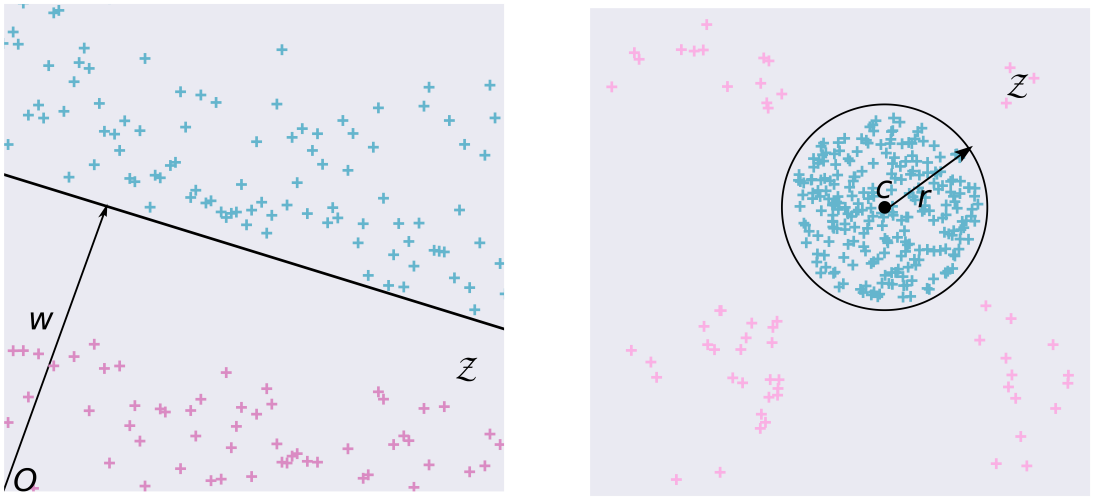
where $\mathscr{R}$ is a regularization term without which equation (4) would be trivial. $\mathscr{R}$ serves as a means to minimize the miss rate and improve generalization. A supplementary term is sometimes added to the empirical risk minimized in (4) to include labeled data as in (Ruff et al., 2020).

Tax and Duin, 2004 propose enclosing *normal* data within a hyper-sphere and flag data samples as anomalies when they lie outside the estimated hyper-sphere. Considering a training set $\mathscr{D}_n^{train} = \{\mathbf{x}_i\}_{i=1}^n$, this objective is met through the following minimization problem

$$\begin{cases} \min\limits_{r,\mathrm{C},\xi} & r^2 + \frac{1}{\nu n}\sum_{i=1}^n \xi_i \\ & \text{s.t. } \|\mathbf{x}_i - \mathrm{C}\|^2 \le r^2 + \xi_i,\ \xi_i \ge 0,\ \forall i. \end{cases} \tag{5}$$

The minimization problem described in equation (5) aims at finding a hyper-sphere with radius $r > 0$ and center $\mathrm{C} \in \mathscr{X}$ that encloses the data. Minimizing r^2 allows controlling for the volume of the hyper-sphere, while slack variables $\xi_i \ge 0$ allow some points to fall outside the hyper-sphere: the boundary is considered *soft* and $\nu \in [0,1]$ controls the allowed proportion of points outside the sphere. This thus allows $\mathscr{D}_n^{train}$ to contain a small portion of anomalous points as in the unsupervised AD setting.

The standard SVDD (Tax & Duin, 2004) was then refined using kernels to allow mapping data samples to Hilbert space instead of keeping them within the original data space.



(a) OCSVM **(b)** SVDD

Figure 2.1: OCSVM vs SVDD

$$\begin{cases} \min\limits_{r,C,\xi} & r^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i \\ & \text{s.t. } \|\phi_k(\mathbf{x}_i) - C\|^2 \leq r^2 + \xi_i,\ \xi_i \geq 0,\ \forall i, \end{cases} \tag{6}$$

where $C, r \in \mathcal{Z}$.

Another approach related to SVDD is the one of One-Class SVM (Schölkopf et al., 1999), which involves finding a hyperplane as distant as possible from the origin, instead of a hyper-sphere, that separates *normal* data from the origin. In the latter setting, points below the separating hyperplane and closer to the origin are considered anomalies. The hyperplane $\omega \in \mathcal{Z}$ that separates the data in the feature space with maximum margin to the origin is obtained through the following minimization problem

$$\begin{cases} \min\limits_{\omega,\rho,\xi} & \frac{1}{2}\|\omega\|^2 - \rho + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i \\ & \text{s.t. } \rho - \langle\phi_k(\mathbf{x}_i), \omega\rangle \leq \xi_i,\ \xi_i \geq 0,\ \forall i. \end{cases} \tag{7}$$

The margin to the origin is given by $\frac{\rho}{\|\omega\|}$ which is maximized by maximizing ρ while $\|\omega\|$ acts as a normalizer. A visualization of the difference between OCSVM and SVDD is given in figure 2.1.

Other one-class classification methods involve tree-based approaches. For instance, isolation forest (Liu, Ting, et al., 2008) is an unsupervised AD method that relies on a binary tree structure to isolate anomalies from the normal distribution. Isolation forests are composed of several isolation trees that are constructed such that at each iteration, a feature is randomly selected, and a random value is also selected within the range of the feature values to partition the data into two groups. Selecting a random feature and a random splitting value is repeated recursively until specific stopping criteria are met. The underlying hypothesis that will serve to identify anomaly samples is that anomalies are rare instances that are few and far between compared to normal instances. In the tree construction process, anomalies are expected to be isolated closer to the tree's root as they should require fewer partitions to be isolated. The average depth of anomalies across multiple trees is used as a measure of anomaly score: the lower, the more likely the sample is to be an anomaly. Isolation Forests can handle large datasets efficiently because they require fewer trees to perform well compared to other tree-based methods. Hariri et al., 2021 put forward an augmented version of Isolation Forest: Extended Isolation Forest (EIF). Their method aims to mitigate the limitations of isolation forests in the case of high-dimensional data. Instead of using only one random split for each feature, as in the isolation forest algorithm, EIF employs multiple random splits. The anomaly score is then constructed similarly to that of the isolation forest.

RRCF, another notable approach relying on trees, was proposed in Guha et al., 2016 and focuses on high-dimensional datasets. As in the isolation forest approach, RRCF employs random partitioning but involves random cut trees instead of binary trees. These random cut trees split the data space into subtrees using random dimensions and split points. The authors also introduce a " robustifying " technique to handle correlated features. It randomly selects a subset of dimensions for splitting at each node, which helps to mitigate the effects of correlated features and prevents the algorithm from being biased towards any particular dimension. Overall, each sample's anomaly score is computed based on its depth across all trees.

Another recent tree-based approach is PIDForest (Gopalan et al., 2019), in which a collection of trees is also constructed to partition the data space. In this approach, the splits at each node are chosen to favor partitions of varying sparsity rather than choosing random cuts as in other methods.

Reconstruction Based Methods

Reconstruction-based methods include both shallow and deep approaches to anomaly detection. According to Ruff et al., 2021, these methods rely on one of the following hypotheses:

- **Manifold Assumption**: *normal* data lies on some lower-dimensional manifold $\mathscr{M} \subset \mathscr{X}$ with $\dim(\mathscr{M}) < \dim(\mathscr{X})$. However, samples that come from a different distribution than $p(\mathbf{x} \mid y = 0)$ do not lie in such a manifold.

- **Prototype Assumption**: there exist a finite number of prototypical elements in $\mathscr{X}$ that characterize the data. *Normal* data samples can be characterized by a discrete set of vectors $\{c_1, \ldots, c_k\}$.

Thus, based on these two assumptions, training a model to reconstruct samples from the *normal* distribution $p(\mathbf{x} \mid y = 0)$ should allow to achieve low reconstruction error for any sample belonging to that distribution. However, since anomalies are believed to stem from a different distribution $p(\mathbf{x} \mid y = 1)$, the reconstruction error should be significantly higher for anomalies. Based on this premise, consider a model $f : \mathscr{X} \to \mathscr{X}$ a model trained exclusively on *normal* samples to reconstruct a sample $\mathbf{x}$. Let $\hat{x}$ be the reconstructed sample using model f, $\hat{\mathbf{x}} = f(\mathbf{x})$, then an anomaly score can be derived as such

$$Score(x) = \|\mathbf{x} - \hat{\mathbf{x}}\| \tag{8}$$

The most common shallow reconstruction-based anomaly detection method is based on Principal Component Analysis (PCA) or Bayesian PCA. PCA has been used in a wide range of applications as a means of detecting anomalies, as in Dutta et al., 2007 or Schölkopf et al., 2007. Recall that the core objective of PCA is to find an orthogonal basis W in the original data space $\mathscr{X}$ which minimizes the empirical variance of the centered data samples in $\mathscr{D}_n^{train} = \{\mathbf{x}_i\}_{i=1}^n, \mathbf{x}_i \in \mathscr{X} \subseteq \mathbb{R}^{\mathrm{D}}$. The $d \leq \mathrm{D}$ components that account for the most variance are given by the d eigenvectors that have the largest eigenvalues. Finding W as such is equivalent to finding the orthogonal projection $\mathrm{W}^\top \mathrm{W}$ to a d-dimensional linear subspace such that the mean squared reconstruction error is minimized,

$$\min_{\mathrm{W}} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathrm{W}^\top \mathrm{W} \mathbf{x}_i\|^2 \text{ s.t. } \mathrm{W}\mathrm{W}^\top = \mathrm{I}.$$

Thus, PCA optimally solves the reconstruction objective for a linear encoder ϕ_e and transposed linear decoder ϕ_d with the constraint that $\mathrm{W}\mathrm{W}^\top = \mathrm{I}$ as follows,

$$f(\mathbf{x}) = \phi_d(\phi_e(\mathbf{x})) = \mathrm{W}^\top(\mathrm{W}x).$$

It is also possible to use nonlinear PCA, such as kernel-PCA, to extend the reconstruction learning capacity to improve anomaly detection performance. In both cases, the reconstruction error can serve as an anomaly score in the inference phase.

2.2.2 Deep Anomaly Detection Methods

Scholars have adapted a certain number of shallow anomaly detection methods to include deep networks, such as Ruff et al., 2018, which replaced the use of kernels of the SVDD approach with the use of a deep network as a feature extractor. Other types of approaches were also recently proposed

following the rapid increase in the performance of deep models on a wide variety of tasks. These methods include reconstruction, as previously discussed, and self-supervised approaches.

Deep One-Class Classification

Ruff et al., 2018 propose Deep-SVDD, an extension of the SVDD approach discussed in section 2.2.1 in which a neural network replaces kernels to achieve a similar objective. Formally, let us consider an input space $\mathscr{X} \subseteq \mathbb{R}^d$ and an output space $\mathscr{Z} \subseteq \mathbb{R}^p$. The core objective of Deep-SVDD is to learn the weights $\mathscr{W}$ of a neural network $\phi(.;\mathscr{W}) : \mathscr{X} \to \mathscr{Z}$, where $\mathscr{W} = \{\mathrm{W}^1, \ldots, \mathrm{W}^\mathrm{L}\}$ are the weight of the L layers: W^ℓ is the weight of the $\ell - th$ layer. SVDD aims to map the training data into a hyper-sphere in $\mathscr{Z}$ of minimum volume. Thus, the weights of the network are obtained through the following minimization problem:

$$\min_{\mathscr{W}} \frac{1}{n} \sum_{i=1}^{n} \|\phi(\mathbf{x}_i;\mathscr{W}) - c\|^2 + \frac{\lambda}{2} \sum_{\ell=1}^{\mathrm{L}} \|\mathrm{W}^\ell\|_\mathrm{F}^2 \tag{9}$$

where $\|.\|_\mathrm{F}$ is the Frobenius norm. Without imposing any restriction on the deep neural network's architecture, this approach can lead to a trivial situation where the learned model maps each sample in $\mathscr{X}$ to a single point in the hyperspace $\mathscr{Z}$. This situation, known as *model collapse*, can be avoided by using two recently proposed regularization techniques (Chong et al., 2020):

- Regularization with hinge loss on minibatch variance: force hidden representations of samples $\mathbf{x}_i \in \mathrm{B}_j$ (batch j) to be diverse by putting a lower bound on the minibatch variance of the hidden representations.
- Regularization with random noise injection via the loss layer: add a supplementary layer to the original network, used to classify random labels during training, dropped at inference time.

Other regularizations have been proposed to avoid model collapse. Sendera et al., 2021 to use Flow-Based Models as a feature extractor to benefit from the invertibility property of Flow-Based models to avoid model collapse. Another approach, DROCC (Goyal et al., 2020), involves adversarial learning and a classifier on top, which prevents model collapse.

Ruff et al., 2020 propose an extension of Deep-SVDD to include labeled data during training to refine the original model's performance. For instance, a simple augmentation of the loss is considered to ensure that the known anomalies are mapped farther away from the hyper-sphere, which includes the *normal* samples in $\mathscr{Z}$. Consider the case where we dispose of m samples with known labels during training, where $\tilde{y}_i = 1$ if sample $\tilde{\mathbf{x}}_i$ is normal and $\tilde{y}_i = -1$ otherwise. Then the solved problem in (9) becomes,

$$\min_{\mathscr{W}} \frac{1}{n+m} \sum_{i=1}^{n} \|\phi(\mathbf{x}_i;\mathscr{W}) - c\|^2 + \frac{1}{n+m} \sum_{j=1}^{m} \|\phi(\tilde{\mathbf{x}}_j;\mathscr{W}) - c\|^{2\tilde{y}_j} + \frac{\lambda}{2} \sum_{\ell=1}^{\mathrm{L}} \|\mathrm{W}^\ell\|_\mathrm{F}^2. \tag{10}$$

Reconstruction-based method

As discussed in section 2.2.1, a large spectrum of methods have investigated reconstruction-based methods, especially since deep neural networks have shown strong capacity in reconstructing samples. For instance, a wide range of empirical and theoretical evidence supports the fact that when properly regularized, deep networks achieve better generalization capacity than shallow methods to characterize datasets (Vincent et al., 2010). Many anomaly detection methods consist in using autoencoders, *e.g.* Variational Autoencoders (VAEs) (Kingma & Welling, 2014), to learn the *normal* distribution

$p(\mathbf{x} \mid y = 0)$ and to evaluate in inference whether a point belongs to $p(\mathbf{x} \mid y = 0)$ through measuring the reconstruction error of the model. As previously discussed, one considers learning the *normal* distribution through exclusively including *normal* points from $\mathscr{X}$ in $\mathscr{D}_n^{train}$ to learn the weights W of deep neural network composed of a decoder $\phi_d : \mathscr{Z} \to \mathscr{X}$ and an encoder $\phi_e : \mathscr{X} \to \mathscr{Z}$ such that:

$$\phi(.;\mathrm{W}) = \phi_d \circ \phi_e(.;\mathrm{W}).$$

Then, at inference time, an anomaly score is constructed based on the reconstruction error of a data sample $\mathbf{x} \in \mathscr{X}$ measured as in equation (8):

$$Score(\mathbf{x}) = \|\mathbf{x} - \phi(\mathbf{x};\mathrm{W})\|_2^2$$

Standard autoencoders (Baldi & Hornik, 1989) involve a bottleneck architecture to prevent the network from learning the identity function. The parameters of the network are then optimized to minimize the reconstruction error $\mathscr{L}(\mathbf{x}, \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$ where $\hat{\mathbf{x}} = \phi(\mathbf{x};\mathrm{W})$. Standard autoencoders have been used for anomaly detection purposes, such as in (Finke et al., 2021) or (Roy & Vijay, 2020); other approaches have also considered regularized autoencoders such as VAEs (Pol et al., 2019) or memory-augmented deep autoencoders (Gong et al., 2019a).

Recently, Kim et al., 2020 have proposed RaPP, a more elaborate methodology to detect anomalies using autoencoders than simply using the reconstruction error. Instead of only comparing the reconstructed sample and the original sample as in (8), authors propose to also compare the hidden representations of both samples by passing the reconstructed sample through the autoencoder.

Formally, consider at inference time the trained autoencoder $\phi(.,\mathscr{W})$ and a test sample $\mathbf{z} \in \mathscr{X}$. RaPP (Kim et al., 2020) consists in first feeding $\mathbf{z}$ to the autoencoder $\phi(.;\mathscr{W})$ to store its hidden activations $\mathbf{h}(\mathbf{z}) = \{h_1, \ldots, h_l\}$. Secondly, the reconstructed sample $\hat{\mathbf{z}} = \phi(z;\mathrm{W})$ is also fed to the learned autoencoder to obtain its hidden activation as well $\mathbf{h}(\hat{\mathbf{z}}) = \{\hat{h}_1, \ldots, \hat{h}_l\}$. The augmented reconstruction error thus becomes

$$s_{\mathsf{RaPP}}(\mathbf{z}) = \sum_{i=0}^{\ell} \|h_i - \hat{h}_i\|_2^2 = \|\mathbf{h}(\mathbf{x}) - \mathbf{h}(\hat{\mathbf{z}})\|_2^2, \tag{11}$$

where $h_0 = \mathbf{z}$ and $\hat{h_0} = \hat{\mathbf{z}}$.

Reconstruction-based methods rely on the idea that when properly trained on normal samples, autoencoders will yield higher reconstruction errors for anomalies if they stem from a different distribution. However, it has been observed that autoencoders may generalize so well that some anomalies could also be well reconstructed (Gong et al., 2019a). This situation may cause the regular reconstruction-based methods to struggle. Thus, Gong et al., 2019a propose MemAE, a memory-augmented autoencoder, to mitigate this drawback. This approach relies on the prototype assumption discussed in section 2.2.1. For instance, during the training stage, memory contents are created and updated along the training process to contain the most representative prototypical elements of $\mathscr{D}_{train}$. In short, the model is composed of three modules: an encoder, a decoder, as in vanilla autoencoders, and a memory module. The encoder and decoder are trained to minimize the reconstruction error, while the memory module is updated to record the prototypical elements of the encoded samples. During the inference phase, given a test sample x, the encoded sample $z = \phi_e(x)$ is used as a query to find the most relevant prototypical elements in the memory module using an attention-based operator. This allows computing $\hat{z}$ the latent representation of x, which depends linearly on the stored prototypical

elements as

$$\hat{z} = w\mathrm{M},$$

where $\mathrm{M} \in \mathbb{R}^{\mathrm{N} \times p}$ is a matrix composed of N rows denoting memory items $\mathbf{m}_i, i = 1, ...\mathrm{N}$, and $w \in \mathbb{R}^{\mathrm{N}}$ is a row vector with non-negative entries that sum to one. Note that w depends on z. $\hat{z}$ is then used to compute the reconstructed sample using the decoder as $\hat{x} = \phi_d(\hat{z})$. According to the authors, this should favor higher reconstruction error for anomalies while retaining a low reconstruction error for normal samples.

Self-Supervised Anomaly Detection Methods

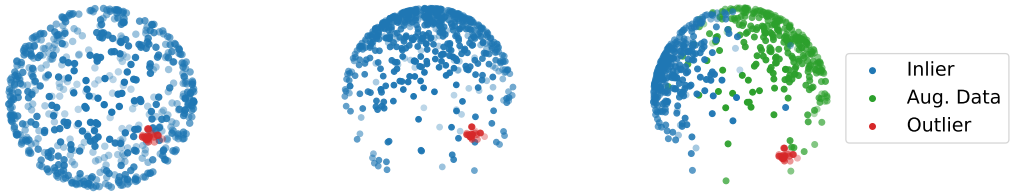
Self-supervision has recently gained attention among scholars, including researchers working on anomaly detection. For instance, one can find in the literature works that include self-supervision as a direct means to detect anomaly through using pretext task (Bergman & Hoshen, 2020; Qiu et al., 2021; Tack et al., 2020), other works have focused on adapting vanilla self-supervised methods to obtain well-suited representation for anomaly detection (Reiss et al., 2021; Reiss & Hoshen, 2023; Sohn et al., 2021). Other works (Sehwag et al., 2021) also consider including self-supervised models as a way to represent data before training classifiers to detect anomalies.

Self-Supervision Self-supervision is often seen as an intermediate between unsupervised and supervised learning. The original goal of self-supervised is to learn a meaningful representation from unlabeled data by using pseudo-label. This representation can be used to improve a model's performance on a supervised or unsupervised task by pushing dissimilar samples farther away while reducing the distance between similar samples. In the literature of self-supervision, two types of samples are usually considered: *positive* and *negative* samples. The term positive sample designates samples related to one another, e.g., two pictures of a dog. In contrast, negative samples include unrelated samples, e.g., a picture of a cat and a dog. Given this terminology, there exist two classes of self-supervised algorithms:

- Contrastive learning techniques that include both negative and positive samples.
- Non-contrastive learning techniques that exclusively rely on positive samples.

Both approaches have offered promising results by giving a meaningful representation of data (Assran et al., 2023; Chen et al., 2020; He et al., 2020; Tian et al., 2019) that allow the improvement of several model performances on a broad range of tasks. Apart from improving the supervised and unsupervised model's performance, Hendrycks et al., 2019 have shown that self-supervision also improves robustness and uncertainty estimation for anomaly detection tasks. Non-contrastive learning techniques such as (Grill et al., 2020) or (Chen & He, 2020) were recently investigated in Tian et al., 2021 in order to show how, without using negative examples, these non-contrastive models manage to avoid representational collapse. (Wang & Isola, 2020) also investigate contrastive learning techniques and show that minimizing the contrastive loss was equivalent to aiming at achieving a two-fold objective:

- Alignment/Closeness, which stipulates that similar samples have similar features,
- Uniformity: obtained representations preserve maximal information and span data over all the feature space.



(a) Perfect uniformity. **(b)** Reduced uniformity. **(c)** Distribution augmentation.

Figure 2.2: Illustration of the concept of Uniformity (figure taken from Sohn et al., 2021)

For instance, authors show that minimizing the vanilla contrastive loss is asymptotically equivalent to minimizing two well-defined losses that respectively try to achieve alignment and uniformity.

Contrastive pretraining for improved Anomaly detection Sohn et al., 2021 build on the two-fold objective of contrastive learning (Wang & Isola, 2020) and propose a pretraining methodology to foster better performance of anomaly detection models. They propose a two-stage methodology, which consists in first learning a self-supervised representation from one-class data and then building a one-class classifier on the learned representations.

The authors discuss how, on the one hand, alignment appears as a desirable feature to achieve good anomaly detection performances. However, on the other hand, uniformity may deteriorate a model's capacity to detect anomalies, as figure 2.2 may indicate. Thus, the authors propose a framework to reduce uniformity learning when optimizing the contrastive loss. To achieve such an objective, they propose to reduce the batch size, which is often set to be quite large in contrastive training frameworks. Moreover, they also propose to involve augmented data for training, *i.e.* use samples in $\mathscr{X} \cup a(\mathscr{X})$ instead of simply $\mathscr{X}$ for learning the contrastive representation, where $a(\mathscr{X}) = \{a(\mathbf{x}) | \mathbf{x} \in \mathscr{X}\}$ and a is an augmentation, *e.g.* a rotation in the case of image datasets.

Self-supervised Auxiliary Tasks for Anomaly Detection Recent works have investigated self-supervised learning as a direct method to detect anomalies. The self-supervised approach proposed by Bergman and Hoshen, 2020 relies on the hypothesis that all normal points lie in a subspace $\mathrm{X} \subset \mathscr{X}$ while anomalous points do not:

$$\forall (\mathbf{x}, y) \in \mathscr{D}_n : y = 0, \mathbf{x} \in \mathrm{X} \subset \mathscr{X} \tag{12}$$

$$\forall (\mathbf{x}, y) \in \mathscr{D}_n : y = 1, \mathbf{x} \notin \mathrm{X} \subset \mathscr{X} \tag{13}$$

Based on the formulated hypothesis, the authors propose to perform M affine transformations,

$$\mathrm{X} \xrightarrow{\text{M Transformations}} \begin{matrix} \mathrm{X}_1 \\ \vdots \\ \mathrm{X}_\mathrm{M} \end{matrix}$$

For all points $\mathbf{x} \in \mathrm{X} \subset \mathscr{X}$, each transformed sample x_m should lie in a subspace $\mathrm{X}_m \subset \mathscr{X}$ if transformations are one-to-one.

$$\forall x \in \mathrm{X} \sim \underbrace{\mathrm{T}(\mathbf{x}, 1)}_{\text{Transfo.}}, \ldots, \mathrm{T}(\mathbf{x}, \mathrm{M}), \text{where } \mathrm{T}(\mathbf{x}, m) \in \mathrm{X}_m$$

Firstly, a feature extractor $f : \mathscr{X} \to \mathscr{Z}$, in the form of a neural network, is trained to map each data point from $\mathscr{X} \subset \mathbb{R}^d$ to a subspace $\mathscr{Z}$. The feature extractor is trained so that each subspace X_m is

mapped to a subspace of $\mathcal{Z}$, $\{f(x)|x \in \mathrm{X}_m\}$, in the form of a hyper-sphere with center c_m. f is trained by optimizing the *center triplet loss*, which learns clustering with low intra-class variation and high inter-class variation: points belonging to X_m should be close to each other in $\mathcal{Z}$, but points belonging to two different subspaces should be very different from each other.

Secondly, a classifier is trained to predict m from $f(\mathrm{T}(\mathbf{x}, m))$, *i.e.* detect which transformation $m \in \{1, \ldots, \mathrm{M}\}$ was applied to $\mathbf{x}$ to obtain $f(\mathrm{T}(\mathbf{x}, m))$. By construction, $\forall \mathbf{x} \in \mathbb{R}^d \backslash \mathrm{X}$, if transformations T are one-to-one, transformed samples will not fall into the appropriate subspace $\mathrm{T}(\mathbf{x}, m) \in \mathcal{X} \backslash \mathrm{X}_m$ and thus $f(\mathrm{T}(\mathbf{x}, m))$ will fall far away from c_m in $\mathcal{Z}$. Thus, estimated probabilities $\mathbb{P}(m|\mathrm{T}(\mathbf{x}, m))$, will be low for those anomalous points since the classifier is exclusively trained on points that lie in X, implying a high anomaly score. The anomaly score is defined as

$$\begin{aligned} Score(\mathbf{x}) = -\log \mathbb{P}(\mathbf{x} \in \mathrm{X}) &= -\sum_m \log \tilde{\mathbb{P}}(\mathrm{T}(\mathbf{x}, m) \in \mathrm{X}_m) \\ &= -\sum_m \log \tilde{\mathbb{P}}(m|\mathrm{T}(\mathbf{x}, m)) \end{aligned}$$

where $\tilde{\mathbb{P}}(m'|\mathrm{T}(\mathbf{x}, m))$ is constructed to be

$$\tilde{\mathbb{P}}(m'|\mathrm{T}(\mathbf{x}, m)) = \frac{e^{\|f(\mathrm{T}(\mathbf{x}, m) - c_{m'}\|^2} + \epsilon}{\sum_{\tilde{m}} e^{\|f(\mathrm{T}(\mathbf{x}, m) - c_{\tilde{m}}\|^2} + \mathrm{M}\epsilon}$$

Another similar approach was recently proposed by Qiu et al., 2021. In this work, the authors propose NeuTraL-AD, a self-supervised approach that relies on transformations. Qiu et al., 2021 propose to learn simultaneously the feature extractor and the transformations applied to data samples instead of considering simple affine transformations. Their approach relies on a constrastive loss, which also serves as an anomaly score in inference and does not involve any classification tasks as in GOAD (Bergman & Hoshen, 2020) to construct the anomaly score.

As in GOAD (Bergman & Hoshen, 2020), a feature extractor $f : \mathcal{X} \to \mathcal{Z}$ is also learned to encode the data. The applied transformations take the form of a neural network and are trained such that (i) encoded transformed and untransformed data are not distant from each other in the embedding space, and (ii) encoded transformed data from different transformations should be distant from each other in the embedding space.

Formally, M transformations are considered $\mathrm{T} : \mathcal{X} \times \{1, \ldots, \mathrm{M}\} \to \mathcal{X}$, and take the form of a neural network $\phi(., \mathrm{W}_m)$ with learnable parameters W_m. Let $f : \mathcal{X} \to \mathcal{Z}$ be the feature extractor whose parameters W are jointly learned with $\{\mathrm{W}_m\}_{m=1}^{\mathrm{M}}$. Define the score function,

$$h(\mathbf{x}_m, x_l) = \exp\left(\frac{\mathrm{sim}(f(\mathrm{T}(\mathbf{x}, l)), f(\mathrm{T}(\mathbf{x}, m)))}{\tau}\right)$$

where $\mathrm{sim}(.,.)$ is the cosine similarity and τ is a temperature parameter. $h(.,.)$ correlates with the similarity of two vectors in the embedding space. Based on the formulated objectives, you want to maximize $h(\mathbf{x}_m, \mathbf{x})\ \forall m$ while minimizing $h(\mathbf{x}_l, \mathbf{x}_m) \forall l \neq m$. Thus, the parameters of NeuTraL AD, $\mathrm{W}, \mathrm{W}_1, \ldots, \mathrm{W}_{\mathrm{M}}$ are optimized by minimizing the following *deterministic contrastive loss*:

$$\mathcal{L} := \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_n^{train}}\Big[-\sum_{m=1}^{\mathrm{M}} \log \frac{h(\mathbf{x}, \mathbf{x}_m)}{h(\mathbf{x}, \mathbf{x}_m) + \sum_{l \neq m} h(\mathbf{x}_m, \mathbf{x}_l)}\Big]$$

The numerator encourages objective (i) while the denominator pushes towards (ii). The loss can itself be used as an anomaly score.

Other approaches relying on transformations have been proposed in recent studies. For instance, Tack et al., 2020 augment the SimCLR setting[1] (Chen et al., 2020) to suit the task of anomaly detection better. Tack et al., 2020 propose CSI, a contrastive learning method relying on distribution-shifting. Based on the diagnostic of Chen et al., 2020 that some families of transformation can sometimes degrade the discriminative performance of SimCLR when used for positive samples, the authors propose to use these distribution shifting transformations as negative examples. Formally, contrary to equation (2) in which augmentations of a sample $\mathbf{x}_i$ are considered as positive examples, authors propose to use the distribution shifting augmentations, denoted $\mathscr{S}$, as negative samples for $\mathbf{x}_i$. Note that $\mathscr{S}$ also contains the identity function. This gives the *contrasting shifted instances* (con-SI) loss used:

$$\mathscr{L}_{con-\mathrm{SI}} := \mathscr{L}_{\mathrm{S}imCLR}(\bigcup_{\mathrm{S}\in\mathscr{S}} \mathscr{B}_{\mathrm{S}}; \mathscr{T}), \text{ where } \mathscr{B}_{\mathrm{S}} := \{\mathrm{S}(\mathbf{x}_i)\}_{i=1}^{\mathrm{B}} \tag{14}$$

Intuitively, each distribution-shifted sample is regarded as an anomaly with respect to the original. Thus, this loss pushes towards discriminating between samples from $p(\mathbf{x} \mid y = 0)$ and samples that were transformed with any distribution shift in $\mathscr{S}$ except for the identity function. A supplementary layer is added after the feature extractor, which is used for a classification task to predict which transformation from $\mathscr{S}$ was applied. To flag anomalies, authors propose using a custom anomaly score based on the contrastive representation and the transformation prediction as in GOAD (Bergman & Hoshen, 2020).

Shenkar and Wolf, 2022 propose a self-supervised methodology that detects anomalies in a dataset through maximizing mutual information between elements of the features of a sample $\mathbf{x}_{\mathrm{I}} \in \mathscr{X}$ using contrastive learning. Given a sample $\mathbf{x}_i \in \mathscr{X} \subseteq \mathbb{R}^d$, consider the set of pairs of subvector of $\mathbf{x}_i$, $\{(a_i^j, b_i^j)\}_{j=1}^m$ where a_i^j is a vector of k consecutive features from $\mathbf{x}_i$ and b_i^j is the vector of all other features.

$$\mathbf{x}_i = (\underbrace{\mathbf{x}_i^1, \mathbf{x}_i^2, \ldots, \mathbf{x}_i^{k-1}, \mathbf{x}_i^k}_{a_i^1}, \underbrace{\mathbf{x}_i^{k+1}, \ldots, \mathbf{x}_i^d}_{b_i^1})$$

and $a_i^2 = (\mathbf{x}_i^2, \ldots, \mathbf{x}_i^{k-1}, \mathbf{x}_i^k, \mathbf{x}_i^{k+1}), b_i^2 = (\mathbf{x}_i^1, \mathbf{x}_i^{k+2}, \ldots, \mathbf{x}_i^d)$ etc. Thus, there are $m = d - k + 1$ pairs in total. Given such pairs, two feature extractors are learned $\mathrm{F} : \mathbb{R}^{d-k} \to \mathscr{Z}$ and $\mathrm{G} : \mathbb{R}^k \to \mathscr{Z}$, $\mathscr{Z} \subseteq \mathbb{R}^p$. Both feature extractors are trained so as to maximize mutual information between pairs (a_i^j, b_i^j). In inference, the anomaly score is the loss minimized in the training phase. This method is particularly interesting since it involves a simple but straightforward interpretability feature. Since a sliding window is used, it allows us to identify the input vector that most contributes to the high anomaly score.

Contrastive Anomaly Detection Approach using Pretrained Models Another approach was also recently considered to detect anomalies, which relies on adapting an original feature extractor f_0 to the One-Class Classification task using contrastive learning. It was first proposed in PANDA (Reiss et al., 2021), where authors propose to use pretrained deep features and combine them with simple anomaly detection methods. Using pretrained deep features has shown its efficacy in a broad range of applications, especially in the field of computer vision (Zhang et al., 2018). In PANDA, authors propose a three-stage framework: (i) include a pretrained feature extractor on an auxiliary task, (ii) feature adaptation so that the pretrained feature extractor better suits the one-class classification framework, and (iii) scoring, i.e., extract the feature representation to compute an anomaly score.

[1] We report the reader to section A.2 in the appendix for more detail on SimCLR.

They propose a simple pipeline to compare to other more sophisticated methods: they use a large ResNet pretrained on the ImageNet dataset and take the final pooling layer as a feature representation of an image. For the feature adaptation stage, authors rely on the loss in equation (9) but without the second regularization term, first proposed in the Deep-SVDD framework (Ruff et al., 2018). Regarding scoring, they rely on the average distance to the KNN normal image as detailed in equation (19).

In the spirit of the pretraining framework of Sohn et al., 2021, Reiss and Hoshen, 2023 propose to adapt the contrastive loss objective to modify a pretrained feature extractor $f : \mathscr{X} \to \mathscr{Z}$ to better suit the one-class classification task of anomaly detection models. Based on a geometric diagnostic, the authors propose a small but significant change to the contrastive loss. To minimize the contrastive loss given in equation (1), the angles between representations of negative pairs $\mathbf{x}, y \in \mathscr{D}_{train}$ need to be maximized although they might belong to the same class (e.g., normal). As discussed in their paper, authors put forward that by increasing the angle the distance to the center, $c := \mathbb{E}_{\mathscr{D}_{train}}[f(\mathbf{x})]$ also increases. Thus, this goes in opposition with one of the critical objectives considered in one-class classification tasks: distance minimization to the center for samples belonging to the normal class (see equation (9) for the Deep-SVDD objective). This phenomenon is illustrated in figure 2.3 taken from Reiss and Hoshen, 2023. To alleviate this issue, authors propose the *Mean-Shifted Contrastive*

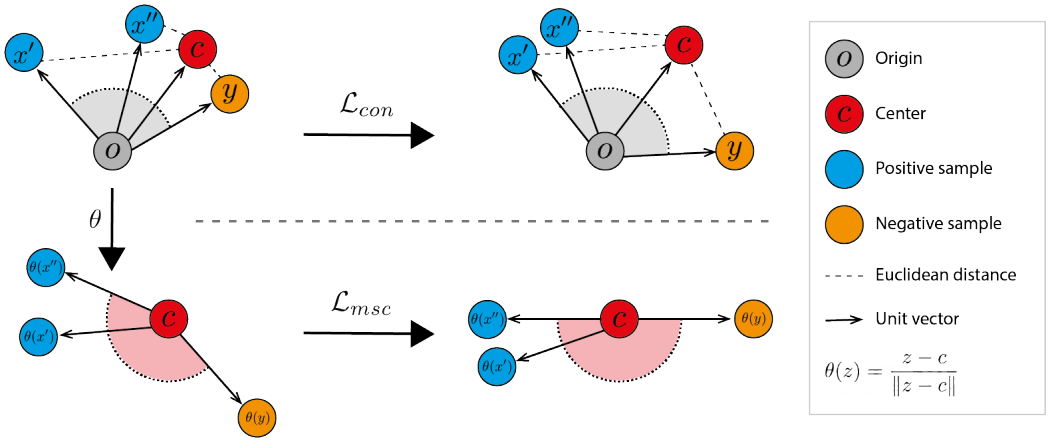


Figure 2.3: Geometric illustration of contrastive learning (figure taken from Reiss and Hoshen, 2023)

Loss that allows for both maximized angle and minimized distance to the center objectives to be compatible in the contrastive framework. The core idea of their new loss is to maximize the angle between negative pairs while maintaining the distance to the normalized center. To achieve such an objective, consider the vanilla contrastive learning pipeline. First, a sample $\mathbf{x} \in \mathscr{D}_{train}$ is augmented to obtain a positive pair $(\mathbf{x}', \mathbf{x}'')$, then both augmented samples are passed through the feature extractor and ℓ_2 normalized to be contained in the unit sphere. Let us denote the ℓ_2 normalized center as

$$\tilde{c} := \frac{\mathbb{E}_{\mathscr{D}_{train}}[f(\mathbf{x})]}{\left\|\mathbb{E}_{\mathscr{D}_{train}}[f(\mathbf{x})]\right\|_2}$$

and the obtained normalized feature representations as

$$\phi(\mathbf{x}) := \frac{f(\mathbf{x})}{\left\|f(\mathbf{x})\right\|_2}, \ \mathbf{x} \in \mathscr{X}$$

The authors propose to consider the mean-shifted normalized representation as

$$\theta(\mathbf{x}) := \frac{\phi(\mathbf{x}) - \tilde{c}}{\|\phi(\mathbf{x}) - \tilde{c}\|_2}, \ \mathbf{x} \in \mathcal{X} \tag{15}$$

This then allows them to define the *Mean-Shifted Contrastive Loss* as the contrastive loss in equation (1) applied to the mean-shifted representations:

$$\mathcal{L}_{msc}(\mathbf{x}, \{\mathbf{x}_+\}, \{\mathbf{x}_-\}) = \frac{1}{|\{\mathbf{x}_+\}|} \log \frac{\sum_{\mathbf{x}' \in \{\mathbf{x}_+\}} \exp(sim(\theta(\mathbf{x}), \theta(\mathbf{x}'))/\tau)}{\sum_{\mathbf{x}' \in \{\mathbf{x}_+\} \cup \{\mathbf{x}_-\}} \exp(sim(\theta(\mathbf{x}), \theta(\mathbf{x}'))/\tau)} \tag{16}$$

Since this loss is used in a one-class classification framework where $\mathcal{D}_{train}$ is solely composed of normal samples, this loss will force normal samples' representations to be uniformly spanned across the unit sphere centered around the mean of the normal data. Thus, this will foster anomaly discrimination as seen in figure 2.4 taken from Reiss and Hoshen, 2023.

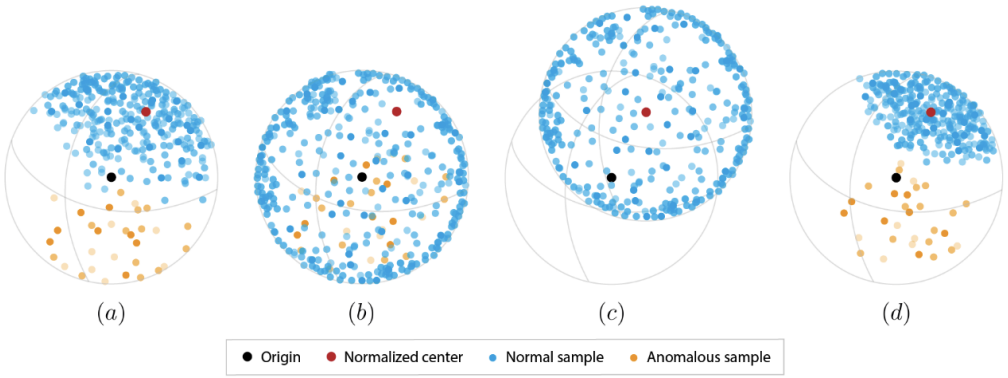


Figure 2.4: Comparison between $\mathcal{L}_{cont}$ and $\mathcal{L}_{msc}$ (figure taken from Reiss and Hoshen, 2023)

Also note that, although the loss defined in equation (16) should not increase the distance to $\tilde{c}$ it may still encourage such behavior, especially when the distance to $\tilde{c}$ is small. Thus, to mitigate this issue, authors propose to learn the weights of the feature extractor using $\mathcal{L}_{msc}$ augmented by an angular loss, which encourages the angular distance between the center and the sample to be minimal. The angular loss as defined in Reiss and Hoshen, 2023 is as follows:

$$\mathcal{L}_{angular} = -\phi(\mathbf{x}).\tilde{c} \tag{17}$$

In the end, the feature extractor's weights are obtained by minimizing

$$\mathcal{L}_{final} = \mathcal{L}_{angular} + \mathcal{L}_{msc} \tag{18}$$

In such a framework, authors then propose to use an anomaly score based on KNN and the cosine distance

$$Score(\mathbf{x}) = \sum_{\phi(y) \in \mathrm{N}_k(\mathbf{x})} 1 - \phi(\mathbf{x}).\phi(y) \tag{19}$$

where $\mathrm{N}_k(\mathbf{x})$ is the k nearest features to $\phi(\mathbf{x})$ in $\{\phi(\mathbf{x})\}_{\mathbf{x} \in \mathcal{D}_{train}}$.

2.3 Deep Learning for Tabular Data

Deep learning has gained more attention from researchers and practitioners due to its success in various tasks involving unstructured data. However, the heterogeneous type of tabular data makes standard architectures inefficient compared to gradient-boosted decision trees (GBDT), as later discussed in chapter 5. Deep models require lengthy investigations to determine the optimal architecture and hyperparameters, and it takes significant time to train them. In comparison, XGBoost (Chen & Guestrin, 2016) or LightGBM (Guolin et al., 2017), which are the go-to methods for classification on tabular data, can be trained and used to make predictions significantly faster than most deep models. Nevertheless, recent works have investigated the question of finding effective training procedures and novel architectures to try to outperform GBDTs on tabular data. Recent advancements in the field can be categorized into three categories:

- Improved/modified training procedures,
- Representation learning for tabular data,
- Novel architectures

2.3.1 Improved training procedures

Hollmann et al., 2023 put forward TabPFN, a Bayesian learning approach tailored for tabular data, based on Prior-data fitted networks (PFNs) as first proposed in Müller et al., 2022. Although displaying strong performance with limited training computational cost, TabPFN is limited to small datasets ($\leq 1,000$ samples).

Shavitt and Segal, 2018 put forward Regularization Learning Networks (RLNs) and show the crucial role of regularization for learning classification tasks on tabular data using deep networks. Regularization often designates the concept of minimizing a loss $\mathscr{L}$ while putting a constraint on the weights of the trained network, as shown in equation (20).

$$\min_{\theta} \tilde{\mathscr{L}}(Z, W, \lambda) = \mathscr{L}((\mathbf{x}_i, y_i), \theta) + \exp(\lambda) \sum_{i=1}^{n} \|\theta_i\| \tag{20}$$

where $\Theta = \{\theta_i\}_{i=1}^{n}$ the weights of the model, $\mathscr{L}$ a differentiable loss and a regularization coefficient λ. To account for the variability between features, Shavitt and Segal, 2018 propose to make λ vary between each weight in the network,

$$\min_{\theta} \mathscr{L}^{\dagger}(Z, W, \lambda) = \mathscr{L}((\mathbf{x}_i, y_i), \theta) + \sum_{i=1}^{n} \exp(\lambda_i) \|\theta_i\|. \tag{21}$$

Requiring to set λ_i as a hyperparameter for each weight in a network is untractable. Moreover, minimizing $\mathscr{L}^{\dagger}$ w.r.t. W and $\Lambda = \{\lambda\}_{i=1}^{n}$ gives a trivial solution for Λ; $\Lambda = \{-\infty\}_{i=1}^{n}$. Hence, the authors introduce a counterfactual loss that allows circumventing this problem: this loss is used to update the value of Λ at each training step, while $\mathscr{L}^{\dagger}$ is evaluated given the estimate Λ to update the weights θ. This training procedure is referred to as Regularization Learning Networks. By using this approach, Shavitt and Segal, 2018 obtain strong performance on a tabular dataset benchmark and compete with GBDTs.

Kadra et al., 2021 extends the work of Shavitt and Segal, 2018 and shows how regularization can be essential for neural networks to learn and generalize on tabular data efficiently. Their approach consists in training a vanilla Multilayer Perceptron (MLP) combined with a well-selected *cocktail* of regularization techniques. To train MLPs, the authors select their regularization cocktail among 13 regularization technics in several categories: weight decay, data augmentations (e.g., Cut-Out (DeVries & Taylor, 2017) which masks some features), ensemble methods (e.g., dropout), structural regularization (e.g., skip-connections) and implicit regularization (e.g., early stopping). Their approach involves selecting the optimal subset of regularization techniques using the multi-fidelity Bayesian optimization method BOHB (Falkner et al., 2018). Using a significant tabular data benchmark comprised of 40 datasets, they show that by selecting the proper regularization techniques, MLPs can significantly outperform most competing deep architectures and gradient-boosted decision trees on a significant share of the tested datasets.

2.3.2 Representation Learning for Tabular Data

Representation learning consists in finding a transformation of the input data into a new feature space where relevant information is preserved or enhanced while noise and irrelevant details are filtered out or minimized. To that end, self-supervised approaches have become prevalent in the field, including contrastive and non-contrastive approaches as discussed in section 2.2.

Representation learning for tabular data has caught increasing attention in recent years. For instance, in SAINT (Somepalli et al., 2021), the authors propose a constrastive pretraining procedure to foster the performance of their neural network model. For each sample $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$, they rely on an embedding layer to obtain a representation $\mathbf{E}(\mathbf{x}_i) \in \mathbb{R}^{(d+1)\times e}$, where the supplementary row dimension comes from a `[CLS]`[2] token added in the original feature vector. This embedding layer is similar to what we describe in section 4.4.4. For their contrastive pretraining, the positive pair for a sample $\mathbf{x}_i$ is obtained by combining MixUP (Zhang et al., 2017) in the embedding space and CutMix (Yun et al., 2019) in the raw data space.

$$\mathbf{x}'_i = \mathbf{x} \odot \mathbf{m} + \mathbf{x}_a \cdot (1 - \mathbf{m}) \qquad \text{(CutMix (Yun et al., 2019))}$$

$$\mathbf{E}(\mathbf{x}_i)' = \alpha * \mathbf{E}(\mathbf{x}'_i) + (1 - \alpha) * \mathbf{E}(\mathbf{x}'_b), \qquad \text{(MixUp (Zhang et al., 2017))}$$

where $\mathbf{x}_a, \mathbf{x}_b$ are random samples from the current batch, $\mathbf{x}'_b$ is the CutMix version of $\mathbf{x}_b$, $\mathbf{m}$ is binary mask vector sampled from a Bernoulli distribution with probability p_{cutmix}, and α is the mixup parameter. Both $\mathbf{E}(\mathbf{x}_i)$ and $\mathbf{E}(\mathbf{x}_i)'$ are passed through the SAINT model (discussed in section 2.3.3) denotes $\mathbf{S}(\cdot)$ and two separate projection heads $g_1(\cdot)$ and $g_2(\cdot)$,

$$z_i = g_1(\mathbf{S}(\mathbf{E}(\mathbf{x}_i))), \quad z'_i = g_2(\mathbf{S}(\mathbf{E}(\mathbf{x}_i)')),$$

where (z_i, z'_i) constitute a positive pair for sample i, all other representations in a batch are considered negative pairs for sample i. Two loss functions are considered to optimize the parameters:

- InfoNCE: views of the same data point (z_i and z'_i) are encouraged to be close, while different points are pushed far away (z_i and z_j).

[2]These classification tokens are widely used to rely on transformers to perform classification tasks.

- Denoising loss: predict the original data from the noisy one. Let $\mathbf{S}(\mathbf{E}(\mathbf{x}_i)') = \mathbf{r_i}'$, one reconstructs the original features using a separate MLP for each feature. The objective is to minimize the difference between the original $\mathbf{x_i}$ and its reconstruction.

The overall pretraining loss is constructed as follows,

$$\mathscr{L}_{\text{pre-training}} = \underbrace{-\sum_{i=1}^{m} \log \frac{\exp(z_i.z_i'/\tau)}{\sum_{k=1}^{m} \exp(z_i.z_k'/\tau)}}_{\text{Contrastive Loss}} + \lambda_{pt} \underbrace{\sum_{i=1}^{m} \sum_{j=1}^{d} \{\mathscr{L}_j(\text{MLP}(\mathbf{r_i'}), \mathbf{x_i})\}}_{\text{Denoising Loss}} \tag{22}$$

where $\mathscr{L}_j$ is the CE loss or the mean squared loss, depending on whether the feature j is categorical or continuous.

Other representation learning methods include STab (Hajiramezanali et al., 2022), an augmentation-free self-supervised representation learning method for tabular data. The core idea in the proposed approach is to replace applying augmentations over an input sample to create different views of that sample by applying augmentation to the layers of an encoder. Formally, let $\mathbf{x} \in \mathbf{R}^d$ be a sample, let f_1 and f_2 be two MLP networks that share the same weights, and let g be an MLP that transforms the output of each encoder to match it with the output of the other. For f_1 and f_2 to produce different outputs, they are regularized by imposing dynamic sparsity in the model. Let $\mathrm{H}^{(i)} = \{h_j^{(i)}\}_{j=0}^{\mathrm{L}}$, $h_0^{(1)} = h_0^{(2)} = x$ the outputs for view $i \in \{1,2\}$ for each of the L layers in the network. For each layer of the encoder, the output is given by

$$h_j^{(i)} = \sigma\left((\mathrm{M}_j^{(i)} \odot \mathrm{W}_j) h_{j-1}^{(i)}\right),$$

where $\mathrm{M}_j^{(i)}$ is a binary matrix encoding the connection in f_i sampled using a bernouilli distribution, $\mathscr{W} = \{\mathrm{W}_j\}_{j=0}^{\mathrm{L}}$ are the shared weights by the encoders, and $\sigma(\cdot)$ is a non-linear activation function. Let us denote the output vectors for a sample $\mathbf{x}$ by $y_1 = g(f_1(\mathbf{x}))$, and $z_2 = f_2(\mathbf{x})$, authors rely on the negative cosine distance a measure of similarity,

$$\mathrm{D}(y_1, z_2) = -\frac{y_1}{\|y_1\|_2} \cdot \frac{z_2}{\|z_2\|_2},$$

where $\|\cdot\|_2$ is the ℓ_2-norm. The weights of the networks are updated by optimizing the following symmetric loss,

$$\mathscr{L} = \frac{1}{2}\mathrm{D}(y_1, z_2) + \frac{1}{2}\mathrm{D}(y_2, z_1). \tag{23}$$

Another notable approach is Scarf (Bahri et al., 2022), a simple method based on contrastive learning in which different views of a sample are obtained by corrupting a random subset of features. Let $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_d) \in \mathbb{R}^d$ be a sample where $\mathbf{x}_j$ represent the value of feature j for sample $\mathbf{x}$. Scarf consists in uniformly sampling features and replacing their values in $\mathbf{x}$ by drawing from the features' empirical marginal distributions. The latter is the uniform distribution over the values this feature takes in the training datasets. The corrupted sample, denoted $\tilde{\mathbf{x}}$, and the original sample $\mathbf{x}$ are then passed from an encoder network f to be later used in the supervised task, followed by a pre-train *only* head network g, to obtain respectively $\tilde{\mathbf{z}}$ and $\mathbf{z}$.

$$\mathbf{z} = g \circ f(\mathbf{x})$$

where g will be later discarded to only keep the encoder f, the networks' weights are optimized by

minimizing the InfoNCE constrastive loss, given as the first term in equation (22).

Recent work has also investigated prototype-based representation learning for tabular data as PTaRL (Ye et al., 2024). In short, this method consists in constructing a prototype-based projection space (P-Space) to learn around global data prototypes, well-separated representations that are discriminative for supervised learning. Their two-stage approach consists of first generating K prototypes using k-means clustering (Hartigan & Wong, 1979) that will serve as basis vectors for their P-Space. Second, they project the samples into the P-Space to learn global data structure information.

Other works, such as XTab (Zhu et al., 2023) or UniTabE (Yang et al., 2023), include self-supervised representation learning for cross-table pretraining of tabular transformers.

In SwitchTab (Wu et al., 2024), the authors discuss the concepts of salient and mutual information and emphasize their crucial role in producing meaningful sample representations. The authors also argue that decoupling the two types is particularly challenging for tabular data, compared with unstructured data, due to the inherent heterogeneous structure of tabular data. For image data, the background pixels would constitute the mutual information that is shared across samples. The arrangement of light and dark pixels represents the distinctive features and constitutes the salient information conveyed in an image. This distinction in the tabular domain is significantly more challenging to construct. In this approach, the authors put forward a method that aims to construct its representation by fostering the decoupling between the salient and mutual information contained in a sample. Their approach relies on an asymmetric encoder-decoder structure and produces a general embedding later projected into salient and mutual embedding spaces.

2.3.3 Novel Deep Architecture for Tabular Data

Some approaches to applying neural networks on structured tabular data have involved proposing ad-hoc architectures tailored to account for the particular heterogeneous structure of this data type.

In FT-Transfomer, Gorishniy et al., 2021 propose a first adaptation of the Transformer model for tabular data through feature tokenizing to make it usable for transformer models to process it. Similarly, Arik and Pfister, 2021 introduces TabNet, an interpretable deep learning architecture for tabular data. Their approach relies on sequential attention to select the feature from which to *reason* as each decision step. Other works, such as Kossen et al., 2021, introduce Non-Parametric Transformers that leverage sample-sample, sample-label, feature-feature, and feature-target dependencies. Their architecture relies on the scaled dot-product attention as first proposed in Vaswani et al., 2017. It describes a mapping between queries $\mathrm{Q}_i \in \mathbb{R}^{1\times h_k}$, keys $\mathrm{K}_i \in \mathbb{R}^{1\times h_k}$ and values $\mathrm{V}_i \in \mathbb{R}^{1\times h_v}$ to an output. The output is computed as a weighted sum of the values, where each weight is obtained by measuring the compatibility between queries and keys. Take $\mathbf{Q} \in \mathbb{R}^{n\times h_k}$, $\mathbf{K} \in \mathbb{R}^{m\times h_k}$ and $\mathbf{V} \in \mathbb{R}^{m\times h_v}$ the corresponding matrices in which queries, keys, and values are stacked. Scaled dot-product attention is computed as

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{h_k}}\right)\mathbf{V} \tag{24}$$

where, for convenience, one often sets $h_k = h_v = h$.

To foster the ability of a model to produce diverse and powerful representations of data samples, one often includes several dot-product attention mechanisms. Multi-head dot-product attention then

describes the concatenation of k independent attention heads:

$$\text{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \underset{\text{axis=k}}{\text{concat}}(\text{O}_1,\ldots,\text{O}_k)\text{W}^{\text{O}} \tag{25}$$

$$\text{where, } \text{O}_j = \text{Attention}(\mathbf{Q}\text{W}_j^{\text{Q}},\mathbf{K}\text{W}_j^{\text{K}},\mathbf{V}\text{W}_j^{\text{V}}) \tag{26}$$

where the embedding matrices $\text{W}_j^{\text{Q}},\text{W}_j^{\text{K}},\text{W}_j^{\text{V}} \in \mathbb{R}^{h\times h/k}$ are learned for each attention head $j \in \{1,\ldots,k\}$ and $\text{W}^{\text{O}} \in \mathbb{R}^{h\times h}$ serves to mix the h attention heads outputs. NPTs only include multi-head *self*-attention mechanisms, which consist in multi-head dot-product attention where queries, keys, and values are identical:

$$\text{MHSelfAtt}(\mathbf{H}) = \text{MultiHead}(\mathbf{Q}=\mathbf{H},\mathbf{K}=\mathbf{H},\mathbf{V}=\mathbf{H}) \tag{27}$$

As described in (Kossen et al., 2021), NPT follows transformer best practices to improve performances and involves a residual branch as well as layer normalization (LN) (Ba et al., 2016) before MHSelfAtt(.).

$$\text{Res}(\mathbf{H}) = \mathbf{H}\text{W}^{\text{res}} + \text{MHSelfAtt}\left(\text{LN}(\mathbf{H})\right) \tag{28}$$

where $\text{W}^{\text{res}} \in \mathbb{R}^{h\times h}$ are learned weights. Layer normalization is also added after the residual branch as well as a row-wised feed-forward network (rFF):

$$\text{MHSA}(\mathbf{H}) = \text{Res}(\mathbf{H}) + \text{rFF}\left(\text{LN}\left(\text{Res}\left(\mathbf{H}\right)\right)\right) \in \mathbb{R}^{n\times h} \tag{29}$$

NPT involves both attention between features and attention between samples, thus allowing the ability to capture feature-feature and sample-sample dependencies. More precisely, two mechanisms involved in NPTs allow anomalies to be identified: Attention Between Datapoints (ABD) and Attention Between Attributes (ABA).

NPT receives as input a representation $\mathbf{H}^0 \in \mathbb{R}^{n\times d\times e}$. A sequence of MHSA layers is applied to the input, alternating between ABA and ABD. The model then outputs a prediction for masked features while keeping unmasked features unchanged $\hat{\mathbf{X}} \in \mathbb{R}^{n\times d}$.

Attention Between Datapoints (ABD) It is the key feature that differentiates NPT from standard transformer models. This mechanism captures pairwise relations between data samples. Consider as an input to the ABD layer the previous layer representation $\mathbf{H}^{(\ell)} \in \mathbb{R}^{n\times d\times e}$ flattened to $\mathbb{R}^{n\times h}$ where $h = d\cdot e$. Then, NPT applies MHSA, as seen in equation (29), between the data samples flattened representations $\{\mathbf{H}_i^{(\ell)} \in \mathbb{R}^{1\times h} | i \in 1,\ldots,n\}$.

$$\text{ABD}(\mathbf{H}^{(\ell)}) = \text{MHSA}(\mathbf{H}^{(\ell)}) = \mathbf{H}^{(\ell+1)} \in \mathbb{R}^{n\times h} \tag{30}$$

After applying ABD, the data representation is reshaped to its original dimension in $\mathbb{R}^{n\times d\times e}$.

Attention Between Attributes (ABA) As already discussed, NPT alternates between ABD and ABA layers. ABA layers should help learn per data sample representation for the inter-sample representations. In contrast with ABD, ABA consists in applying MHSA independently to each row in $\mathbf{H}^{(\ell)}$, *i.e.* to each data sample's intermediate representation $\mathbf{H}_i^{(\ell)} \in \mathbb{R}^{d\times e}, i \in \{1,\ldots,n\}$.

$$\text{ABA}(\mathbf{H}^{(\ell)}) = \underset{\text{axis=n}}{\text{stack}}\left(\text{MHSA}(\mathbf{H}_1^{(\ell)}),\ldots,\text{MHSA}(\mathbf{H}_n^{(\ell)})\right) \tag{31}$$

Other work, such as SAINT (Somepalli et al., 2021), have adopted a similar approach and leveraged the same dependencies. SAINT is composed of L identical stages. Each stage comprises a self-attention block (k attention heads) and an intersample attention block. Self-attention blocks attend to individual features within each sample, while intersample attention allows rows to attend to other rows (i.e., other data samples) in the matrix. Both blocks resemble the ABA and ABD mechanisms introduced in (Kossen et al., 2021).

Other works have proposed using language model-like architectures TP-BERTa (Yan et al., 2024). Yan et al., 2024 put forward the **T**abular **P**rediction adapted **BERT a**pproach (TP-BERTa) a pre-trained language model (LM) targeted explicitly for tabular data classification and regression tasks. Their approach is based on RoBERTa (Liu et al., 2019) and adapts it to allow the model to grasp the particular structure of numeric features. They introduce the concept of relative magnitude tokens (RMT) to discretize numerical feature values so that the LM perceives the relative value magnitudes in the language space. They also put forward the intra-feature attention model to involve the feature names as input to the LM to preserve the semantic signal of the feature name.

Chapter 3

Relying on Intersample Relations: Measuring Influence for Anomaly Detection

Abstract

As with many other tasks, neural networks are very effective for anomaly detection. However, very few deep-learning models are suited for detecting anomalies on tabular datasets. This work proposes a novel methodology to flag anomalies based on `TracIn`, an influence measure initially introduced for explicability purposes. The proposed methods can serve to augment any deep anomaly detection method. We test our approach using Variational Autoencoders and show that the average influence of a subsample of training points on a test point can serve as a proxy for abnormality. This supports the idea that leveraging inter-sample relations can be relevant to detecting anomalies in some datasets. Our model proves competitive compared to state-of-the-art approaches: it achieves comparable or better detection accuracy on medical and cyber-security tabular benchmark data.

This chapter is a modification of the following publication:

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2022). TracInAD: Measuring influence for anomaly detection. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–6. https://doi.org/10.1109/IJCNN55064.2022.9892058 (cited on pages 48, 78)

3.1 Introduction

As a research direction, anomaly detection has caught more and more attention in recent years due to its many successful applications across many domains. This field of research bears several names that point towards relatively similar methods and approaches (Ruff et al., 2021): outlier, novelty, and anomaly detection. The difference in the pursued objectives can account for the dissimilarity in semantics.

Scholars often consider its utility to be two-fold. On the one hand, it is well-suited for fraud and intrusion detection problems that involve highly imbalanced datasets for which standard supervised

learning methods often fail. Most of these supervised approaches fail when over-sampling and under-sampling cannot be efficiently applied, which is often the case for tabular datasets. On the other hand, anomaly detection has also proven effective when very few or no labels are available.

In short, scholars have defined anomaly detection as the process of identifying points that deviate from a pre-defined notion of normality. Examples of anomaly detection include flagging fraudulent payments among standard credit card payments, identifying mislabeled samples within a dataset, or detecting computer intrusion.

The literature investigates two approaches to anomaly detection. The traditional unsupervised approach involves giving classifiers anomalous and normal samples as inputs in the training phase. A more hegemonic approach to anomaly detection involves learning or characterizing a distribution during training by considering a training set only composed of normal data. Anomalies are then identified in the inference stage by evaluating how well each sample fits the estimated distribution.

Many researchers have recently proposed different methods to tackle the critical problem of detecting anomalies. Most of today's state-of-the-art methods rely on deep learning models such as Deep SVDD (Ruff et al., 2018), which mimic the canonical approach of SVDD (Tax & Duin, 2004) without using the computationally costly kernels. More recently, methods that rely on self-supervision, such as (Bergman & Hoshen, 2020), which uses pretext tasks, or (Qiu et al., 2021), which involves a contrastive loss, have also proven to perform well. Except for GOAD (Bergman & Hoshen, 2020), NeuTraL-AD (Qiu et al., 2021), and the method of Shenkar and Wolf, 2022, most proposed methods focus on applications related to image or textual datasets.

This chapter introduces a novel approach to anomaly detection based on influence measures. Measuring influence here describes the task of evaluating how much a sample contributed to increasing or decreasing the loss of another sample during training. Our methodology relies on `TracIn` proposed by Garima et al., 2020 to evaluate the average influence of a subsample of training points on a sample. Following up on Kong and Chaudhuri, 2021 who showed that self-influence in the unsupervised setup is correlated with the loss of the sample, we propose a standalone method based on Variational Autoencoders (Kingma & Welling, 2014), which outperforms or shows comparable results with state-of-the-art anomaly detection methods. This work emphasizes that sample-sample relations can help detect anomalies on tabular data, as exemplified by KNN methods for anomaly detection.

The main contributions involved in our methodology are:

- A novel anomaly detection method based on influence measures that offers competitive results on several benchmark datasets.
- We demonstrate that the proposed approach can be extended to any deep anomaly detection method.

3.2 Illustrative Example

Before presenting formally the method proposed in the present chapter, let us give an introductory example to discuss the main concepts used in the proposed method.

Consider a simple binary classification task in which a supervised classifier learns to classify from an image whether it contains a `dog` or a `cow`. A possible approach to this problem would be to con-

struct a training set, $\mathcal{D}_{train}$ composed of images that belong to both classes and to train a classifier f_θ to recognize to which class a sample belongs. This classifier's parameters θ are obtained by minimizing a loss function on the training set. During training, the classifier will see images of both classes, formulate a decision for each image, evaluate whether its classification was correct or incorrect, and adapt its weights depending on the output. If properly trained, the model should be able to identify each class's main semantic characteristics, enabling it to discriminate between the two classes when faced with an unknown image.

During training, samples that are close to one another will likely contribute to reducing the loss of each other. In other words, pictures of dogs will likely help the classifier better classify images of dogs. Similarly, images of cows that may be confused with dogs can have the opposite effect, disrupting the ability to learn to optimally separate the classes.

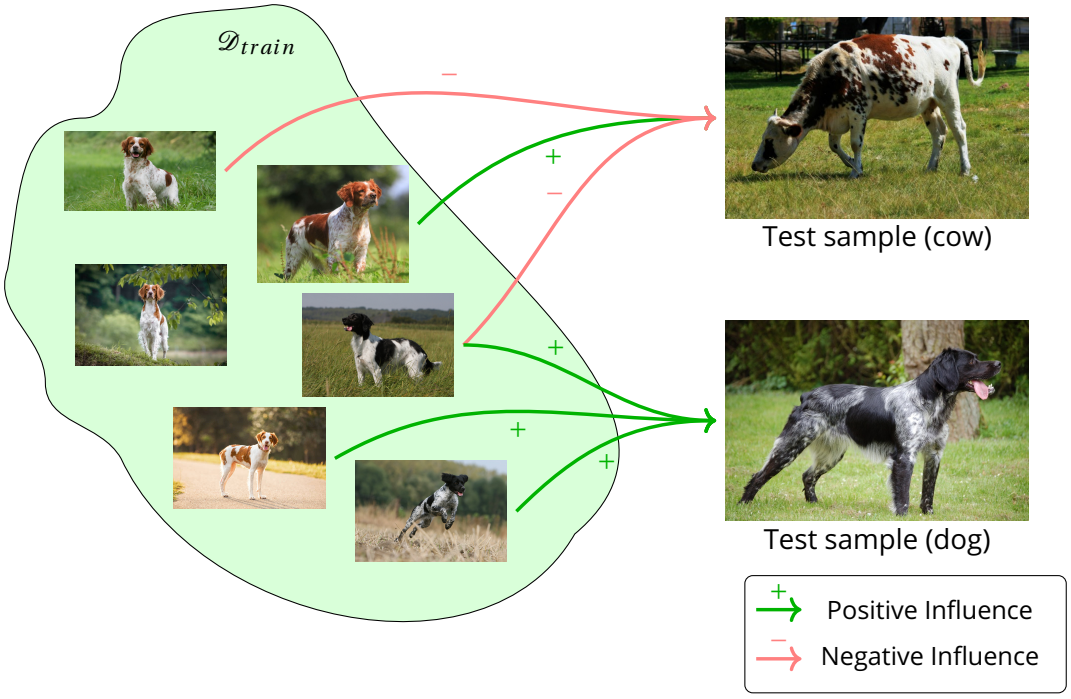


Figure 3.1: Influence relations between samples belonging to the same distribution (dog-to-dog) and different distributions (dog-to-cow). Overall, one expects the average influence to be positive (→) for dog-to-dog, i.e., that training samples, on average, contributed to reduce the loss of a sample belonging to the same distribution. On the contrary, one excepts average influence to be negative (→) for dog-to-cow or at least lower than for dog-to-dog.

Returning to our class of problems, i.e., anomaly detection, consider a similar situation where an autoencoder is trained exclusively on images of dogs to reconstruct them. By seeing exclusively images of dogs during training, some close images (e.g., images of dogs of the same breed) will likely contribute to improving the capacity of the model to reconstruct an unseen image close to these existing ones in the training set. In other words, the images in the training set of dogs of a particular breed will likely contribute to reducing the loss of an image of a dog of the same breed. On the contrary, unseen images during training that belong to a different distribution, e.g., images of a cow, will likely display a higher loss (be less correctly reconstructed) and a more erratic relation to training samples regarding loss decrease or increase. Some images of dogs may contribute to decreasing the loss of the image of a cow, for example, if both images display an animal in a pasture or if the fur of

both animals are close to one another, as shown in figure 3.1 in which a picture of a dog is somewhat similar to the one of the cow. However, as loss is likely higher for images from a different distribution, the contribution of the training samples to the loss reduction of the image of the cow will be more chaotic, and fewer samples should have contributed to reducing it.

This difference in loss relation can be captured by the concept of **influence**, formally defined in sections 3.3 and 3.4.1. In short, the influence between two samples measures the extent to which the former contributed to reducing the loss of the latter. Leveraging this concept can help identify anomalies by training a model on a *normal* class and computing an average influence measure between a sample of interest and the samples in the training set. On average, the influence score of normal training samples (`dogs`) on a test sample should be higher if this sample is also a dog but lower and even negative if the sample stems from a different distribution (`cow`). This simply captures the fact that more images contributed to reducing the loss of an image of a dog than the loss of an image of a cow. This influence relation is illustrated in figure 3.1.

3.3 Related Works

3.3.1 Anomaly Detection

Anomaly detection can be divided into the following non-exhaustive categories:

One-Class Classification One-class classification involves discriminative models for anomaly detection: such methods avoid the challenge of estimating the normal distribution to identify anomalies. Instead, those methods aim at characterizing the density by proposing a decision boundary. In short, One-Class Classification describes methods that use exclusively normal data in the training stage to learn a decision function to flag anomalies. This sub-field includes both shallow and deep anomaly detection methods. For instance, in OCSVM (Schölkopf et al., 1999) and SVDD (Tax & Duin, 2004), authors propose to learn a decision function in the form of a hyperplane and a hypersphere, respectively, in a Hilbert space by using the kernel trick. Regarding deep methods, Ruff et al., 2018; Ruff et al., 2020 put forward Deep-SVDD in which a deep neural network replaces kernels to map data points to a latent space so that normal points will be contained in a hypersphere. Deep-SVDD can suffer from *model collapse*, which designates a trivial mapping to a unique point in the latent space for certain network architectures. Thus, recent studies (Chong et al., 2020; Goyal et al., 2020) have proposed regularization methods to avoid such pitfalls.

Reconstruction based methods Reconstruction-based methods focus on learning a model that reconstructs normal instances well while failing to reconstruct abnormal points. Thus, the reconstruction error can serve as a proxy for anomaly: the higher the error, the higher the probability of a point being an anomaly. Such reconstruction-based methods encompass deterministic methods such as autoencoders, PCA, or probabilistic variants. For instance, PCA has been adapted to anomaly detection in Sharan et al., 2018 or Hawkins, 1974. Other methods have relied on autoencoders to estimate the normal data distribution, such as Kim et al., 2020 who augment the reconstruction error in the original dataspace with reconstruction error in the latent spaces generated by the encoder's layers.

Self-supervised methods In a large spectrum of applications, including anomaly detection, many methods now involve self-supervision to improve models' learning capacity. For instance, Bergman and Hoshen, 2020 propose to transform data samples and use the self-supervised pretext task of identifying which transformation was applied to a sample. Failure to correctly predict the applied transformation can serve as a proxy measuring anomaly. In a similar approach, Qiu et al., 2021 also rely on transformations to flag anomalies but propose to learn them instead of using affine transformation as in (Bergman & Hoshen, 2020). Another recent approach, (Shenkar & Wolf, 2022), fetches internal contrastive learning to flag anomalies on tabular data. Other recent work (Sohn et al., 2021) proposed a contrastive pretraining methodology to improve anomaly detection methods through a two-staged framework.

3.3.2 Influence Estimation

Related to our work, influence estimation has also attracted growing interest from the research community. It designates identifying the training samples most responsible for predicting a particular test sample x. This involves the computation of an influence score, which can take many different forms (Barshan et al., 2020; Basu et al., 2020; Garima et al., 2020; Yeh et al., 2018). Recently, Koh and Liang, 2017 proposed the influence function, which relies on the idea that if an influential sample for a particular test sample is removed from the set used for training, then the test sample's loss should significantly increase. Since estimating such a function may be hard, Garima et al., 2020 propose `TracIn` a computationally efficient first-order approximation to the influence function. As mentioned by the authors, such an influence function can be used to flag odd points in a dataset: mislabeled samples, outliers, or even anomalies. For instance, in an unsupervised setup, where they use β-Variational Autoencoders, Kong and Chaudhuri, 2021 analyze the behavior of the influence function. They show that, for points that stand out from the rest of the dataset, self-influence, which measures the influence of a sample on its loss/likelihood, tends to be larger than for normal points. Based on their diagnostic, we hypothesize that not only do self-influence behaviors differ between normal and abnormal points, but the influence of normal points on abnormal points should significantly differ from the influence of normal points on normal points.

In this chapter we propose using influence measures to flag anomalies among normal samples. We propose a standalone method based on a Variational Autoencoder, which shows that alone influence can be used efficiently for detecting anomalies.

3.4 Method

3.4.1 `TracIn`

Let us briefly discuss `TracIn` (Garima et al., 2020) as a means of measuring influence. Consider the following set up where $\mathcal{X} \subseteq \mathbb{R}^d$ represents the data space, and a dataset $\mathcal{D}_n = \{\mathbf{x}_j\}_{j=1}^n, \mathbf{x}_j \in \mathcal{X}$. Let f_θ denote a model, parametrized by $\theta \in \Theta \subseteq \mathbb{R}^p$, whose parameters are obtained through optimizing a loss function $\ell : \Theta \times \mathcal{X} \to \mathbb{R}$. The loss of the model parametrized by θ for a data point $\mathbf{x} \in \mathcal{X}$ is $\ell(\theta, \mathbf{x})$. For a training set $\mathcal{D}_{train} \subseteq \mathcal{D}_n$, the parameters of the model are obtained through minimizing $\sum_{\mathbf{x} \in \mathcal{D}_{train}} \ell(\theta, \mathbf{x})$.

Define the influence of a sample $\mathbf{x}$ on a test sample $\mathbf{x}'$ as the difference in the loss for the sample $\mathbf{x}'$ incurred by having included $\mathbf{x}$ in the training set. Formally, the influence function of a sample $\mathbf{x}$ on the test sample $\mathbf{x}'$ is:

$$\mathrm{IF}(\mathbf{x},\mathbf{x}') = \ell(\theta,\mathbf{x}') - \ell(\theta_{-\mathbf{x}},\mathbf{x}') \tag{1}$$

where $\theta_{-\mathbf{x}} = \arg\min_{\theta\in\Theta} \sum_{\mathbf{z}\in\mathcal{D}_{train}\setminus\{\mathbf{x}\}} \ell(\theta,\mathbf{z})$.

Consider an iterative optimization process, *e.g.* Stochastic Gradient Descent (SGD), where θ_t denotes the obtained parameters after iteration t, B_t a minibatch of size b at iteration t, and a step size η_t at iteration t. `TracIn` gives the following measure of influence of $\mathbf{x}$ on the test sample $\mathbf{x}'$ when SGD is the optimizer:

$$\texttt{TracIn}(\mathbf{x},\mathbf{x}') = \frac{1}{b}\sum_{t:\mathbf{x}\in\mathrm{B}_t} \eta_t \nabla_\theta \ell(\theta_t,\mathbf{x}')\cdot\nabla_\theta\ell(\theta_t,\mathbf{x}) \tag{2}$$

where $\nabla_\theta\ell(\theta_t,\mathbf{x}')$ denotes the gradient of the loss function evaluated for the sample $\mathbf{x}'$ w.r.t. the parameter θ at $\theta=\theta_t$.

Proof. Let us prove equation (2) when no minibatch is used and weights are updated using one sample at a time. Equation (2) can be shown using a first-order Taylor approximation. As the step size between the updates of the parameters along the training process is small, it is possible to approximate the change in the loss of a sample $\mathbf{x}$ at iteration t as

$$\ell(\theta_{t+1},\mathbf{x}) = \ell(\theta_t,\mathbf{x}) + \nabla_\theta\ell(\theta_t,\mathbf{x})\cdot(\theta_{t+1}-\theta_t) + \mathcal{O}(\|\theta_{t+1}-\theta_t\|^2). \tag{3}$$

Moreover, if the optimizer is set to be the stochastic gradient descent (SGD) using sample x_t, then θ_{t+1} can be expressed as

$$\theta_{t+1} = \theta_t - \eta_t\nabla_\theta\ell(\theta_t,\mathbf{x}_t).$$

Rearranging the terms, replacing in equation (3), and ignoring the term $\mathcal{O}(\|\theta_{t+1}-\theta_t\|^2)$,

$$\begin{aligned}\ell(\theta_{t+1},x) &\approx \ell(\theta_t,\mathbf{x}) + \nabla_\theta\ell(\theta_t,\mathbf{x})\cdot\left(-\eta_t\nabla_\theta\ell(\theta_t,\mathbf{x}_t)\right)\\ \ell(\theta_t,\mathbf{x}) - \ell(\theta_{t+1},\mathbf{x}) &\approx \eta_t\nabla_\theta\ell(\theta_t,\mathbf{x})\cdot\nabla_\theta\ell(\theta_t,\mathbf{x}_t)\end{aligned}$$

Now, to compute the overall reduction of the loss of sample $\mathbf{x}'$ induced by sample $\mathbf{x}$ in the course of training, one simply needs to account for the weight updates for iterations in which sample $\mathbf{x}$ was used. Formally

$$\sum_{t:\mathbf{x}_t=\mathbf{x}} \eta_t\nabla_\theta\ell(\theta_t,\mathbf{x}')\cdot\nabla_\theta\ell(\theta_t,\mathbf{x}).$$

□

One uses `TracInCP` given in (4) to avoid excessive computational overhead. It consists in storing checkpoints (CP) along the training process, *i.e.* parameters $\theta_{t_1},\theta_{t_2},\ldots,\theta_{t_k}$ corresponding to iteration $t_1,t_2,\ldots,t_k$ assuming that between checkpoints each training sample is visited only once, *e.g.* one epoch, and that the step size remains constant between checkpoints.

$$\texttt{TracInCP}(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{k} \eta_i\nabla_\theta\ell(\theta_{t_i},\mathbf{x}')\cdot\nabla_\theta\ell(\theta_{t_i},\mathbf{x}) \tag{4}$$

Algorithm 1 Pseudo Python Code for `TracInAD`

Require: $\mathscr{D}_{train}, \mathscr{D}_{val}, \{\theta_{t_1}, \ldots, \theta_{t_n}\}, \{\eta_{t_1}, \ldots, \eta_{t_n}\}$, $\ell(\theta, .), m$
  `TracInAD` $\leftarrow dict()$
  B $\leftarrow$ random sample of size m from $\mathscr{D}_{train}$
  for $t \in \{t_1, \ldots, t_n\}$ **do**
    $\theta \leftarrow \theta_t$
    $\eta \leftarrow \eta_t$
    for $\mathbf{x} \in \mathscr{D}_{val}$ **do**
      `TracInAD`$[x]$ += $\frac{1}{m} \sum_{\mathbf{x}' \in \mathrm{B}} \eta \nabla_\theta \ell(\theta, \mathbf{x}') \cdot \nabla_\theta \ell(\theta, \mathbf{x})$
    end for
  end for

3.4.2 Influence as an anomaly score: `TracInAD`

Consider a deep anomaly detection model f_θ whose parameters were optimized through minimizing a loss function $\ell(., \theta)$ as detailed in section 3.4.1, and a training set $\mathscr{D}_{train}$ solely composed of normal samples.

Under the hypothesis that the normal samples belonging to the training set and the validation set were obtained from a similar distribution p_{normal}, the overall influence of training samples on normal validation samples should be positive. In other words, training samples should mostly reduce the loss of normal samples. Using the terminology proposed in Garima et al., 2020, training samples should mostly be strong *proponents* for any normal sample in the validation set. On the other hand, since anomalies' distributions differ from p_{normal}, training points should mostly be *opponents*, in the sense that they contributed to increase the loss, or weak *proponents* to the anomalies in the test set.

Based on these premises, we propose to derive an anomaly score based on `TracInCP`. Consider a sample $\mathbf{x}' \in \mathscr{D}_{val}$ for which we wish to assess whether it is an anomaly. At each checkpoint t_i, randomly select a subsample of the training set of fixed size m, denoted $\mathrm{B}_t \subseteq \mathscr{D}_{train}$, and compute the average `TracInCP` influence. This gives the following anomaly score:

$$\texttt{TracInAD}(\mathbf{x}') = \frac{1}{m} \sum_{x \in \mathrm{B}_t} \sum_{i=1}^{k} \eta_i \nabla_\theta \ell(\theta_{t_i}, \mathbf{x}') \cdot \nabla_\theta \ell(\theta_{t_i}, \mathbf{x}) \tag{5}$$

The pseudo-code of the `TracInAD` algorithm is presented in Algorithm 1.

3.4.3 Application

To evaluate the pertinence of our methodology, we experiment using a vastly used deep anomaly detection method based on Variational Autoencoders (VAE) and the reconstruction error.

Variational Autoencoders

First proposed in Kingma and Welling, 2014, Variational Autoencoders (VAE) are a particular form of autoencoders that rely on a Bayesian framework. Assume that every sample $\mathbf{x} \in \mathscr{D}_{train}$ is obtained from some random process involving an unobserved continuous variable $\mathbf{z}$. The latter is sampled from some prior distribution $p_{\psi^*}(\mathbf{z})$, while $\mathbf{x}$ is obtained from a conditional distribution $p_{\psi^*}(\mathbf{x}|\mathbf{z})$. In the VAE framework, the core objective is two-fold: (i) estimating the latent variable $\mathbf{z}$ given a sample $\mathbf{x} \in \mathscr{X}$ and

(ii) estimating the parameter ψ^* which fully describes the conditional and prior distributions. To do so, a recognition model $q_\phi(\mathbf{z}|\mathbf{x})$, in the form of a neural network, is involved in the process to compensate for the intractability of the true posterior distribution $p_{\psi^*}(\mathbf{z}|\mathbf{x})$. Note that the recognition process $q_\phi(\mathbf{z}|\mathbf{x})$ and the conditional distribution $p_{\psi^*}(\mathbf{x}|\mathbf{z})$ are also referred to as, respectively, the encoder and the decoder.

One seeks to find the parameter ψ of the marginal distributions $p_\psi(\mathbf{x})$ such that the log-likelihood of the training set, $\sum_{\mathbf{x}\in\mathcal{D}_{train}} \log p_\psi(\mathbf{x})$, is maximized. Conjointly, the recognition model is estimated to be as close as possible to the true posterior distribution. This objective is met by minimizing the Kullback-Leibler divergence between the two distributions, $\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\psi(\mathbf{z}|\mathbf{x}))$. In the end, the VAE loss which is minimized w.r.t. $\{\psi,\phi\}$, can be expressed as

$$\begin{aligned}\mathscr{L}_{\mathrm{VAE}}(\psi,\phi) &= -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\log p_\psi(\mathbf{x}|\mathbf{z}) \\ &\quad +\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\psi(\mathbf{z}|\mathbf{x}))\end{aligned} \tag{6}$$

since $\log p_\psi(\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}\log p_\psi(\mathbf{x}|\mathbf{z})$. This loss can be minimized through backpropagation using the reparameterization trick (Kingma & Welling, 2014).

Given equation (6), the influence function described in equation (1) is expressed as follows:

$$\begin{aligned}\mathrm{IF}_{\mathrm{VAE}}(\mathbf{x},\mathbf{x}') &= -(\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}')}\log p_\psi(\mathbf{x}'|\mathbf{z}) - \mathbb{E}_{\mathbf{z}\sim q_{\phi_{-\mathbf{x}}}(\mathbf{z}|\mathbf{x}')}\log p_{\psi_{-\mathbf{x}}}(\mathbf{x}'|\mathbf{z})) \\ &\quad +((\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}')\|p_\psi(\mathbf{z}|\mathbf{x}')) - \mathrm{D}_{\mathrm{KL}}(q_{\phi_{-\mathbf{x}}}(\mathbf{z}|\mathbf{x}')\|p_{\psi_{-\mathbf{x}}}(\mathbf{z}|\mathbf{x}')))\end{aligned} \tag{7}$$

In the present case, we consider a Gaussian prior $p_\psi(\mathbf{z}) = \mathcal{N}(\mathbf{z};0,\mathrm{I})$. Similarly, let the conditional distribution $p_\psi(\mathbf{x}|\mathbf{z})$ be a multivariate Gaussian whose distribution parameters are computed from $\mathbf{z}$ using a neural network. The variational approximate posterior is set to be a multivariate Gaussian with a diagonal covariance matrix. Let a sample $\mathbf{x}\in\mathcal{D}_{train}$, then $\log q_\phi(\mathbf{z}|\mathbf{x}) = \log\mathcal{N}(\mathbf{z};\mu(\mathbf{x}),\sigma^2(\mathbf{x})\cdot\mathrm{I})$. Empirically, a sample $\mathbf{x}\in\mathcal{D}_{train}$ is fed to the encoder which outputs parameters $\{\mu(\mathbf{x}),\sigma(\mathbf{x})\}$. Those parameters then allow sampling a latent vector $\mathbf{z}$ corresponding to the original sample $\mathbf{x}$. Then, the latent vector is fed to the decoder, which outputs a reconstructed sample $\tilde{\mathbf{x}}$.

Reconstruction Error

A standard approach to anomaly detection that relies on Autoencoders and VAE consists in using exclusively normal samples in the training phase to estimate the normal distribution. In inference, the anomaly score is derived from the reconstruction error. Formally, consider a sample $\mathbf{x}'\in\mathcal{D}_{val}$, and a trained autoencoder f_{θ^*} whose parameters θ^* were obtained through an iterative process described in section 3.4.1. Denote by $\tilde{\mathbf{x}}' = f_{\theta^*}(\mathbf{x}')$ the reconstructed sample $\mathbf{x}'$ using the learned autoencoder. The anomaly score s is obtained as follows

$$s(\mathbf{x}') = \|\mathbf{x}'-\tilde{\mathbf{x}}'\|_2^2 \tag{8}$$

The described method relies on the same hypothesis as described in section 3.4.2.

`TracInAD` applied to VAE

Algorithm 1 can be applied in this setup: instead of relying on the reconstruction error $s(x')$ as defined in equation (8) to construct an anomaly score, we can use `TracInAD`. However, as seen in equation

(7), computing the exact influence function for a VAE can be challenging since it would involve computing an expectation over the encoder. However, Kong and Chaudhuri, 2021 have proven that under mild conditions, the empirical average of the influence function over l *i.i.d.* samples is close to the true influence function with high probability when l is properly selected. Therefore, empirically, we compute the average loss over l reconstructed samples for each sample considered, where l is a hyperparameter.

3.5 Experiments

3.5.1 Tabular Datasets

We propose to evaluate the model's performances presented in 3.4.3 concerning anomaly detection in a tabular dataset setup. We experiment on four benchmark datasets used in the literature. We follow the procedure first proposed in Zong et al., 2018.

Arrhythmia dataset It consists of a cardiology dataset from the UCI repo: it contains attributes related to the diagnosis of cardiac arrhythmia in patients. The dataset is comprised of 16 classes: class 1 are normal patients, and class 2 to 15 are arrhythmia-suffering patients. The smallest classes $(3, 4, 5, 7, 8, 9, 14, 15)$ are considered anomalous, while the rest normal. Note that categorical attributes are dropped: in total, there are 274 attributes. The train set comprises 193 samples, while the validation set contains 259 samples.

Thyroid Dataset It is another medical dataset from the UCI repo that contains attributes corresponding to whether a patient suffers from hyperthyroid. There are three classes in the dataset, among which *hyper-function* is designated as the anomalous class while the rest is normal. Only the 6 continuous attributes are used. The train set and validation set comprise $1,839$ and $1,933$ samples, respectively.

KDD and KDDRev Datasets KDD is an intrusion dataset that was created by an extensive simulation of a US Air Force LAN Network. The dataset consists of a normal data class and 4 simulated attack types (denial of service, unauthorized access from a remote machine, unauthorized access from a local superuser, and probing). 41 different attributes are considered: 34 are continuous, and 7 are categorical. Categorical features are encoded using one-hot encoding.

This dataset is used to construct two datasets. Firstly, the KDD dataset in which non-attack classes corresponding to 20% of the dataset, are treated as the anomaly class. Secondly, the KDD Reverse dataset (KDDRev), in which the attack class is subsampled to consist of 25% of the number of non-attack samples and is considered the anomaly class. The train set of KDDRev (resp. KDD) is composed of $48,639$ (resp. $198,371$) samples, while the validation contains $72,958$ (resp. $295,650$) samples.

Following the configuration of Zong et al., 2018, each training set is composed of 50% of the normal data. The 50% remaining and the entire anomaly samples compose the validation set. The share of anomalies in each validation set is the following: (i) Arrhythmia: 25.48% (ii) Thyroid: 4.8%, (iii) KDD: 32.9% (iv) KDDRev: 33.33%.

Table 3.1: Anomaly Detection Accuracy

Method	Dataset							
	Arrhythmia		Thyroid		KDD		KDDRev	
	F_1 Score	σ	F_1 Score	σ	F_1 Score	σ	F_1 Score	σ
OC-SVM	45.8		38.9		79.5		83.2	
E2E-AE	45.9		11.8		0.3		74.5	
LOF	50.0		52.7		83.8		81.6	
DAGMM	49.8		47.8		93.7		93.8	
GOAD	52.0	2.3	74.5	1.1	98.4	0.2	98.9	0.3
NeuTraL AD	60.3	1.1	76.8	1.9	99.3	0.1	99.1	0.1
Shenkar et al.	**61.8**	1.8	76.8	1.2	**99.4**	0.1	**99.2**	0.3
`TracInAD`	54.6	2.1	**77.6**	5.4	82.1	0.6	98.8	0.3

3.5.2 Hyperparameters

As discussed in section 3.4.3, we train a VAE for each dataset presented in the previous section exclusively on normal samples. For the smaller datasets, namely Arrhythmia and Thyroid, we consider an encoder and decoder comprised of 3 layers for the former dataset and 2 for the latter. Regarding the larger datasets, KDD and KDDRev, we resort to 6-layer networks for both the encoder and decoder. We consider a batch size of 32 for the Arrhythmia dataset and 16 for Thyroid; the models were trained for 20 epochs and 250 epochs, respectively. Note that the small size of both datasets involves a swift training process, allowing many epochs without significant computational overhead. Regarding the KDD and KDDRev datasets, models were trained for 20 epochs, with a batch size of 32 and 128, respectively. We used the SGD algorithm to optimize the parameters of the network along the training process with a learning rate set to 10^{-4} for both Thyroid and Arrhythmia datasets, while 10^{-5} for KDDRev and 10^{-7} for KDD. Due to the absence of validation set for early stopping, we considered stopping training when the training loss stopped improving for 10 consecutive epochs. We refer the reader to the code made available online for more detail on the hyperparameter set-up[1].

Regarding the computation of the anomaly score, we consider a step size between each saved checkpoint of 1 for Arrhythmia, 10 for Thyroid, and 2 for both KDD and KDDRev. In other words, we compute the `TracIn` measure of influence every 10 epochs, *i.e.* for 25 CP in the case of Thyroid. The size m of the batch containing random samples belonging to $\mathcal{D}_{train}$ used to compute `TracInAD`, as detailed in (5), is set to 128 for Arrhythmia, 64 for Thyroid datasets while 256 for KDD and KDDRev. We set l to 8 for the Thyroid dataset, 32 for Arrhythmia. Note that for faster training, we considered $l = 1$ for both larger datasets and did not observe any deterioration of the performances compared to larger values.

3.5.3 Results

The results of the experiments can be observed in Table 3.1. We compare our model to both shallow and deep anomaly detection methods. Note that results for models such as OCSVM, Local Outlier Factor (LOF), DAGMM (Zong et al., 2018), and GOAD (Bergman & Hoshen, 2020) were directly taken

[1] https://github.com/hugothimonier/TracInAD

from Bergman and Hoshen, 2020, while results for NeuTraL AD (Qiu et al., 2021) were taken from their paper. The F1-Score was chosen as the metric to compare the competing models per the literature.

For our model, we performed 10 iterations for the smaller datasets and 5 for the larger datasets. The rows of the table indicate the model used, while columns indicate the obtained mean metric for the F1-Score over the different iterations (the higher, the better) as well as its standard deviation in the columns denoted by σ for every dataset.

We observe that our model outperforms all state-of-the-art models on the Thyroid dataset. This may indicate that our model performs well on severe imbalanced setups: the Thyroid dataset stands out from the other three since it disposes of a much lower proportion of anomalies in the validation set. However, we also observe a relatively high standard deviation compared to other approaches. Our model competes well with (Shenkar & Wolf, 2022), NeuTraL AD (Qiu et al., 2021), and GOAD (Bergman & Hoshen, 2020) on the KDDRev dataset and outperforms all other considered approaches. Also, note that for the arrhythmia dataset, our method performs on par compared to other methods, while it fails to perform well on the KDD dataset.

3.6 Discussion

Generality of `TracInAD` As discussed in section 3.4.2, measuring influence as an anomaly score can be used for any deep anomaly detection method. For instance, one can easily apply `TracInAD` in the Deep SVDD (Ruff et al., 2018) framework, where $\ell(.,\theta)$ is taken to be the standard Deep-SVDD loss used to optimize the network's parameters. To support our statement regarding the generality of our approach, we experimented with the thyroid dataset in the Deep SVDD framework. We train a 3-layer encoder by minimizing the Deep SVDD loss. Note here that no pretraining was performed to initialize the weights of the network. The model is trained for 50 epochs with batch size 16. The step used to compute the `TracInAD` score is set to 10, as in the VAE setup. We compare the accuracy obtained with the standard Deep SVDD anomaly score, with the score computed as in (5) where $\ell(\theta, x)$ is the standard Deep-SVDD loss used to optimize the parameters of the network. We obtain a mean F1-Score of 50.1% for `TracInAD`-DSVDD over 200 iterations, in comparison with standard Deep-SVDD anomaly score, which obtains an F1-score of 40.1%. The difference is statistically significant. However, note that the results obtained suffer from high standard deviation, which may mitigate the strength of our statement.

High Standard Deviation Our model suffers from a higher standard deviation than competing models. We hypothesize that our model performances fluctuate more than other approaches mostly because they tend to exacerbate the original model's performance. On the one hand, when f_θ performs well, it tends to outperform the original model. On the other hand, when the original model performs poorly, `TracInAD` appears to perform worse than the original model.

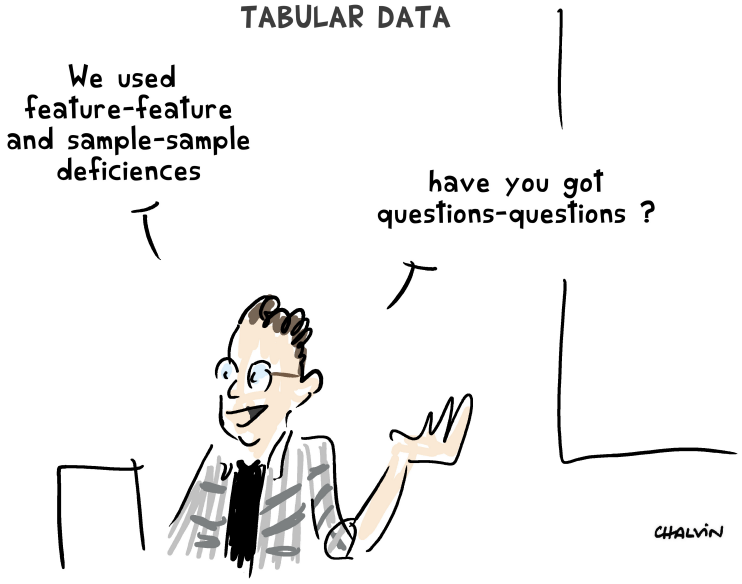
3.7 Conclusion

This chapter presents a novel methodology to detect anomalies based on `TracIn`, an influence measure originally proposed for explicability purposes. Through leveraging differences in influence rela-

tions between normal samples and between normal samples and anomalies, our method explicitly relies on sample-sample dependencies. This method offers competitive performance in comparison with existing methods in the literature. Moreover, our methodology can be adaptable to any deep anomaly detection and can improve on the standard anomaly score as discussed in section 3.6. Indeed, the `TracInAD` score can augment any feature-dependency-based method with little computational overhead for deep models with a reasonable number of parameters.

Chapter 4

Combining Feature-Feature and Sample-Sample Dependencies for Enhanced Anomaly Detection on Tabular Data



Abstract

Deep learning for tabular data has gained more and more attention in recent years. Deep models have shown strong performance on unstructured data for various tasks but remain challenging to use on structured data. Recent work has proposed using retrieval-augmented models for supervised tasks on tabular data, and it has shown encouraging performance on both classification and regression tasks. In this work, we investigate using retrieval-augmented models for anomaly detection on tabular data. Using a reconstruction-based approach, we employ a transformer model to reconstruct masked features from normal samples. We test the effectiveness of KNN-based modules and

attention mechanisms to select relevant samples to help reconstruct the sample of interest. Our experiments reveal that augmenting this reconstruction-based anomaly detection (AD) method with non-parametric relationships via these retrieval modules leads to a significant boost in performance on a large benchmark of 31 tabular datasets used in the literature.

This chapter is a modification of the following publications:

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024a). Beyond individual input for deep anomaly detection on tabular data. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, & F. Berkenkamp (Editors), *Proceedings of the 41st international conference on machine learning* (Pages 48097–48123). PMLR. (Cited on pages 78, 81, 82).

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024b). Retrieval augmented deep anomaly detection for tabular data. *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA*. https://doi.org/https://doi.org/10.1145/3627673.3679559

4.1 Introduction

Anomaly detection is a critical task that aims to identify samples that deviate from a pre-defined notion of normality within a dataset. Traditional approaches to anomaly detection characterize the *normal*[1] distribution almost exclusively using samples considered as *normal*, and flag data points as anomalies based on their deviation from this distribution. Anomaly detection (AD) benefits applications involving imbalanced datasets, where standard supervised methods may fail to achieve satisfactory performance (Yanmin et al., 2011). Those applications include fraud detection (Hilal et al., 2022), intrusion detection in cybersecurity (Malaiya et al., 2018), or astronomy (Reyes & Estévez, 2020).

Anomaly detection encompasses both unsupervised and supervised methods. In most real-world scenarios, labeled datasets that differentiate normal samples from anomalies are unavailable or costly to obtain. To address this, efficient anomaly detection methods must be robust to dataset contamination, where the training set is predominantly composed of normal samples but also includes anomalies. However, when labeled data is available, one can consider a supervised approach to create a training set consisting solely of *normal* samples, thereby indirectly incorporating label information into the anomaly detection model.

Many general AD methods tend to work well on tasks that involve unstructured data (*e.g.*, natural language processing or computer vision) such as (Kim et al., 2020; Liu, Ting, et al., 2008; Liznerski et al., 2021; Ruff et al., 2018; Schölkopf et al., 1999; Tax & Duin, 2004). However, recent work (Bergman & Hoshen, 2020; Qiu et al., 2021; Shenkar & Wolf, 2022) has revealed that the best-performing methods for tabular data involve models tailored to consider the particular structure of this data type. AD methods for structured data typically identify anomalies by using either *feature-feature* or *sample-sample* dependencies. For instance, Shenkar and Wolf, 2022 assume a class-dependent relationship between a subset of variables in a sample's feature vector and the rest of its variables. The authors thus propose a contrastive learning framework to detect anomalies based on this assumption. Another recent method (Thimonier et al., 2022) identifies anomalies in tabular datasets by focusing on sample-sample

[1]The term *normal* here relates to the concept of normality in opposition to *abnormal*.

dependencies by measuring the influence of training *normal* samples on the validation samples. Both approaches have demonstrated competitive results for anomaly detection in tabular datasets.

Recent work on supervised deep learning methods for tabular data (Arik & Pfister, 2021; Gorishniy et al., 2023; Kossen et al., 2021; Shavitt & Segal, 2018; Somepalli et al., 2021) has also highlighted the importance of considering the particular structure of tabular data. In particular, in (Gorishniy et al., 2023; Kossen et al., 2021; Somepalli et al., 2021) relax the traditional *i.i.d* assumption as the authors emphasize the significance of considering both feature-feature and sample-sample dependencies for supervised regression and classification problems on tabular data.

Based on the latter observation, we hypothesize that feature-feature relations and sample-sample dependencies are class-dependent, and **both dependencies should be used conjointly to identify anomalies**. In particular, since interactions between samples are learned exclusively using *normal* samples in the anomaly detection setup, they should be especially discriminative in identifying anomalies during inference. Han et al., 2022 discuss how anomalies on tabular data can be categorized into 4 types of anomalies. We believe leveraging both interactions is necessary to identify all 4 types efficiently. First, *dependency anomalies*, referring to samples that do not follow the dependency structure that normal data follow, require feature-feature dependencies to be efficiently identified. Second, *global anomalies* refer to unusual data points that deviate significantly from the norm. Relying on both dependencies should improve a model's capacity to detect these anomalies. Third, *local anomalies* that refer to the anomalies that are deviant from their local neighborhood can only be identified by relying on sample-sample dependencies. Finally, *clustered anomalies*, also known as group anomalies, are composed of anomalies that exhibit similar characteristics. This type of anomaly requires feature-feature dependencies to be identified.

To test this hypothesis, we employ transformers (Vaswani et al., 2017) in a mask-reconstruction framework to construct an anomaly score and test several approaches to leverage interactions between samples. We evaluate both **external retrieval methods** to augment transformers, namely KNN and attention mechanisms, and Non-Parametric Transformers (NPT) (Kossen et al., 2021) that **internally** incorporate sample-sample dependencies in their architecture. We test each approach on an extensive benchmark of tabular datasets to demonstrate the capacity of the models to detect anomalies and compare our performances to existing AD methods. We empirically show that the tested approaches incorporating retrieval modules to account for the sample-sample relations outperform the vanilla transformer that only attends to feature-feature dependencies. Moreover, the NPT-based method stands out from other methods by significantly outperforming competing retrieval-based methods and existing AD methods for tabular data. These findings align with recent work (Han et al., 2022) demonstrating the effectiveness of new deep learning architectures, such as transformers, for anomaly detection on tabular data. We also test the robustness of this approach to dataset contamination and give evidence that it can serve for unsupervised anomaly detection when the training set contamination is mild. Finally, our ablation study, conducted with a reconstruction-based approach similar to our proposed method but utilizing K-nearest neighbors (KNN) imputation, provides evidence that considering both dependencies can be crucial to detect anomalies on specific datasets accurately.

The present chapter offers the following contributions:

- We propose an extensive evaluation of retrieval-based methods for AD on tabular data.
- We show that sample-sample dependencies are key to identifying anomalies on certain datasets.

- Through our ablation study, we empirically show that sample-sample dependencies do not suffice to obtain consistent performance across datasets and should be combined with feature-feature dependencies.
- We show that the best-performing retrieval-based method displays state-of-the-art anomaly detection capacity on an extensive benchmark of 31 tabular datasets.

4.2 Illustrative Example

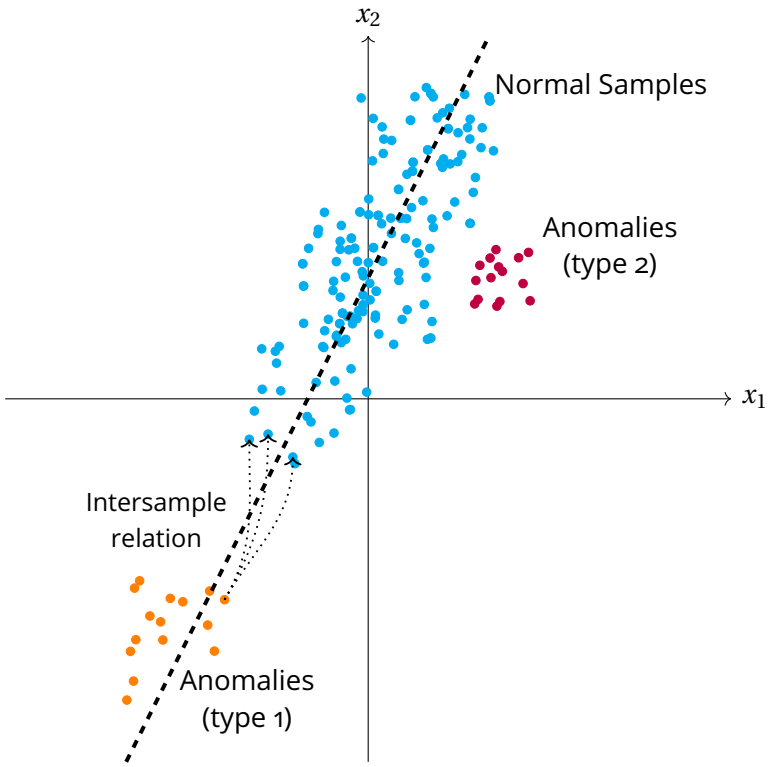


Figure 4.1: This graph illustrates the pertinence of combining both dependencies to identify anomalies correctly. Anomalies of type 1 (●) require intersample dependencies to be correctly identified with high probability. Indeed, the inter-feature relation is identical to normal samples and follows the relation detailed in eq. (1), hence leveraging inter-feature relation may not be sufficient to identify them properly. However, intersample relation, as shown in the figure, can be relevant as the distance to the closest normal samples is higher than for normal samples. Anomalies of type 2 (●) on the other hand require inter-feature dependencies to be correctly identified as the feature relation differs from the one displayed in (1) and only relying on inter-sample dependencies would be insufficient to correctly identify them as not belonging to the normal distribution as they are close to other normal samples.

Before discussing in depth the main concepts on which the methods proposed in this chapter rely, let us briefly discuss a situation in which combining both feature-feature and sample-sample dependencies is necessary to identify anomalies.

Consider a simple two dimensional data space, $\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$, in which the relation between the features of *normal* sample are entirely defined by a linear relation

$$x_2 = \alpha + \beta x_1 + \varepsilon, \tag{1}$$

where ε is some white noise.

Models only leveraging inter-feature relations may have limited capacities to identify anomalies if they satisfy the same relation as in eq. (1) but in a different subspace, e.g. anomalies of type 1 in Figure 4.1. For instance, a mask reconstruction model, as proposed in the present chapter, would likely correctly reconstruct features for both anomalies and normal samples, making identifying anomalies challenging. Similarly, a model that would only rely on inter-sample relations, e.g., KNN (Ramaswamy et al., 2000), which identifies anomalies based on a sample's average distance to its k-closest neighbors in the training set (higher score indicating anomalousness), would fail to identify anomalies of type 2 in Figure 4.1 correctly. This simple two-dimensional example illustrates the pertinence of combining both dependencies to ensure that all types of anomalies can be detected, as only leveraging one at a time would only permit identifying one of the two types displayed in Figure 4.1. We further explore this in section 4.5.1.

4.3 Related works

Anomaly detection approaches can be categorized into four main types: density estimation, one-class classification, reconstruction-based, and self-supervised.

Density Estimation The most straightforward approach to detecting samples that do not belong to a distribution is to estimate the distribution directly and to measure the likelihood of a sample under the estimated distribution. Several approaches found in the literature have considered using non-parametric density estimation methods to estimate the density of the *normal* distribution, such as KDE (Parzen, 1962), GMM (Roberts & Tarassenko, 1994b), or Copula as in COPOD (Li et al., 2020). Other approaches also focused on local density estimation to detect outliers, such as Local Outlier Factor (LOF) (Breunig et al., 2000). In inference, one flags the samples that lie in low-probability regions under the estimated distribution as anomalies.

Reconstruction Based Methods Other methods have consisted in learning to reconstruct samples that belong to the *normal* distribution. In this framework, the models' incapacity to reconstruct a sample correctly serves as a proxy to measure anomaly. A high reconstruction error would indicate that a sample does not belong to the estimated *normal* distribution. Those approaches can involve PCA (Hawkins, 1974) or neural networks such as diverse types of autoencoders (Chen & Konukoglu, 2018; Kim et al., 2020; Principi et al., 2017), or GANs (Schlegl et al., 2017).

One-Class Classification The term *one-class classification* was coined in Moya and Hush, 1996 and describes identifying anomalies without directly estimating the *normal* density. One-class classification (OCC) involves discriminative models that estimate a decision boundary directly. For instance, in kernel-based approaches (Schölkopf et al., 1999; Tax & Duin, 2004), authors propose to characterize the support of the *normal* samples in a Hilbert space and to flag as anomalies the samples that would lie outside of the estimated support. Similarly, recent work has extended their approach by replacing kernels with deep neural networks (Ruff et al., 2018). Goyal et al., 2020 propose DROCC that involves generating, in the course of training, synthetic anomalous samples in order to learn a classifier on

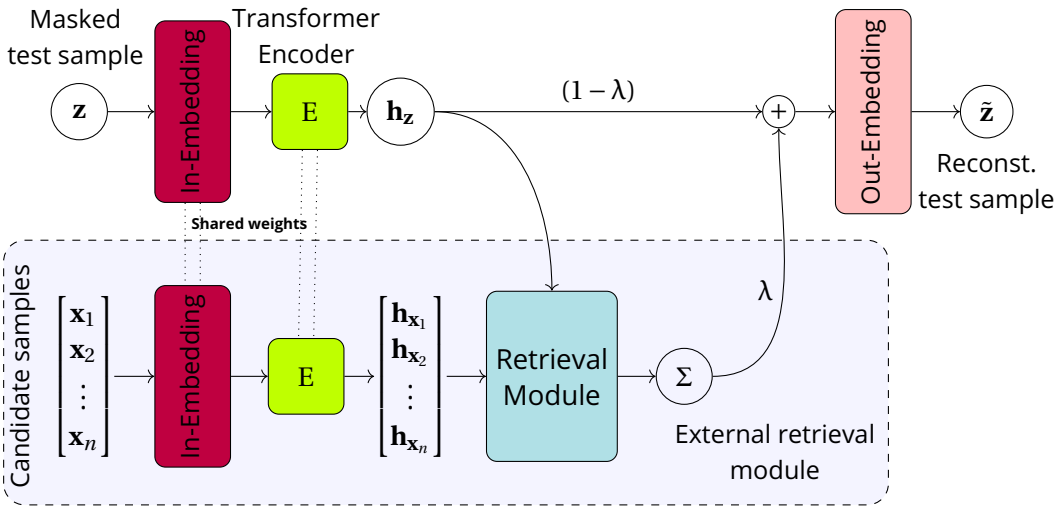


Figure 4.2: Forward pass for sample $\mathbf{z}$, see appendix 4.4.4 for more detail on training procedure. In the case of no retrieval module or an NPT as the transformer encoder, the prediction for a sample $\mathbf{z}$ consists of the upper part of the figure with $\lambda = 0$. Moreover, when considering NPTs as the transformer encoder, the model receives as input a matrix composed of the masked test sample and the candidate unmasked samples, $\mathbf{X} = [\mathbf{z}^o, \mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n]^\top$ and outputs $\tilde{\mathbf{X}} = [\tilde{\mathbf{z}}, \mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n]^\top$.

top of the one-class representation. Other OCC approaches have relied on tree-based models such as isolation forest (IForest) (Liu, Ting, et al., 2008), extended isolation forest (Hariri et al., 2021), RRCF (Guha et al., 2016) and PIDForest (Gopalan et al., 2019).

Self-Supervised Approaches Recent methods have also considered self-supervision as a means to identify anomalies. In GOAD (Bergman & Hoshen, 2020), authors apply several affine transformations to each sample and train a classifier to identify from the transformed samples which transformation was applied. The classifier only learns to discriminate between transformations using *normal* transformed samples: assuming this problem is class-dependent, the classifier should fail to identify transformation applied to anomalies. Qiu et al., 2021 propose NeuTraL-AD, a contrastive framework in which samples are transformed using neural mappings and are embedded in a latent semantic space using an encoder. The objective is to learn transformations so that transformed samples still share similarities with their untransformed counterpart while different transformations are easily distinguishable. The contrastive loss then serves as the anomaly score in inference. Similarly, Shenkar and Wolf, 2022 also propose a contrastive framework in which they identify samples as anomalies based on their inter-feature relations. Other self-supervised approaches (Reiss & Hoshen, 2023; Sohn et al., 2021) have focused on representation learning to foster the performance of one-class classification models.

Attention Mechanisms Popularized in Vaswani et al., 2017, the concept of attention has become ubiquitous in the machine learning literature. Scholars have successfully applied transformers on a broad range of tasks, including computer vision, *e.g.* image generation with the Image Transformer (Parmar et al., 2018) or image classification with the Vision Transformer (ViT) (Dosovitskiy et al., 2021), natural language processing *e.g.* Masked Language Models (MLM) such as BERT (Devlin et al., 2019), and classification tasks on structured datasets (Kossen et al., 2021; Somepalli et al., 2021).

Deep Learning for Tabular Data Despite the effectiveness of deep learning models for numerous tasks involving unstructured data, non-deep models remain the prevalent choice for machine learning tasks such as classification and regression on tabular data (Grinsztajn et al., 2022; Shwartz-Ziv & Armon, 2021). However, in recent years, scholars have shown that one could successfully resort to deep learning methods for various tasks on tabular datasets. For instance, in (Jeffares et al., 2023; Shavitt & Segal, 2018), authors discuss how regularization is crucial in training a deep learning model tailored for tabular data. Hence, they propose a new regularization loss to accommodate the variability between features. Similarly, Kadra et al., 2021 show that correctly selecting a combination of regularization techniques can suffice for a Multi-Layer Perceptron (MLP) to compete with GBDT. Kossen et al., 2021; Somepalli et al., 2021 propose deep learning models based on attention mechanisms that rely on feature-feature, feature-label, sample-sample, and sample-label attention. Both models achieve competitive results on several baseline datasets and emphasize sample-sample interaction's role in classifying samples correctly. Similarly, Gorishniy et al., 2023 explore the effectiveness of leveraging retrieval modules to augment MLP for classification and regression tasks on tabular data.

4.4 Method

4.4.1 Learning Objective

Let $\mathscr{D}_{train} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$ represent the training set composed of n *normal* samples with d features. The standard approach to anomaly detection involves learning some function $\phi_\theta : \mathbb{R}^d \to \mathscr{Z}$ by minimizing a loss function $\mathscr{L}$. The chosen loss function and $\mathscr{Z}$ will vary according to the class of the considered anomaly detection algorithm. Nevertheless, the overall aim of $\mathscr{L}$ is to characterize the distribution of the samples in the training set as precisely as possible. Depending on the chosen AD algorithm, the obtained representation $\phi_\theta(\mathbf{x})$ of sample $\mathbf{x}$ can be used directly or indirectly to compute an anomaly score.

Formally, the training objective can be summarized as follows

$$\min_{\theta \in \Theta} \sum_{\mathbf{x} \in \mathscr{D}_{train}} \mathscr{L}(\mathbf{x}, \phi_\theta(\mathbf{x})), \tag{2}$$

where $\mathscr{L}(\mathbf{x}, \phi_\theta(\mathbf{x}))$ will vary according to the chosen task. For a reconstruction-based method, $\mathscr{Z}$ can be the original data space and $\mathscr{L}$ can be the squared ℓ_2-norm of the difference between the original sample $\mathbf{x}$ and its reconstructed counterpart, $\mathscr{L}(\mathbf{x}, \phi_\theta(\mathbf{x})) = \|\mathbf{x} - \phi_\theta(\mathbf{x})\|^2$

Introducing an external retrieval module permits keeping the original objective unchanged while augmenting ϕ_θ with non-parametric mechanisms. The model involves non-parametric relations because it leverages the entire training dataset to form its prediction. Hence, the model can conjointly attend to feature-feature and sample-sample interactions to optimize its objective.

Formally, instead of minimizing the loss as described in equation (2), the parameters θ of function ϕ_θ are optimized to minimize the loss function as follows

$$\min_{\theta \in \Theta} \sum_{\mathbf{x} \in \mathscr{D}_{train}} \mathscr{L}\left(\mathbf{x}, \phi_\theta\left(\mathbf{x}; \mathscr{D}_{train}\right)\right). \tag{3}$$

Nevertheless, not all approaches to AD may benefit from such non-parametric mechanisms. Some pretext tasks involving sample-sample dependencies may lead to degenerate solutions, *e.g.* approaches

based on contrastive learning (Qiu et al., 2021; Shenkar & Wolf, 2022). However, reconstruction-based AD methods appear as a natural class of algorithms that may benefit from these non-parametric mechanisms.

Mask Reconstruction We empirically investigate the pertinence of external retrieval modules, as detailed in section 4.4.2, on a method involving stochastic masking and mask reconstruction for tabular data. Stochastic masking consists in masking each entry in a sample vector $\mathbf{x} \in \mathbb{R}^d$ with probability p_{mask}, and the objective task is to predict the masked-out features from the unmasked features. Formally, let $\mathbf{m} \in \mathbb{R}^d$ be a binary vector taking value 1 when the corresponding entry in $\mathbf{x}$ is masked, 0 otherwise. Let $\mathbf{x}^m, \mathbf{x}^o \in \mathbb{R}^d$ represent respectively the masked and unmasked entries of sample $\mathbf{x}$ defined as

$$\begin{aligned} \mathbf{x}^m &= \mathbf{m} \odot \mathbf{x} \\ \mathbf{x}^o &= (\mathbf{1}_d - \mathbf{m}) \odot \mathbf{x}, \end{aligned} \tag{4}$$

where $\mathbf{1}_d$ is the d-dimensional unit vector.

Formally, a model $\phi_\theta : \mathbb{R}^d \times \{0,1\}^d \to \mathbb{R}^d$ is trained to reconstruct the mask features $\mathbf{x}^m$ from its unmasked counterpart $\mathbf{x}^o$ and the mask vector $\mathbf{m}$. By construction, $\phi_\theta(\mathbf{x}^o; \mathbf{m})$ only has non-zero values for corresponding masked features in $\mathbf{m}$.

We evaluate the benefit of replacing the traditional reconstruction learning objective defined as

$$\min_{\theta \in \Theta} \sum_{\mathbf{x} \in \mathscr{D}_{train}} d(\mathbf{x}^m, \phi_\theta(\mathbf{x}^o; \mathbf{m})), \tag{5}$$

where $d(.,.)$ is a distance measure; with its equivalent augmented with a retrieval module as follows

$$\min_{\theta \in \Theta} \sum_{\mathbf{x} \in \mathscr{D}_{train}} d\left(\mathbf{x}^m, \phi_\theta\left(\mathbf{x}^o; \mathbf{m}, \mathscr{D}^o_{train}\right)\right), \tag{6}$$

where $\mathscr{D}^o_{train} = \{\mathbf{x}^o_i \in \mathbb{R}^d\}_{i=1}^n$. In inference, $\mathscr{D}^o_{train}$ is replaced by $\mathscr{D}_{train}$ in which none of the features of the training samples are masked.

4.4.2 Retrieval methods

Table 4.1: Retrieval modules characteristics

Retrieval type	Trainable end-to-end	Internal	Deterministic k
KNN	×	×	✓
External attention	✓	×	✓
NPT	✓	✓	×

Let $\mathbf{z}$ denote the sample of interest for which we wish to reconstruct its masked features $\mathbf{z}^m$ given its observed counterpart $\mathbf{z}^o$. Let $\mathscr{C}$ denote the candidate samples from the training set from which k *helpers* are to be retrieved, and $\mathscr{H}$ the retrieved *helpers*, $\mathscr{H} \subseteq \mathscr{C}$. We consider two types of retrieval modules whose main characteristics are displayed in table 4.1.

External modules

We consider several *external* retrieval modules that rely on similarity measures to identify relevant samples to augment the encoded representation of the sample of interest $\mathbf{z}$. It involves placing a retrieval module after the transformer encoder and before the output layer, as shown in figure 4.2. In section 4.7 we investigate the impact of modifying the location of the retrieval module and consider placing it before the encoder as an alternative.

For each method, we select the top-k elements that maximize a similarity measure $\mathscr{S}(\cdot,\cdot)$ and use a value function to obtain representations of the chosen samples $\mathscr{V}(\cdot,\cdot)$ to be aggregated with sample $\mathbf{z}$.

KNN-based First, we consider a simple method that identifies the k most relevant samples in $\mathscr{C}$ using a `KNN` approach. Formally, the similarity and value functions are defined as follows

$$\begin{aligned}\mathscr{S}(\mathbf{z},\mathbf{x}) &= -\|\mathbf{h}_\mathbf{z} - \mathbf{h}_\mathbf{x}\| \\ \mathscr{V}(\mathbf{z},\mathbf{x}) &= \mathbf{h}_\mathbf{x},\end{aligned} \tag{7}$$

where $\mathbf{h}_\mathbf{x}$ and $\mathbf{h}_\mathbf{z}$ denote the representations of respectively sample $\mathbf{x}$ and $\mathbf{z}$ and $\|.\|$ is the ℓ_2-norm.

Attention-based Second, we consider attention mechanisms to select $\mathscr{H}$. We consider three types of attention inspired by those proposed in Gorishniy et al., 2023. First, the vanilla attention (later referred to as `v-attention`), where the score and value function used to select the retrieved samples are defined as

$$\begin{aligned}\mathscr{S}(\mathbf{z},\mathbf{x}) &= \mathrm{W}_\mathrm{Q}(\mathbf{h}_\mathbf{z})^\top \mathrm{W}_\mathrm{K}(\mathbf{h}_x) \\ \mathscr{V}(\mathbf{z},\mathbf{x}) &= \mathrm{W}_\mathrm{V}(\mathbf{h}_\mathbf{x}).\end{aligned} \tag{8}$$

Second, we also consider a second type of attention module, `attention-bsim`, which involves replacing the score function defined in eq. (8) as follows

$$\begin{aligned}\mathscr{S}(\mathbf{z},\mathbf{x}) &= -\|\mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{z}) - \mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{x})\|^2 \\ \mathscr{V}(\mathbf{z},\mathbf{x}) &= \mathrm{W}_\mathrm{V}(\mathbf{h}_\mathbf{x}).\end{aligned} \tag{9}$$

Third, we consider `attention-bsim-bval` a modification of the value function in eq. (9) as

$$\begin{aligned}\mathscr{S}(\mathbf{z},\mathbf{x}) &= -\|\mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{z}) - \mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{x})\|^2 \\ \mathscr{V}(\mathbf{z},\mathbf{x}) &= \mathrm{T}(\mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{z}) - \mathrm{W}_\mathrm{K}(\mathbf{h}_\mathbf{x})),\end{aligned} \tag{10}$$

where $\mathrm{T}(\cdot) = \texttt{LinearWithtoutBias} \circ \texttt{Dropout} \circ \texttt{ReLU} \circ \texttt{Linear}(\cdot)$.

Internal attention modules

We consider Non-Parametric Transformers (NPTs) (Kossen et al., 2021) as an alternative retrieval-based method for tabular data. In NPTs, a sequence of MHSA layers is applied to the input, alternating between Attention Between Datapoints (ABD) and Attention Between Attributes (ABA). While ABA consists of a regular attention module capturing the relation between the features of a sample, ABD performs a retrieval-like task that identifies the relevant samples in a dataset to perform the

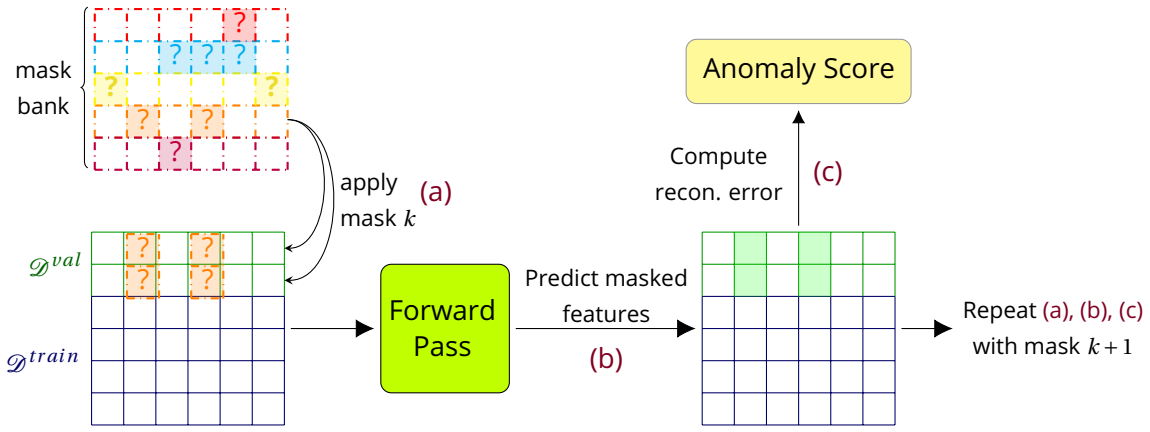


Figure 4.3: Transfomer inference pipeline. In step (a), mask j is applied to each validation sample. We construct a matrix $\mathbf{X}$ composed of the masked validation samples and the whole *unmasked* training set. If no retrieval module is involved, only the masked validation samples are included in $\mathbf{X}$. In step (b), we perform a forward pass on $\mathbf{X}$ as detailed in figure 4.2, to reconstruct the masked features for each validation sample. In step (c), we compute the reconstruction error that we aggregate in the anomaly score score.

chosen task. This mechanism captures pairwise relations between data samples. Consider as an input to the ABD layer the previous layer representation $\mathbf{H}^{(\ell)} \in \mathbb{R}^{n \times d \times e}$ flattened to $\mathbb{R}^{n \times h}$ where $h = d \cdot e$. Then, NPT applies MHSA, as seen in equation (29) in section 2.3.3, between the data samples' flattened representations $\{\mathbf{H}_i^{(\ell)} \in \mathbb{R}^{1 \times h} | i \in 1, \ldots, n\}$,

$$\text{ABD}(\mathbf{H}^{(\ell)}) = \text{MHSA}(\mathbf{H}^{(\ell)}) = \mathbf{H}^{(\ell+1)} \in \mathbb{R}^{n \times h}. \tag{11}$$

After applying ABD, the data representation is reshaped to its original dimension in $\mathbb{R}^{n \times d \times e}$. See section 2.3.3 for more details on non-parametric transformers. NPTs can be seen as a more complex version of the transformer `+v-attention` model in the sense that it is *approximately* equivalent to a sequence of ℓ 1-layer transformer encoder followed by a `v-attention` retrieval module, where ℓ would designate the number of ABA and ABD layers. Contrary to other considered modules, k is not a hyperparameter as it is handled by the model when it makes its prediction by setting attention weights to smaller values for irrelevant samples in $\mathscr{C}$. We later refer to this architecture setup as NPT-AD.

Aggregation While NPT does not require explicit aggregation as it performs it internally, the external retrieval modules necessitate aggregating the obtained retrieved representations $\mathcal{V}(\mathbf{z}, \mathbf{x})$ with the representation of the sample of interest $\mathbf{z}$. We aggregate the value of the selected top-k *helpers*, to be fed to the final layer

$$\tilde{\mathbf{h}}_{\mathbf{z}} = (1 - \lambda) \cdot \mathbf{h}_{\mathbf{z}} + \lambda \cdot \frac{1}{k} \sum_{x \in \mathscr{H}} \mathcal{V}(\mathbf{z}, \mathbf{x}). \tag{12}$$

where $\lambda \in [0, 1)$ is a hyperparameter.

4.4.3 Anomaly score

We directly derive the anomaly score from the loss optimized during training. The loss corresponds to the squared difference between the reconstructed feature and its actual value for numerical features. Meanwhile, for categorical features, we use the cross-entropy loss function. The anomaly score relies on our model's capacity to reconstruct masked features correctly and assumes that the model should better reconstruct *normal* samples. Two reasons support this assumption. First, relations between features are class-dependent; thus, as the model only learns from *normal* samples in the training phase, it should be unable to fetch the learned feature-feature interactions to reconstruct anomalies properly. Second, sample-sample interactions seen by the model only correspond to interactions between *normal* samples, making it difficult to successfully exploit interactions between *normal* samples and anomalies.

We consider a mask bank composed of m d-dimensional deterministic mask vectors that designate which of the d features of *each* validation sample will be hidden. We set the maximum number of features to be masked simultaneously r and construct $m = \sum_{k=1}^{r} \binom{d}{k}$ masks. Each mask is applied to each validation sample $\mathbf{z} \in \mathscr{D}^{val}$ to obtain m different masked samples $\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ of the original sample $\mathbf{z}$. We use the whole unmasked training set[2] $\mathscr{D}^{train}$ to predict the masked features of each sample for each of the m masked vectors and construct the anomaly score for a validation sample $\mathbf{z}$ as

$$\text{Anomaly-Score}(\mathbf{z}; \mathscr{D}^{train}) = \frac{1}{m} \sum_{k=1}^{m} \mathscr{L}(\mathbf{z}^{(k)}; \mathscr{D}^{train}), \tag{13}$$

where $\mathscr{L}(\mathbf{z}^{(k)}; \mathscr{D}^{train})$ designates the loss for the sample $\mathbf{z}$ with mask k. We also considered other forms of aggregation, such as the maximum loss overall masks.

4.4.4 Training pipeline

Let $\mathbf{x} \in \mathscr{X} \subseteq \mathbb{R}^d$ be a sample with d features, which can be either numerical or categorical. Let e designate the hidden dimension of the transformer. The training pipeline consists of the following steps:

Masking We sample from a Bernouilli distribution with probability p_{mask} whether each of the d features is masked.

$$\text{mask} = (m_1, \ldots, m_d),$$

where $m_j \sim \mathscr{B}(1, p_{mask})$ $\forall j \in [1, \ldots, d]$ and $m_j = 1$ if feature j is masked.

Encoding For numerical features, we normalize to obtain 0 mean and unit variance, while we use one-hot encoding for categorical features. At this point, each feature j for $j \in [1, 2, \ldots, d]$ has an e_j-dimensional representation, $encoded(\mathbf{x}_j) \in \mathbb{R}^{e_j}$, where $e_j = 1$ for numerical features and for categorical features e_j corresponds to its cardinality. We then mask each feature according to the sampled mask vector and concatenate each feature representation with the corresponding mask indicator function. Hence, each feature j has an $(e_j + 1)$-dimensional representation

$$((1 - m_j) \cdot encoded(\mathbf{x}_j), m_j) \in \mathbb{R}^{e_j + 1},$$

[2] For large datasets, we resort to a random subsample of the training set for computational reasons.

where $\mathbf{x}_j$ is the j-th features of sample $\mathbf{x}$.

In-Embedding We pass each of the features encoded representations of sample $\mathbf{x}$ through learned linear layers $\texttt{Linear}(e_j + 1, e)$. We also learn e-dimensional index and feature-type embeddings as proposed in Kossen et al., 2021. Both are added to the embedded representation of sample $\mathbf{x}$. The obtained embedded representation is thus of dimension $d \times e$

$$e_{\mathbf{x}} = (e^1_{\mathbf{x}}, e^2_{\mathbf{x}}, \ldots, e^d_{\mathbf{x}}) \in \mathbb{R}^{d \times e},$$

and $e^j_{\mathbf{x}} \in \mathbb{R}^e$ corresponds to the embedded representation of feature j of sample $\mathbf{x}$.

Transformer Encoder The embedded representation obtained from the previous step is then passed through a transformer encoder. The output of the transformer $\mathbf{h}_{\mathbf{x}}$ is of dimension $d \times e$.

Out-Embedding The output of the transformer, $\mathbf{h}_{\mathbf{x}} \in \mathbb{R}^{d \times e}$ is then used to compute an estimation of the original d-dimensional vector. To obtain the estimated feature j, we take the j-th d dimensional representation which is output by the transformer encoder, $e_{\mathbf{x}_j} \in \mathbb{R}^d$, and pass it through a linear layer $\texttt{Linear}(e, e_j)$, where e_j is 1 for numerical features and the cardinality for categorical features.

External Retrieval Modules During training, for a batch $\mathscr{B}$ composed of b samples, for each sample $\mathbf{x} \in \mathscr{B}$, the entire batch serves as the candidates $\mathscr{C}$. In inference, a random subsample of the training set is used as $\mathscr{C}$. Both for training and inference, when possible memory-wise, we use as $\mathscr{B}$ and $\mathscr{C}$ the entire training set.
As input, the retrieval module receives a $\mathbb{R}^{d \times e}$ representation for each sample. Operations described in eq. (7), (8), (9) and (10) are performed on the flattened representations of samples $\mathbf{x}$, $\mathbf{h}^{\mathbf{x}}_{flatten} \in \mathbb{R}^{d \cdot e}$. After selecting $\mathscr{H} \subseteq \mathscr{C}$, each sample is transformed back to its original dimension to allow aggregation as described in equation (12).

The training pipeline of NPT-AD slightly differs from the one we have just described. The main difference originates from the different types of input expected by the transformer and the non-parametric transformer. While the transformer will treat each sample independently, the non-parametric transformer expects a matrix $\mathbf{X} \in \mathbb{R}^{n \times d \times e}$ as input, where each of the n rows corresponds to a sample and each of the d columns to a feature's embedded e-dimensional representation. So for a batch $\mathscr{B}$, where $\mathscr{B} = \mathscr{C}$ by construction, NPT receives direclty $\mathscr{B}$ as input, where $\mathscr{B} \in \mathbb{R}^{b \times d \times e}$ and b is the batch size.

4.5 Experiments

4.5.1 Why combine dependencies?

We further explore the illustrative example discussed in section 4.2 in which we hypothesized that *different types of anomalies require different dependencies to identify them*. In this section, we provide a simple example to test this statement. Consider a simple three dimensional data space, $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^2$,

in which the relation between the features of *normal* sample are defined as follows

$$\begin{aligned} x_2 = &\quad \alpha_1 + \beta_1 x_1 + \varepsilon \\ x_3 = &\quad \alpha_2 + \beta_2 x_2^2 + \varepsilon \end{aligned} \tag{14}$$

where ε is some white noise and $(\alpha_1, \alpha_2, \beta_1, \beta_2) \in \mathbb{R}^2$ are scalars, which is very similar to the illustrative example discussed in section 4.2.

Table 4.2: Share (%) of each class correctly identified (↑). Average over 5 data splits. The table should be read as follows: On average, the transformer correctly classified 78.5% of anomalies of type 1 as anomalies.

	Normal	Anomalies (type 1)	Anomalies (type 2)
Transformer	91.4% (±1.4)	78.5%(±1.0)	**100.0**%(±0.0)
`+att-bsim`	94.6% (±0.5)	88.0%(±0.8)	**100.0**%(±0.0)
Mask-KNN	93.0% (±0.4)	**100.0**%(±0.0)	77.3%(±4.4)
NPT-AD	**98.4**% (±0.2)	**100.0**%(±0.0)	96.0%(±0.6)

Let us consider two types of anomalies as shown in figure 4.4. First anomalies of type 1, in which the relations between features are identical to those given in eq (1) but in a different subspace. Now consider type 2 anomalies, for which the values of the generating feature x_1 are in the same subspace as *normal* samples, the relation between x_1 and x_2 is the same, but the parameters of the relation between x_2 and x_3 differs.

To test our hypothesis, we introduce a reconstruction-based technique similar to NPT-AD, Mask-KNN, that relies on KNN imputation to reconstruct masked features (see alg. 2 in section 4.6.2). Our experiment also included the vanilla transformer model trained in a framework similar to NPT-AD. As later discussed in section 4.6.2, Mask-KNN (resp. the transformer) can be seen as approximately equivalent to NPT-AD without considering the feature-feature dependencies (resp. the sample-sample dependencies) as illustrated in figure 4.7 in appendix 4.6.2.

In the present framework, models only leveraging inter-feature relations, such as the vanilla transformer, may have limited capacities to identify anomalies if they satisfy the same relations as given in eq. (14) but in a different subspace, i.e., anomalies of type 1. For instance, a mask reconstruction model relying on inter-feature relation, such as the vanilla transformer, would correctly reconstruct features for anomalies and normal samples, making identifying anomalies challenging. Similarly, a model that would only rely on inter-sample relations, e.g., Mask-KNN, which reconstructs the masked feature anomalies based on the distance to its k-closest neighbors in the training set, would struggle to correctly identify anomalies of type 2 as they would lie in a subspace close to normal samples.

Synthetic Dataset We effectively test this hypothesis by constructing a synthetic three-dimensional dataset where the features of normal samples satisfy the relations described in eq. (14). We also construct two types of anomalies where type 1 anomalies follow the same inter-feature relation as normal samples but with values in a non-overlapping interval as those of the normal class. Type 2 anomalies are constructed to be close to the normal population but with inter-feature relation differing from eq. (14). The normal population comprises 1000 samples, and we generate 200 anomalies for each type. We use half of the normal samples to train models and use the rest merged with the anomalies as

the validation set. We compare the capacity of Mask-KNN, the vanilla transformer, the transformer augmented with the `attention-bsim` retrieval module, and the NPT to identify anomalies and display the obtained results in table 4.2. We consider the same mask bank, $\{(1,0,0),(0,1,0),(0,0,1)\}$, for all three approaches and train both transformers until no loss improvement for 10 consecutive epochs. We use the same strategy for selecting the decision threshold as detailed in section 4.5. The synthetic three-dimensional dataset was generated as follows.

- **Normal samples**: We consider an interval of values $[-2,3]$ from which we uniformly sample the first feature x_1. We then sample x_2, x_3 following the relation given in equation (14) with parameters, $(\alpha_1,\beta_1) = (2,3)$, $(\alpha_2,\beta_2) = (4,3)$ and $\varepsilon \sim \mathcal{N}(0,1)$.
- **Anomalies (type 1)** (●): We consider an interval of values $[3.3,4]$ from which we uniformly sample the first feature x_1 and keep the rest as for normal samples.
- **Anomalies (type 2)** (●): We consider an interval of values $[1.5,2.5]$ from which we uniformly sample the first feature x_1 and sample x_2, x_3 following equation (14) but with parameters $(\alpha_1,\beta_1) = (-7.5,-1)$ and $(\alpha_2,\beta_2) = (4,3)$.

NPT-AD and the transformer were trained with the same hyperparameter as in section 4.6.1. For Mask-KNN, we considered $k = 5$.

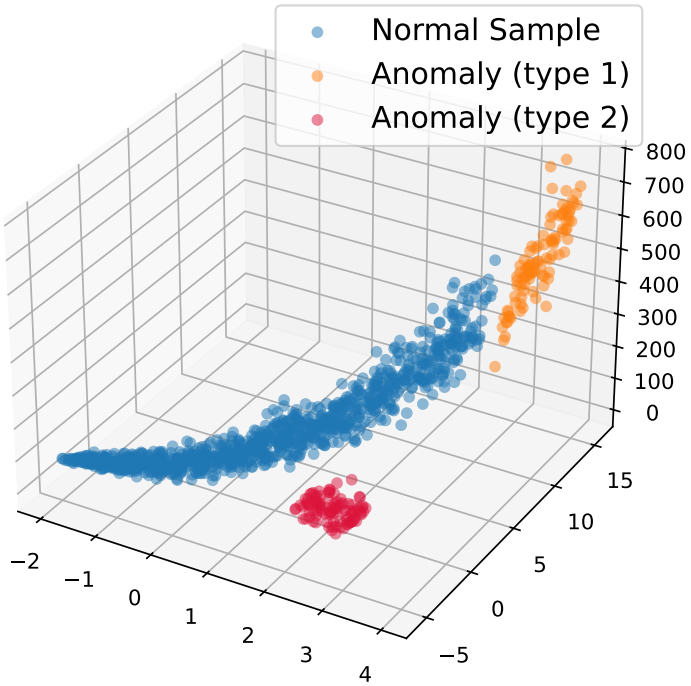


Figure 4.4: Representation of the synthetic dataset used for the experiment. Anomalies of type 1 (●) require inter-sample dependencies to be correctly identified with high probability. Anomalies of type 2 (●), on the other hand, require inter-feature dependencies to be correctly identified.

Results We observe in table 4.2 that the vanilla transformer struggles to detect type 1 anomalies compared other methods, as they explicitly require inter-sample dependencies to be identified. Nevertheless, it correctly identifies 91.4% of normal samples on average and perfectly detects type 2

anomalies. Conversely, Mask-KNN obtains significantly lower performance than competing methods for type 2 anomalies while correctly identifying type 1 anomalies and normal samples. Notice how both methods cannot correctly identify anomalies from one of the two anomaly types despite using a perfectly separable dataset. On the contrary, we observe that NPT-AD and the retrieval-augmented transformer can identify both types of anomalies as they can leverage inter-sample and inter-feature dependencies. Moreover, we observe that NPT-AD outperforms the augmented transformer in the detection of type 1 anomalies and normal samples. Overall, this experiment provides information as to why NPT-AD outperforms existing methods (see section 4.5.4): it can detect most anomaly types, while other approaches are confined to some anomaly categories. While NPT-AD may not outperform all methods on all datasets, this finding accounts for the regularity of NPT-AD across datasets.

4.5.2 Datasets

We experiment on an extensive benchmark of tabular datasets following previous work (Shenkar & Wolf, 2022). The benchmark is comprised of two datasets widely used in the anomaly detection literature, namely Arrhythmia and Thyroid, a second group of datasets, the "Multi-dimensional point datasets", obtained from the Outlier Detection DataSets (ODDS)[3] containing 28 datasets. We omit Heart and Yeast datasets following the literature and the KDD dataset since it presents several limitations (Silva et al., 2020). Instead, we include three real-world datasets from Han et al., 2022 that display relatively similar characteristics to KDD in terms of dimensions: fraud, campaign, and backdoor. See table 4.3 for more detail on the datasets' characteristics.

4.5.3 Experimental Settings

Per the literature (Bergman & Hoshen, 2020; Zong et al., 2018), we construct the training set with a random subsample of the *normal* samples representing 50% of the *normal* samples, we concatenate the 50% remaining with the entire set of anomalies to constitute the validation set. Following previous work (Bergman & Hoshen, 2020), the decision threshold for the AD score is chosen such that the number of predicted anomalies is equal to the number of existing anomalies.

To evaluate to which extent sample-sample dependencies are relevant for anomaly detection, we compare models that attend to relations between samples to the vanilla transformer model. We compare the different methods discussed in section 4.4.2 to the vanilla transformer using both the F1-score (↑) and AUROC (↑) metrics following the literature. For each dataset, we report an average over 20 runs for 20 different seeds in table 4.4 the average F1 and AUROC. We refer the reader to appendices B.4 and B in which we display the detailed results for all six transformer-based methods in tables B.10 and B.2 for the F1-score and B.11 and B.4.

We considered three regimes for the transformer dimensions depending on the dataset size. The transformer encoder comprises 4 layers with 4 attention heads and hidden dimension $e \in [8, 16, 32]$ for smaller to larger datasets. For all datasets, we train the transformer with a mask probability $p_{mask} = 0.15$ and rely on the LAMB optimizer (You et al., 2020) with $\beta = (0.9, 0.999)$ and also included a Lookahead (Zhang et al., 2019) wrapper with slow update rate $\alpha = 0.5$ and $k = 6$ steps between updates. We also include a dropout regularization with $p = 0.1$ for attention weights and hidden layers. We ensure that during training, all samples from a batch are not masked simultaneously so that the retrieval

[3] http://odds.cs.stonybrook.edu/

Table 4.3: Datasets characteristics

Dataset	n	d	Outliers
Wine	129	13	10 (7.7%)
Lympho	148	18	6 (4.1%)
Glass	214	9	9 (4.2%)
Vertebral	240	6	30 (12.5%)
WBC	278	30	21 (5.6%)
Ecoli	336	7	9 (2.6%)
Ionosphere	351	33	126 (36%)
Arrhythmia	452	274	66 (15%)
BreastW	683	9	239 (35%)
Pima	768	8	268 (35%)
Vowels	1456	12	50 (3.4%)
Letter Recognition	1600	32	100 (6.25%)
Cardio	1831	21	176 (9.6%)
Seismic	2584	11	170 (6.5%)
Musk	3062	166	97 (3.2%)
Speech	3686	400	61 (1.65%)
Thyroid	3772	6	93 (2.5%)
Abalone	4177	9	29 (0.69%)
Optdigits	5216	64	150 (3%)
Satimage-2	5803	36	71 (1.2%)
Satellite	6435	36	2036 (32%)
Pendigits	6870	16	156 (2.27%)
Annthyroid	7200	6	534 (7.42%)
Mnist	7603	100	700 (9.2%)
Mammography	11183	6	260 (2.32%)
Shuttle	49097	9	3511 (7%)
Mulcross	262144	4	26214 (10%)
ForestCover	286048	10	2747 (0.9%)
Campaign	41188	62	4640 (11.3%)
Fraud	284807	29	492 (0.17%)
Backdoor	95329	196	2329 (2.44%)

module receives encoded representations of unmasked samples as it will in the inference stage. We considered the same transformer architecture for the same dataset and identical hyperparameters for situations including no retrieval or an external attention module. For external retrieval modules, we chose for simplicity $\lambda = 0.5$ for aggregation as detailed in eq. (12). We study in the appendix the effect of varying the value of λ on the model's performance. For the KNN module, we set $k = 5$ as the cardinality of $\mathscr{H}$. In contrast, for the attention modules, we set $\mathscr{H} = \mathscr{C}$ and use the attention weights to compute a weighted mean to be aggregated as in equation (12). We further discuss the choice of k in appendix 4.7.1.

Regarding the hyperparameters of NPTs, we considered the same NPT architecture for all datasets composed of 4 layers alternating between Attention Between Datapoints and Attention Between Attributes and 4 attention heads. Per Kossen et al., 2021, we consider a row-wise feed-forward (rFF) network with one hidden layer, 4x expansion factor, GeLU activation, and also include dropout with $p = 0.1$ for both attention weights and hidden layers. We use the same optimizer as for the vanilla

transformers with identical hyperparameters as described above. We extensively detail in appendix B.1 the varying hyperparameters and experimental settings of our experiments. Overall, hyperparameters were selected to obtain training loss convergence with the following pipeline:

- Hidden dimension: We started from smaller to larger models to achieve convergence, starting with hidden dimension 8 and increasing to 64. Each step would consist in multiplying the previous hidden dimension by 2.
- Batch size: We maximized batch size to fit into memory. Larger batch sizes benefit our approach since a larger number of samples in the same batch increases the samples to which each sample can attend and fosters better learning of sample-sample dependencies.
- Learning rate was selected to achieve the fastest loss convergence for each architecture.
- Masking probability: we started from 0.25 and reduced with a 0.05 step until loss convergence.

All models were trained on $[1, 2, 4, 8]$ Nvidia GPUs V100 16Go/32Go, depending on the dataset dimension, thanks to HPC resources of IDRIS under the allocation 2023-101424 made by GENCI. We observed that vanilla transformers, even when augmented with a retrieval module, were easier to train than NPTs and did not necessitate too many iterations to obtain loss convergence. Each experiment can be replicated using the code made available on the online repositories.

4.5.4 Results

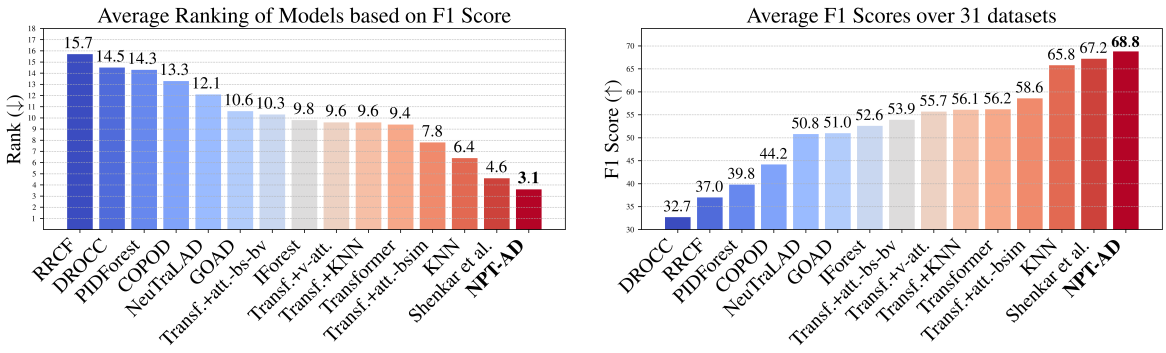


Figure 4.5: For each of the 31 datasets on which models were evaluated, we report the average F1-score over 20 runs for 20 different seeds. For both figures, the model displayed on the far left is the worst-performing model for the chosen metric, and on the far right is the best-performing model. We also highlight the metric of the best-performing model in bold. We refer the reader to tables B.2, B.3, and B.10 for details on the obtained metrics for each dataset.

As reported in table 4.4, we observe that not all retrieval modules significantly boost anomaly detection performance. First, regarding external retrieval modules, we observe that only the transformer augmented by the `attention-bsim` and the KNN modules perform significantly better than the vanilla transformer. Second, the NPT-based approach outperforms all other methods by a sizable margin for both F1-score and AUROC. Two factors could account for such results. First, NPT appears more flexible than external modules in attending to other samples. Indeed, it attends more regularly to other samples to learn the encoded representations as its architecture alternates between ABD and ABA layers. On the contrary, external retrieval modules are only attended to once and for

Table 4.4: Comparison of transformer-based methods.

Metric	Transformer	+KNN	+v-att.	+att-bsim	+att-bsim-bval	NPT-AD
F1-Score	56.2	56.1	55.7	58.6	53.9	**68.8**
AUROC	83.4	83.1	83.1	84.4	82.1	**88.9**

all before the output layer. Secondly, external retrieval modules necessitate an explicit aggregation function to use samples' representation, while NPT relies on MHSA to incorporate other samples' representations. Moreover, external modules imply explicitly setting the number of samples to attend to and the weight to put on other samples λ. We further discuss in the section 4.7 the effect of varying λ and k on the performance of the retrieval modules.

4.6 NPT-AD Ablation Study

4.6.1 Training set contamination

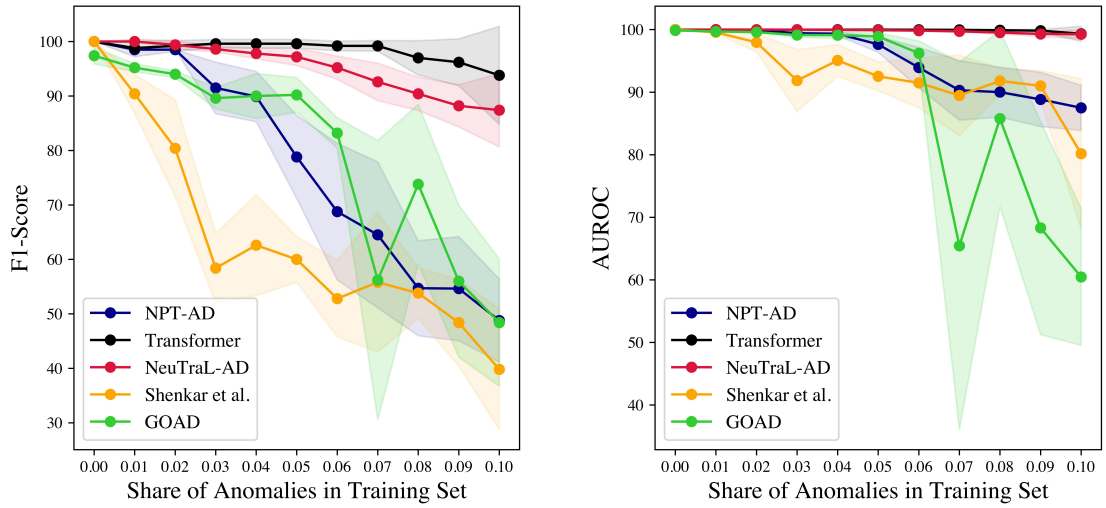


Figure 4.6: Training set contamination impact on the F1-score and AUROC. Each model was trained 5 times for each contamination share. The architecture used for NPT-AD is the same as for all experiments (see section 4.5). The NPT and Transformer were trained for 100 epochs with batch size equal to the dataset size, with learning rate 0.01, optimizer LAMB (You et al., 2020) with $\beta = (0.9, 0.999)$, per-feature embedding dimension 16, r set to 1, and masking probability $p_{mask} = 0.15$. NeuTraL-AD and GOAD were trained with hyperparameters as for the thyroid dataset in the original papers and (Shenkar & Wolf, 2022) with its default parameters in their implementation.

Real-life anomaly detection applications often involve contaminated training sets; anomaly detection models must, therefore, be robust to small levels of dataset contamination. We experimented using a synthetic dataset to evaluate how much NPT-AD suffers from dataset contamination compared to recent deep AD methods. We constructed a synthetic dataset using two perfectly separable distributions for *normal* and anomaly samples. Our training set contained 900 *normal* samples, and we kept aside 100 anomaly samples that we could add to the training set. We considered 11 different

training sets with contamination shares ranging from 0% to 10% with a 1% step while keeping the validation set constant with a fixed composition of 10% anomalies and 90% *normal* samples. We display the results of this experiment in Figure 4.6 in which we show how the performance of NPT-AD varies when the contamination share increases in comparison with a vanilla transformer (Vaswani et al., 2017) for mask reconstruction, NeuTraL-AD (Qiu et al., 2021), GOAD (Bergman & Hoshen, 2020) and the internal contrastive approach of Shenkar and Wolf, 2022.

Our experimental results show that, as expected, the performance of NPT-AD deteriorates as the proportion of anomalies in the training set increases. For contamination shares lower than 2% (resp. 5%), the F1-score (resp. AUROC) remains close to its maximum value of 100%. However, the F1-score and AUROC deteriorate significantly for higher contamination levels while displaying a higher standard deviation. When anomalies constitute 10% of the training set, our approach achieves an average F1-score slightly lower than 50% and an average AUROC of 87%. We observe that NPT-AD suffers less from dataset contamination than the method of Shenkar and Wolf, 2022 for both F1-score and AUROC. We also notice that the method of Shenkar and Wolf, 2022 is particularly sensible to dataset contamination regarding the F1-score compared with NeuTraL-AD, GOAD, the transformer approach, and NPT-AD even for low contamination shares.

Finally, we observe that the transformer approach suffers significantly less from training set contamination than NPT-AD, especially for high contamination shares. What could account for such a difference is the fact that the contamination effect is double for NPT-AD. First, during training, the model acquires the ability to reconstruct normal samples and anomalies, thereby compromising the reconstruction error as a good proxy for anomalousness. Second, in inference, the model relies on the unmasked training set to reconstruct samples: normal samples may attend to anomaly samples, which may hamper the ability of the model to reconstruct them properly. Conversely, anomalies can attend to other anomalies, which helps reduce the reconstruction error for this class.

4.6.2 Sample-sample dependencies ablation study

Table 4.5: Ablation study. Comparison in AD performance between the vanilla transformer, Mask-KNN, and NPT-AD.

	Transformer	Mask-KNN	NPT-AD
F1	56.2	57.5	**68.8**
AUROC	83.4	84.5	**89.8**

To further explore the combined impact of sample-sample and feature-feature dependencies, we introduce a reconstruction-based technique similar to NPT-AD. We put forward Mask-KNN that relies on KNN imputation to reconstruct masked features (see alg. 2). We also compare it to the vanilla transformer model trained in a framework similar to NPT-AD. Mask-KNN (resp. the transformer) can be seen as approximately equivalent to NPT-AD without considering the feature-feature dependencies (resp. the sample-sample dependencies) as illustrated in figure 4.7. Our experiment is summarized in tables 4.5, 4.6 and 4.7. We observe that, indeed, NPT-AD outperforms the vanilla transformer and Mask-KNN based on both AUROC and F1-score as shown in table 4.5, emphasizing that combining both types of dependencies boosts anomaly detection performance.

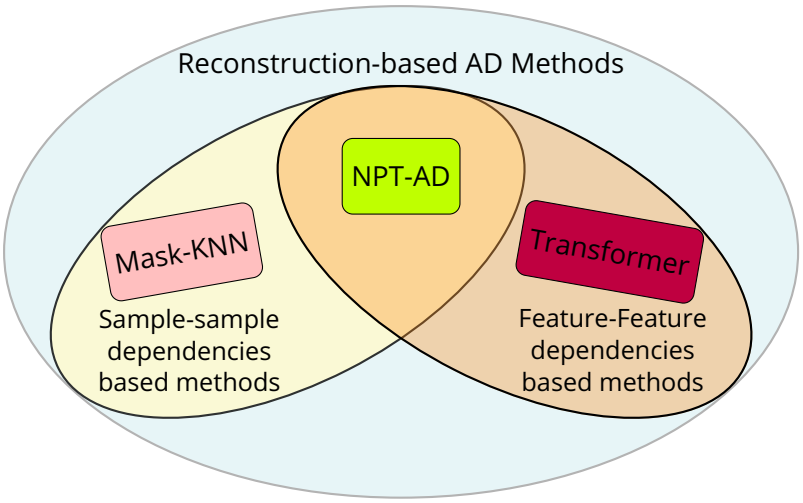


Figure 4.7: While Mask-KNN only relies on sample-sample dependencies and the vanilla transformer attends to feature-feature dependencies, NPT-AD combines both for anomaly detection.

K-Nearest Neighbor Imputation Take a dataset $\mathscr{D} = \{\mathbf{x}_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^d$ and for which some samples might display missing values in the feature vector. K-nearest neighbor imputation for a sample $\mathbf{z} \in \mathscr{D}$ consists in identifying the k nearest neighbors of sample $\mathbf{z}$ given a distance measure $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, where k is a hyperparameter that must be discretionary chosen. This distance measure only considers the non-missing features of sample $\mathbf{z}$. Let $\mathscr{I}$ designate the index of the non-missing features and $\mathbf{z}^{[\mathscr{I}]}$ the corresponding features of sample $\mathbf{z}$, then the k-nearest neighbors of sample $\mathbf{z}$ are identified through evaluating the distance $d(\mathbf{z}^{[\mathscr{I}]}, \mathbf{x}_j^{[\mathscr{I}]})$ for each $\mathbf{x}_j \in \mathscr{D}$ and ordering them to find the k smallest. Let $\mathscr{K}(z)$ designate the k nearest neighbors of sample $\mathbf{z}$, $\bar{\mathscr{I}}$ the missing values of $\mathbf{z}$, then $\forall i \in \bar{\mathscr{I}}$

$$\hat{\mathbf{z}}^i = \frac{1}{k} \sum_{\mathbf{x} \in \mathscr{K}(\mathbf{z})} \mathbf{x}^i. \tag{15}$$

Other imputation methods include weighting each sample in $\mathscr{K}(\mathbf{z})$ by its inverse distance to $\mathbf{z}$, denoted $\omega_{(\mathbf{z},\mathbf{x})}^{[\mathscr{I}]} = 1/d(\mathbf{z}^{[\mathrm{I}]}, \mathbf{x}_j^{[\mathscr{I}]})$. This gives

$$\hat{\mathbf{z}}^i = \frac{1}{\sum_{\mathbf{x}} \omega_{(\mathbf{z},\mathbf{x})}^{[\mathscr{I}]}} \sum_{\mathbf{x} \in \mathscr{K}(\mathbf{z})} \omega_{(\mathbf{z},\mathbf{x})}^{[\mathscr{I}]} \mathbf{x}^i. \tag{16}$$

This approach leverages only sample-sample dependencies, while the vanilla transformer only leverages feature-feature dependencies, and NPT-AD combines these two types as illustrated in figure 4.7.

Mask-KNN Anomaly Score Consider a training set $\mathscr{D}_{train} = \{\mathbf{x}_i\}_{i=1}^{n_{train}}$, $\mathbf{x}_i \in \mathbb{R}^d$ comprised of only *normal* samples and a validation set $\mathscr{D}_{val} = \{\mathbf{x}_i\}_{i=1}^{n_{val}}$ for which we wish to predict the label. In a reconstruction-based approach, we construct an anomaly score based on how masked samples are well-reconstructed using KNN imputation, as described in the previous paragraph. First, we construct a mask bank comprised of m masks, where $m = \sum_{j=1}^{r} \binom{d}{j}$ and r designates the maximum number of features masked simultaneously. The mask bank comprises all possible combinations of j masked features for $j \leq r$. Each mask corresponds to a d-dimensional vector of 0 and 1, where 1 indicates that the corresponding feature will be masked. Let us denote as $\hat{\mathbf{z}}^{(\ell)}$ the reconstructed sample $\mathbf{z}$ for mask ℓ, take $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

Algorithm 2 Pseudo Python Code for Mask-KNN

Require: $\mathscr{D}_{train} \in \mathbb{R}^{n_{train} \times d}$, $\mathscr{D}_{val} \in \mathbb{R}^{n_{val} \times d}$, k, $mask\_bank$, $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$
Mask-KNN $\leftarrow dict()$
B $\leftarrow$ random sample of size b from $\mathscr{D}_{train}$
for `mask` $\in mask\_bank$ **do**
  for `idx` $\in$ `range`(n_{val}) **do**
    $\mathbf{z} \leftarrow \mathscr{D}_{val}$[`idx`, :]
    $\tilde{\mathbf{z}} \leftarrow apply\_mask(\mathbf{z}$, `mask`$)$
    $\mathbf{X} \leftarrow (\tilde{\mathbf{z}}, \mathrm{B})^\top$
    $\hat{\mathbf{X}} \leftarrow$ KNNImputer$(\mathbf{X}, k)$
    $\hat{\mathbf{z}} \leftarrow \hat{\mathbf{X}}[0, :]$
    Mask-KNN[`idx`] += $d(\mathbf{z}, \hat{\mathbf{z}})$
  end for
end for

a distance measure, *e.g.* the ℓ_2-norm, then the anomaly score for sample $\mathbf{z}$ is given as

$$\text{Mask-KNN}(\mathbf{z}) = \sum_{\ell=1}^{m} d(\mathbf{z}, \hat{\mathbf{z}}^{(\ell)}) \tag{17}$$

We give the pseudo-code of this method in alg. 2.

Implementation For simplicity, we set r to 2 for all experiments, except for large datasets ($n > 200,000$), for which r was set to 1 for computational reasons. We set k, the number of neighbors, to 5 as for the vanilla KNN implementation. When present, categorical features were encoded using one-hot encoding. Except for large datasets ($n > 200,000$) with many features, d, such as ForestCover, Fraud, and Backdoor, we set B as the entire training set. Otherwise, we take a random subsample of size $b = 10,000$. We use the imputation strategy described in equation 16 to reconstruct the masked sampled. We report the results of this experiment in table 4.6 and compare the performance of Mask-KNN to the vanilla transformer and NPT-AD. We run the algorithm 20 times for each dataset, except for ForestCover, Fraud, and Backdoor, for which report an average over 10 runs for computational reasons. The mean rank, provided in table 4.6, was computed, including each architecture of each approach. For completeness, we also include a table containing the mean rank of all approaches, including MasK-KNN, in table 4.7.

Results We observe that Mask-KNN obtains satisfactory results on a significant share of the tested datasets, *e.g.* pendigits, satellite; while also displaying poor performance on some datasets such as forest or backdoor compared to NPT-AD. Several factors can account for this. First, NPTs automatically select the number of relevant samples on which to rely to reconstruct the masked features, thus making this approach much more flexible than Mask-KNN, which has a fixed number of neighbors. Second, NPT-AD relies on attention mechanisms to learn the weights attributed to relevant samples while Mask-KNN relies on the ℓ_2-distance. Although the ℓ_2-distance offers a precise measure of similarity based on geometric distance, the attention mechanism can capture much more complex relations between samples. Finally, NPT-AD not only relies on sample-sample dependencies to reconstruct the mask features, but it also attends to feature-feature dependencies.

The strong performance of NPT-AD on datasets where Mask-KNN also performs well supports the

Table 4.6: Anomaly detection F1-score (↑). We perform a 5% T-test to test whether the difference between the highest metrics for each dataset is statistically significant. **Apart from this table, Mask-KNN was not included in the computation of the mean rank.** The mean rank for the F1-score of all approaches including Mask-KNN is displayed in table 4.7.

Method	Transformer	NPT-AD	Mask-KNN
Wine	23.5±7.9	**72.5**±7.7	28.0±18.1
Lympho	88.3±7.6	**94.2**±7.9	60.0±12.2
Glass	14.4±6.1	**26.2**±10.9	**26.7**±5.4
Vertebral	12.3±5.2	20.3±4.8	**24.7**±5.9
Wbc	66.4±3.2	**67.3**±1.7	**68.1**±3.0
Ecoli	75.0±9.9	**77.7**±0.1	63.9±6.9
Ionosph.	88.1±2.8	**92.7**±0.6	89.7±0.9
Arrhyth.	59.8±2.2	60.4±1.4	**62.9**±2.4
Breastw	**96.7**±0.3	95.7±0.3	96.2±0.7
Pima	65.6±2.0	**68.8**±0.6	63.5±1.8
Vowels	28.7±8.0	**88.7**±1.6	84.3±4.9
Letter	41.5±6.2	**71.4**±1.9	56.7±3.2
Cardio	68.8±2.8	**78.1**±0.1	69.7±2.0
Seismic	19.1±5.7	**26.2**±0.7	**26.2**±1.7
Musk	**100**±0.0	**100**±0.0	**100**±0.0
Speech	6.8±1.9	**9.3**±0.8	**10.2**±2.9
Thyroid	55.5±4.8	**77.0**±0.6	31.6±5.4
Abalone	42.5±7.8	**59.7**±0.1	43.2±7.0
Optdigits	61.1±4.7	62.0±2.7	**89.0**±1.0
Satimage	89.0±4.1	**94.8**±0.8	93.7±1.7
Satellite	65.6±3.3	74.6±0.7	**77.8**±0.4
Pendigits	35.4±10.9	**92.5**±1.3	**93.1**±1.2
Annthyr.	29.9±1.5	**57.7**±0.6	19.6±1.2
Mnist	56.7±5.7	**71.8**±0.3	69.7±2.0
Mammo.	17.4±2.2	**43.6**±0.5	38.7±1.7
Shuttle	85.3±9.8	**98.2**±0.3	95.2±0.5
Mulcross	**100**±0.0	**100**±0.0	**100**±0.0
Forest	21.3±3.1	**58.0**±10	7.6±1.2
Campaign	47.0±1.9	**49.8**±0.3	42.0±0.3
Fraud	54.3±5.2	**58.1**±3.2	41.8±1.1
Backdoor	**85.8**±0.6	84.1±0.1	10.2±0.6
mean	57.4	**68.8**	57.5
mean std	4.3	**2.0**	3.1
mean rk	8.0	**3.4**	6.3

fact that NPT-AD effectively captures sample-sample dependencies. Moreover, NPT-AD outperforms Mask-KNN on most datasets where the vanilla transformer performs well, highlighting the crucial role of feature-feature dependencies on specific datasets. The results displayed in table 4.6 show that NPT-AD manages to capture feature-feature and sample-sample dependencies to reconstruct samples when sample-sample dependencies are not sufficient.

Furthermore, we anticipate observing distinct performance dynamics in NPT-AD compared to the vanilla transformer and Mask-KNN. In scenarios where the identification of anomalies relies heavily

on feature-feature dependencies, such as datasets where the vanilla transformer significantly outperforms Mask-KNN, we anticipate NPT-AD to surpass Mask-KNN. This superiority is attributed to NPT-AD's capacity to harness feature-feature dependencies effectively. Conversely, in datasets where sample-sample dependencies play a pivotal in detecting anomalies, *i.e.* datasets on which Mask-KNN outperforms the vanilla transformer, we expect NPT-AD to surpass the vanilla transformer. This relationship between the performance metrics across the three approaches would validate that NPT-AD effectively integrates feature-feature and sample-sample dependencies.

As displayed in table 4.6, our experiments show that we observe this performance behavior. For instance, on datasets where Mask-KNN significantly outperforms the transformer by a sizable gap, *e.g.* glass, vertebral, or vowels, NPT-AD also outperforms the transformer significantly. Similarly, on datasets where the transformer outperforms Mask-KNN, *e.g.* lymphography, WBC, or Forest Cover, NPT-AD also performs significantly better than Mask-KNN. This performance relation can help identify which datasets require which type of dependency to correctly flag anomalies using reconstruction-based methods. We propose such classification in table B.9 in appendix B.3.

Table 4.7: Mean rank (F1-score) for the experiments conducted, without Mask-KNN and with Mask-KNN

Method	mean rank	mean rank (w/ Mask-KNN)	diff.
DROCC (abalone)	11.5	12.4	+0.9
GOAD (thyroid)	8.4	9.1	+0.7
NeuTraL-AD (arrhythmia)	9.6	10.3	+0.7
Internal Cont.	3.7	4.0	**+0.3**
COPOD	10.5	11.0	+0.5
IForest	7.6	8.1	+0.5
KNN	5.2	5.6	+0.4
PIDForest	11.5	12.2	+0.7
RRCF	12.6	13.4	+0.8
Transformer	7.5	8.0	+0.5
Mask-KNN	N/A	6.6	N/A
NPT-AD	**3.1**	**3.4**	**+0.3**

4.7 External Retrieval Modules Ablation Study

We randomly selected a subset of the dataset from the benchmark for the ablation study conducted hereafter. We use this subset of datasets in each experiment for computational reasons and to avoid cherry-picking the hyperparameters.

4.7.1 Choice of k

We investigate the impact of varying k, the number of *helpers*, for both the transformer augmented by `attention-bsim` and the KNN module since they are the two best-performing retrieval-augmented models. We keep hyperparameters constant and only make k vary between runs. We report in table 4.8 the obtained results for both architectures for values of $k \in \{0, 5, 25, 50, 200, 500, -1\}$, where -1

implies that $\mathscr{H} = \mathscr{C}$.

A noticeable trend is that both architectures obtain the best results with moderate numbers of helper, *i.e.* $k \in \{25, 50\}$, and have worse performance for smaller values of k. This observation suggests that performance displayed in tables 4.4, B.10, and B.11 could be improved for optimized values of k

Table 4.8: Comparison of transformer+`attention-bsim` and transformer+`KNN` across values of k. Here -1 stands for $\mathscr{H} = \mathscr{C}$. Some values are N/A either because it is not relevant to compute (*e.g.* -1 for KNN) or when there are not enough samples in the training set for the selected value of k.

k	0	5	25	50	200	500	-1
			transformer+`attention-bsim`				
Abalone	42.5±7.8	53.0±6.4	**54.9**±5.4	**55.0**±5.4	52.0±5.6	54.0±6.5	53.0±5.7
Satellite	65.6±3.3	71.5±2.4	71.3±1.3	71.2±1.6	70.8±1.8	71.2±1.7	**71.9**±1.5
Lympho	88.3±7.6	91.7±8.3	**93.3**±8.2	91.7±8.3	N/A	N/A	90.0±8.1
Satimage	89.0±4.1	88.8±3.8	88.4±3.8	89.4±4.2	88.8±4.3	89.1±4.3	**93.2**±1.7
Thyroid	55.5±4.8	**56.9**±5.3	55.9±5.2	56.3±5.2	56.4±5.2	55.9±5.6	55.8±6.3
Cardio	81.0±4.1	81.2±1.6	**81.9**±1.4	**81.9**±1.4	**81.9**±1.4	**81.9**±1.4	80.6±2.4
Ionosphere	88.1±2.8	89.4±5.0	90.2±4.5	89.8±4.3	N/A	N/A	**91.7**±2.1
mean	70.3	76.1	**76.6**	76.5	70.0	70.4	**76.6**
mean std	4.9	4.7	4.3	4.3	3.7	3.9	4.0

4.7.2 Choice of λ

Table 4.9: Comparison of transformer+`attention-bsim` across values of λ

λ	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Abalone	42.5	44.3	47	47.3	46.5	**53**	46.5	45.8
Satellite	65.6	72	72	72.5	73	71.9	**73.5**	73.4
Lympho	88.3	90.8	90.8	91.7	**92.5**	90	91.7	90
Satim.	89	94.5	**94.6**	94.1	94.1	93.2	93.8	94.1
Thyroid	55.5	57.2	**57.4**	56.6	56.6	55.8	57.1	57.3
Cardio	68.8	**80.9**	80.6	**80.9**	**80.9**	**80.9**	**80.9**	80.7
Ionosp.	88.1	90.3	92.1	92.2	91.2	91.7	**92.3**	91.7
mean	71.1	75.7	76.4	76.5	76.4	**76.6**	76.5	76.1

We analyze the performance variation for the best-performing external retrieval module, namely transformer+`attention-bsim`, for varying values of λ. We compute the average F1-score over 10 runs for each value of λ in $\{0.1, 0.2, \ldots, 0.7\}$ while keeping the same hyperparameters. We report the results in table 4.9.

We observe a slight variation of the average metric across the datasets when setting lambda values from 0.1 to 0.7. The maximum value is obtained for 0.5, but the difference with other values of λ is non-significant. Nevertheless, we observe significant differences between the obtained results for isolated datasets for different values of λ. This observation supports the idea that an optimal

value of λ exists for each dataset, which may differ between datasets.

4.7.3 Location of the retrieval module

Table 4.10: Comparison of the performance of the transformer + `attention-bsim` model for different retrieval module architecture based on the F1-score (↑).

Retrieval Aggregation	`post-enc` `post-enc`	`post-emb` `post-enc`	`post-emb` `post-emb`
Abalone	**53.0**±5.7	47.0±7.6	30.5±14
Satellite	**71.9**±1.5	60.5±0.9	65.1±2.3
Lympho	**90.0**±8.1	**91.6**±8.3	84.2±11.1
Satimage	**93.2**±1.7	50.8±17.0	65.9±15.3
Thyroid	55.8±6.3	55.2±5.9	**58.1**±5.0
Ionosphere	**91.7**±2.1	88.2±1.4	**91.3**±2.0
mean	**75.9**	65.6	65.9
mean std	**4.2**	6.9	8.3

We investigate the impact of the retrieval module's location on the retrieval-augmented models' performance. To do so, we focus on the transformer+`attention-bsim` model since it has shown to be the best-performing retrieval-based method. We compare three different architectures:

- (`post-enc`, `post-enc`) for *post-encoder* location and *post-encoder* aggregation: the architecture detailed in figure 4.2,
- (`post-emb`, `post-enc`), the architecture in which the retrieval module is located after the embedding layer, but the aggregation is still located after the encoder,
- (`post-emb`, `post-emb`) the architecture where both retrieval and aggregation are located after the embedding layer.

We do not report the results for the cardio dataset since it failed to converge for the (`post-emb`,`post-enc`) and (`post-emb`,`post-emb`) architectures and output NaN values in inference. We used the same hyperparameters for all three architectures. For (`post-emb`,`post-enc`) and (`post-emb`,`post-emb`) architectures we report the average over 10 runs. We display the results in table 4.10.

We observe that the (`post-enc`, `post-enc`) architecture obtains the highest mean and lowest mean standard deviation over the tested datasets by a sizable margin. The transformer encoder's expressiveness allows for better representations of a data sample than the embedding layer and may account for such results. Indeed, this shows that the retrieval modules that receive the embedded representation as inputs are less able to select the relevant sample to foster mask reconstruction and anomaly detection performance. Moreover, since the encoder and retrieval modules are trained conjointly, the retrieval module can help the encoder converge to a state that favors sample representations that allow relevant clusters to be formed.

Table 4.11: Comparison of the performance of the transformer + `attention-bsim` model for two mask bank set-ups. The same model was used for inference in both set-ups. We report an average over 10 different splits of the data.

Mask bank	random	deterministic
Abalone	43.0±14.9	**53.0**±5.7
Satellite	58.4±5.5	**71.9**±1.5
Lympho	86.7±8.5	**90.0**±8.1
Satimage	47.7±20.5	**93.2**±1.7
Thyroid	**55.6**±6.1	**55.8**±6.3
Cardio	**81.3**±1.6	80.6±2.4
Ionosphere	85.2±4.3	**91.7**±2.1
mean	65.4	**76.6**
mean std	8.8	**4.0**

4.7.4 Random mask bank

We also investigate the impact of constructing a random mask bank for inference instead of selecting a deterministic mask bank as discussed in section 4.4.3. We construct for inference a random mask bank composed of the *same number of masks* as for the deterministic mask bank and use the same probability p_{mask} as used to train the model. We compare the performance of the transformer model+`attention-bsim` based on the F1-score for the two set-ups and display the results in table 4.11. The deterministic mask bank detects anomalies better on most tested datasets. When computing the anomaly score with the deterministic mask bank, the model obtains an average F1-score of 76.6 over the 7 datasets, while with the random mask bank, the model obtains 65.4. Moreover, as expected, we also observed a significantly more significant standard deviation between runs. We might expect the standard deviation to decrease as the number of masks increases, which would induce significant computational overhead.

4.8 Conclusion

Limitations As with most non-parametric models, our retrieval-augmented models display a higher complexity than parametric approaches. These approaches can scale well for datasets with a reasonable number of features d; however, for large values of d, these models involve a high memory cost.

In these experiments, we have proposed a novel deep anomaly detection method designed explicitly for tabular datasets. To the best of our knowledge, our approach is the first to utilize feature-feature and sample-sample dependencies to identify anomalies successfully. Our experiments involving an extensive benchmark of tabular datasets demonstrate the effectiveness of retrieval-based approaches since NPT-AD outperforms existing state-of-the-art methods in terms of F1-score and AUROC. While NPT-AD outperforms other external retrieval module-augmented approaches, this statement only applies to anomaly detection relying on the self-supervised mask reconstruction approach. NPTs are bound to some inductive biases and cannot be used to learn all possible tasks relevant for anomaly detection. On the contrary, external retrieval modules can augment most existing AD methods by including inter-sample dependencies and improving the performance of some approaches.

Future Work Overall, NPT-AD has shown strong performance for anomaly detection on tabular data. It relies on standard multi-head self-attention through the Attention Between Datapoints (ABD) mechanism, close to the `v-attention` module, to leverage inter-sample relations. Our findings may invite further research on modifying the ABD mechanism involved in NPTs to improve their AD performance.

Chapter 5

Experiments on a Real-Life Fraud Dataset

Abstract

This study explores the application of anomaly detection (AD) methods in imbalanced learning tasks, focusing on fraud detection using real online credit card payment data. We assess the performance of several recent AD methods and compare their effectiveness against standard supervised learning methods. Offering evidence of distribution shift within our dataset, we analyze its impact on the tested models' performances. Our findings reveal that LightGBM exhibits significantly superior performance across all evaluated metrics but suffers more from distribution shifts than AD methods. Furthermore, our investigation reveals that LightGBM also captures the majority of frauds detected by AD methods. This observation challenges the potential benefits of ensemble methods to combine supervised, and AD approaches to enhance performance. In summary, this research provides practical insights into the utility of these techniques in real-world scenarios, showing LightGBM's superiority in fraud detection while highlighting challenges related to distribution shifts.

This chapter is a modification of the following publication:

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2023). Comparative evaluation of anomaly detection methods for fraud detection in online credit card payments.

5.1 Introduction

Detecting fraudulent behaviors has emerged as a critical problem that has garnered significant attention from practitioners and scholars alike. In sectors such as banking, frauds incurred an estimated annual cost of $28.58 billion in 2021, as highlighted by the Nilson Report 2021[1]. To address the challenge of identifying frauds within regular credit card payments, banks have increasingly turned to machine learning techniques, known for their effectiveness in many classification tasks, particularly with unstructured data.

Two critical fraud detection features pose challenges to constructing effective and accurate classifiers: highly imbalanced classes and distribution shifts.

[1] https://nilsonreport.com/

Imbalanced datasets arise due to the significant disparity in the number of genuine transactions compared to fraudulent ones, making it difficult for traditional classification algorithms to generalize accurately. Highly imbalanced datasets are a specific subset of imbalanced datasets in which the positive class represents less than 1% of the samples; this situation is also referred to as rarity (Bauder et al., 2018). Moreover, distribution shift occurs as fraudsters constantly adapt their strategies, causing a discrepancy between the training and testing data distributions, thereby hampering the performance of machine learning models.

Learning from imbalanced datasets is a critical topic with implications across various real-life applications. Extensive research has highlighted the consequences of imbalanced learning on canonical classifiers, revealing that most standard classifiers are ill-suited for imbalanced settings. For instance, the limitations of standard machine learning techniques when confronted with imbalanced datasets are examined comprehensively by Yanmin et al., 2011. This study also sheds light on the struggles faced by backpropagation algorithms in converging within imbalanced set-ups, as the dominant majority class can overwhelm the gradient vector used for weight updates in neural networks. In contrast, Gradient Boosted Decision Trees (GBDT) are often considered more resilient to imbalanced settings due to their focus on particularly challenging examples (Frery, 2019), thus enabling them to prioritize the minority class more effectively. Highly imbalanced datasets are among the most challenging as research (Japkowicz & Stephen, 2002) has shown how learners display decreasing performance as imbalance becomes more severe.

In addition to imbalanced datasets, distribution shift is another crucial challenge in detecting fraud. Standard machine learning techniques perform well when the training and testing dataset distributions are similar, if not identical. However, domain shift occurs when the distribution of the test dataset deviates from the original training distribution and thus hinders standard classifiers' performance. Fraud detection involves an iterative game between fraudsters and banks. Fraudsters continually strive to produce increasingly inconspicuous fraudulent behaviors. At the same time, banks aim to detect fraud as accurately as possible while avoiding false negatives. This dynamic nature of fraud detection presents significant challenges for machine learning algorithms.

These characteristics of fraud detection underscore the need for methodologies capable of effectively handling distribution shifts and highly imbalanced datasets. In response to these challenges, researchers have proposed using anomaly detection (AD) methods, which promise to exhibit robustness when confronted with distribution shifts and extreme class imbalances. Specifically, AD involves the identification of anomalies within a dataset by delineating deviations from a predefined notion of normality. Anomaly detection methods typically characterize the normal distribution solely based on normal samples during training. Consequently, AD has been regarded as particularly well-suited for imbalanced and extremely imbalanced settings. By design, AD methods do not experience performance deterioration when faced with highly skewed class distributions, as the training process solely requires normal samples. Moreover, assuming only fraudulent behaviors change over time, AD models should be more robust to distribution shifts than standard supervised approaches. Indeed, if the normal distribution is well characterized, they should always be able to exclude new types of anomalies.

In this work, we empirically investigated AD for fraud detection tasks, exploring their capabilities and limitations. By empirically evaluating various AD methods on a real-world dataset characterized by distribution shifts and extreme class imbalances, we aim to provide insights into the suitability

and effectiveness of these techniques for addressing the challenges inherent to fraud detection. In addition to evaluating the performance of AD methods, we conduct a comparative analysis with Gradient Boosted Decision Trees (GBDT), the prevalent choice for machine learning tasks on tabular data (Grinsztajn et al., 2022), to gauge the added value of AD approaches. We rely on the LightGBM implementation (Guolin et al., 2017) and show that GBDT suffer significantly more from distribution shift than AD methods while displaying substantially better fraud detection performance than all tested AD methods. This chapter is structured as follows: in the next section, we discuss works related to our application; in section 5.3, we discuss in detail the experiments conducted on our dataset; in section 5.4 we present the obtained results; in section 5.5 we discuss the results and finally in section 5.6 we conclude.

5.2 Related Works

Anomaly detection encompasses two types of algorithms: supervised and unsupervised. In the case of supervised AD, one disposes of the label and indirectly uses it in the training process by building a training set solely composed of samples belonging to the *normal* class. Unsupervised AD methods involve situations where the label is unavailable, and anomalies must be directly identified within a dataset containing both *normal* samples and anomalies. While supervised approaches can be used in situations where the imbalance is too severe for standard supervised approaches to work, unsupervised approaches are usually confined to applications that consist in removing samples that may hinder another model's performance on a particular task, e.g., mislabeled samples or outliers. Since we dispose of labels in the context of fraud detection, we will focus on supervised anomaly detection.

Supervised anomaly detection methods differ from standard supervised approaches because labels are only used indirectly. Indeed, standard supervised approaches consist in training a classifier using a dataset

$$\mathscr{D}_{train} = \{(\mathbf{x}_i, y_i) : x_i \in \mathscr{X}, y_i \in \mathscr{Y}\}_{i=1}^{n}, \tag{1}$$

using both sample features x_i and labels y_i. Moreover, $\mathscr{D}_{train}$ contains samples from each class in $\mathscr{Y}$. On the contrary, supervised anomaly detection methods use the label to build a training set solely composed of a single class, referred to as the *normal* class. In the case of binary classification, the *normal* class is the majority class, e.g., the legit payments in the case of fraud detection, and the training set can then be constructed as

$$\mathscr{D}_{train}^{\text{AD}} = \{\mathbf{x}_i : y_i = 0\}_{i=1}^{n}. \tag{2}$$

This anomaly detection framework aims to characterize the normal distribution, $p(\mathbf{x} \mid y = 0)$. In inference, this characterization determines whether a sample belongs to the normal distribution or should be seen as an anomaly.

As a field of research, anomaly detection can be divided into several non-exhaustive categories: one-class classification (OCC), reconstruction-based methods, and self-supervised methods.

One-Class Classification In contrast to traditional machine learning classification problems, one-class classification (OCC) approaches aim to identify samples that do not belong to a specific class by characterizing the distribution of that class. These discriminative models learn a decision boundary

using only samples from the designated *normal* class, thereby circumventing the direct estimation of the class distribution. During the inference phase, samples are classified as either belonging to the *normal* class or not, without making any assumptions about the *anomaly* class. One-class support vector machines (OCSVM) (Schölkopf et al., 1999) and support vector data description (SVDD) (Tax & Duin, 2004) are popular OCC methods that rely on kernels to map the data space to a Hilbert space, where a decision boundary is learned. Recent methods (Ruff et al., 2018; Ruff et al., 2020) have introduced extensions to OCC methods that incorporate deep neural networks to alleviate the computational complexity associated with kernels. Other OCC tree-based approaches can be found in the OCC, such as isolation forest (IForest) (Liu, Ting, et al., 2008), extended isolation forest (Hariri et al., 2021), Robust Random Cut Forest (RRCF) (Guha et al., 2016) and PIDForest (Gopalan et al., 2019). Other methods have relied on sample-sample dependencies to identify anomalies, such as TracInAD (Thimonier et al., 2022) relying on influence measures or approaches based on k-nearest neighbors (KNN). In the latter, anomalies are identified by measuring the distance of each sample to its k-nearest neighbors (Angiulli & Pizzuti, 2002; Ramaswamy et al., 2000): higher distance indicating abnormality.

Reconstruction-based methods Reconstruction-based anomaly detection methods rely on the assumption that different distributions generate normal samples and anomalies. Consequently, training a model to reconstruct samples from the *normal* distribution aims to achieve low reconstruction error for any sample belonging to this distribution. Conversely, anomalies that are believed to stem from a distinct distribution should exhibit significantly higher reconstruction errors.

One of the most prevalent shallow reconstruction-based anomaly detection methods employs Principal Component Analysis (PCA) or Bayesian PCA (Dutta et al., 2007; Huang et al., 2006). Autoencoders (Finke et al., 2021), regularized autoencoders like Variational Autoencoders (VAEs) (Pol et al., 2019), and memory-augmented deep autoencoders (Gong et al., 2019b) have also been leveraged for anomaly detection. Recently, Kim et al., 2020 proposed a novel methodology for anomaly detection using autoencoders that incorporate the hidden representations of the original and reconstructed samples. Instead of solely comparing the reconstructed sample and the original sample, the authors suggest comparing the hidden representations of both samples by passing the reconstructed sample through the autoencoder. In addition, recent approaches have explored attention-based architectures for reconstructing masked features of samples, as exemplified by NPT-AD (Thimonier et al., 2024a). Other related methods do not directly compute a reconstruction error and only focus on estimating either the entire *normal* distribution such as ECOD (Li et al., 2022) or local *normal* distributions as proposed in local outlier factor (LOF) (Breunig et al., 2000).

Self-supervised methods The literature also features self-supervised approaches employing pretext tasks for anomaly detection (Bergman & Hoshen, 2020; Qiu et al., 2021; Tack et al., 2020). In GOAD (Bergman & Hoshen, 2020), several affine transformations are applied to each sample in the training set, while a classifier is trained to predict the specific transformation applied to a transformed sample. During testing, since the classifier was exclusively trained on *normal* samples, it is expected to struggle in correctly predicting the transformation for anomaly samples. Similarly, Qiu et al., 2021 propose NeuTraL-AD, a contrastive framework in which they transform samples using neural mappings instead of affine transformations. The objective is to learn transformations that maintain similarities in a semantic space between transformed samples and their untransformed counterparts

while different transformations are easily distinguishable. In inference, the contrastive loss utilized to optimize the parameters serves as the anomaly score. More recently, Shenkar and Wolf, 2022 introduced a self-supervised methodology for anomaly detection that maximizes the mutual information among the elements of a sample's features using contrastive learning. By maximizing the mutual information, the method effectively captures the underlying structure of normal samples and identifies deviations indicative of anomalies.

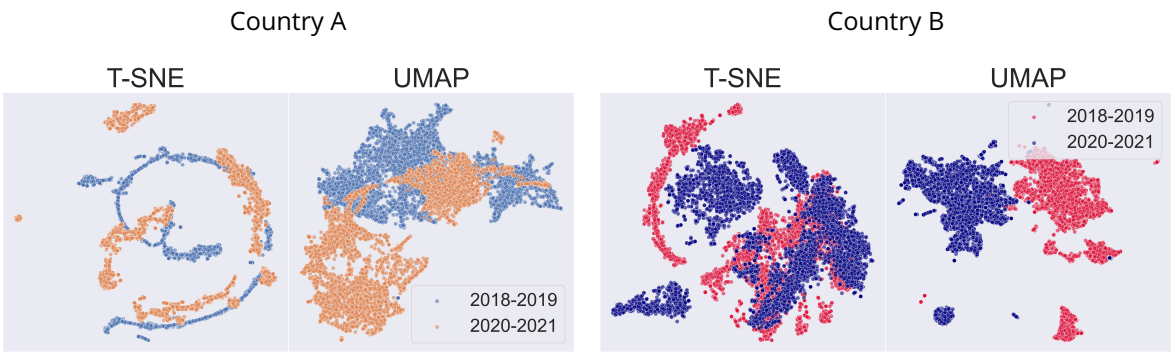
Supervised classification on tabular data Although deep-learning models have become ubiquitous for various tasks involving natural language processing (NLP) and computer vision (CV), applying these models to tabular data remains very challenging. Some recent methods (Gorishniy et al., 2021; Kadra et al., 2021; Somepalli et al., 2021) have shown promising results when applying deep learning models tailored for tabular data. However, in recent work (Gorishniy et al., 2023; Grinsztajn et al., 2022), authors discuss how neural networks tend to struggle with this data type in comparison with other methods based on gradient-boosted decision trees (GBDT). In most scenarios, approaches such as XGBoost (Chen & Guestrin, 2016) or LightGBM (Guolin et al., 2017) have been shown to surpass deep learning algorithms. This type of approach remains the go-to method for practitioners due to its strong classification performance and its simplicity of training in comparison with deep methods. Moreover, GBDT models such as LightGBM and XGBoost are often considered particularly suited for imbalanced and extremely imbalanced set-ups since these models focus on particularly hard-to-classify samples (Frery, 2019), generally the minority class, and thus offer strong performance in comparison to other standard machine learning models.

5.3 Experiments and Datasets

5.3.1 Dataset

We dispose of a labeled dataset of online credit card payments made available by a major French bank. Our dataset contains 145 features describing raw characteristics of payments (e.g., amount, currency) as well as features that we computed, such as rolling sums or rolling means. Our dataset contains 480 million online transactions from the first day of 2018 until the last day of 2021. The dataset comprises two independent datasets merged, one from 2018 to 2019 and the other from 2020 to 2021. This dataset displays the characteristic of highly imbalanced classes[2] discussed in section 5.1 since the proportion of legit payments vastly outnumbers the proportion of frauds. We remove cards with less than 50 payments, cards for which the proportion of frauds exceeds 50%, and cards with too few payments since they would not have derived features (e.g., rolling means) with meaningful values and would risk hindering the models' performance. Similarly, we also omit cards with too many frauds in their payment history since they would also be problematic as they might pollute the fraud distribution. Overall, this preprocessing reduces the dataset to 192 million payments. Most methods we wish to test would be intractable with such dataset size and require further dimension reduction. Thus, we restrict our analysis to two countries with 3 million payments (1.5% of total dataset size) and 20 million payments (10.3% of total dataset size), respectively. Moreover, restricting our analysis

[2] Due to confidentiality, we cannot discuss the exact proportion of frauds within our dataset.



OTDD	2018	2020
2018	56.5	**64.4**
2020	**61.9**	53.1

OTDD	2018	2020
2018	54.7	**69.2**
2020	**73.6**	56.9

Figure 5.1: T-sne (van der Maaten & Hinton, 2008) and UMAP (McInnes et al., 2020) bi-dimensional representation of payments in countries A and B for each period. These graphs give evidence of a distribution shift for both countries' payment behaviors between 2018 and 2020 since we observe very few sample overlays. Tables give the Optimal Transport Dataset Distance (Alvarez-Melis & Fusi, 2020) between subsamples for each country. We observe a much higher distance between subsamples from the same country between different periods than for the same period

to one country at a time should help models learn since payment distributions likely differ between countries.

Distribution shift

We argue that our datasets undergo a distribution shift and that the distributions of legit and fraudulent payments differ between 2018 and the end of 2021. For instance, as supported by (Gu et al., 2021), COVID-19 has caused online payment behaviors to change drastically over time. To further support our statement, we display in figure 5.1 the t-sne (van der Maaten & Hinton, 2008) and UMAP (McInnes et al., 2020) representations for years 2018-2019 and 2020-2021 for both countries. We observe significant dissimilarities between datasets. For country A, we observe an entire subsample of payments made in 2020-2021 at the bottom left of the UMAP graph, which does not exist for the 2018-2019 period. Similarly, for the t-sne representation of country A, we observe a similar pattern with few data sample overlays between periods. Moreover, for country B, we also observe a very scarce overlay on the graph between periods, especially for the UMAP representation. To further investigate whether distribution shift is present in our datasets, we rely on the Optimal Transport Dataset Distance method (OTDD) (Alvarez-Melis & Fusi, 2020) to measure the distance between datasets from each period. This method relies on optimal transport, which measures the distance between distributions. For each country, we created two subsamples of 5000 observations for each period and compared the distance between the subsample of the same period and between periods. Results of this analysis are shown in the tables in figure 5.1 and indicate that the distance between the dataset increases across periods. This is especially true for country B.

Data splits and preprocessing

For a fair comparison between supervised approaches and anomaly detection methods, we split the 2018 datasets of each country into two separate datasets constituted of legit payments and frauds. We take a training set representing 75% of the 2018 dataset for each country and use the 25% remaining for the test set. We include the frauds in the training set for LightGBM, while the frauds are excluded from the training set for anomaly detection approaches. The 2020 dataset of each country serves entirely as test sets. Overall, the considered dataset used for every tested model contains features describing characteristics of the payment (e.g., amount of the transaction, the currency used, duration since the last transaction.), among which eight are categorical. All eight categorical features are encoded using Catboost encoding following (Bourdonnaye & Daniel, 2021). Continuous features are scaled to be in $(0, 1)$ through standard normalizing by removing the mean and reducing to unit variance.

5.3.2 Experimental Settings

In order to evaluate the suitability of anomaly detection methods for fraud detection, we conduct a comprehensive analysis using state-of-the-art approaches on our dataset. We investigate both deep learning-based techniques designed for tabular data, such as the self-supervised approaches of GOAD (Bergman & Hoshen, 2020), NeuTraL-AD (Qiu et al., 2021), the contrastive approach proposed in Shenkar and Wolf, 2022, and the reconstruction-based approach of NPT-AD (Thimonier et al., 2024a). Additionally, we include non-deep learning methods, namely Isolation Forest (Liu, Ting, et al., 2008), ECOD (Li et al., 2022), COPOD (Li et al., 2020), and the KNN AD approach (Angiulli & Pizzuti, 2002; Ramaswamy et al., 2000), as they have demonstrated remarkable performance on various tabular datasets. As a baseline, we employ LightGBM (Guolin et al., 2017), a well-established supervised classification method, to assess the added value of anomaly detection compared to standard supervised techniques in this context.

To effectively compare the performance of various models in detecting fraud, we employ three commonly used metrics from the anomaly detection literature and the banking industry for real-life model evaluation. The anomaly detection literature has widely adopted the F1-score and the AUROC as evaluation metrics. While the AUROC is suitable for balanced class distributions, it may not fully account for class proportions, which is crucial in assessing performance in imbalanced set-ups. We include the Area Under the Precision-Recall Curve (AUPRC) to address this limitation, which is better suited for imbalanced datasets. In support of this choice, Davis and Goadrich, 2006 has demonstrated that a model dominates in the ROC space if and only if it also dominates in the PR space.

For the F1-Score, whose value depends on a threshold, we adhere to the practices of the anomaly detection literature by selecting the threshold for the anomaly score that predicts an equal number of fraud cases as those present in the dataset. This approach ensures a fair and consistent evaluation across all models.

5.4 Results

5.4.1 Models Hyperparameters

We implemented the non-deep models using the PyOD library (Zhao et al., 2019) with default hyperparameter values and LightGBM (Guolin et al., 2017) also with default parameters. For the deep learning approaches, we adopted the hyperparameters suggested in the original papers for the dataset that most resembled our datasets. Specifically, we set the hyperparameters to those used for the KDD dataset for NeuTraL-AD (Qiu et al., 2021), GOAD (Bergman & Hoshen, 2020), and NPT-AD (Thimonier et al., 2024a) using their official implementations available on GitHub. Regarding the approach proposed by Shenkar and Wolf, 2022, we kept the parameters at their default values as specified in their implementation. Deep models were trained on 4 Nvidia GPUs V100 16Go/32Go, while non-deep models were trained on 2 Intel Cascade Lake 6248 processors (20 cores at 2.5 GHz), thus 40 cores.

5.4.2 Fraud Detection Performances

Table 5.1: Performance metrics of AD models in comparison with LightGBM (Guolin et al., 2017). Results are averaged over 10 runs for 10 different data splits; the standard deviation is displayed below the metrics. We report the F1-Score in terms of percentage. The highest metric over all models is highlighted in bold, while the highest metrics among the AD method are underlined. We perform 5% t-test between the highest metrics to measure whether they are statistically different

	Country A						**Country B**					
Model	F1 (↑)		AUROC (↑)		AUPRC (↑)		F1 (↑)		AUROC (↑)		AUPRC (↑)	
	2018	2020	2018	2020	2018	2020	2018	2020	2018	2020	2018	2020
LightGBM	**21.52**	**0.7**	**90.0**	**66.5**	**18.7**	**0.3**	**17.2**	0.5	**93.5**	**75.5**	**34.8**	**2.7**
	1.6	0.3	0.4	2.4	1.8	0.05	1.9	0.4	0.3	1.2	1.5	0.3
ECOD	0.5	0.2	62.2	62.5	0.4	0.3	0.6	1.04	54.02	51.6	1.1	0.8
	0.2	0.2	0.6	1.02	0.02	0.01	0.3	0.4	0.5	0.9	0.06	0.04
COPOD	0.3	0.2	64.8	62.6	0.4	0.3	0.5	1.1	51.7	50.2	1.0	0.7
	0.2	0.2	0.5	1.0	0.02	0.01	0.2	0.4	0.6	0.9	0.05	0.04
IForest	0.2	0.2	64.1	60.6	0.4	0.2	0.7	**1.3**	60.5	46.9	1.4	0.7
	0.1	0.2	0.8	0.8	0.02	0.01	0.2	0.3	0.5	0.8	0.07	0.03
KNN	0.3	0.01	68.9	55.6	0.5	0.2	0.8	0.4	65.9	49.3	1.6	0.6
	0.12	0.04	0.8	0.9	0.02	0.01	0.2	0.3	0.6	0.7	0.08	0.02
GOAD	0.1	0.2	53.7	52.7	0.2	0.2	0.7	0.7	50.4	64.5	0.7	1.0
	0.09	0.1	1.4	1.7	0.01	0.01	0.4	0.3	2.4	1.3	0.05	0.06
NeuTraL-AD	0.6	0.02	59.1	51.5	0.4	0.2	1.5	0.4	53.2	45.2	1.08	0.6
	0.2	0.08	3.6	1.2	0.1	0.01	0.44	0.2	1.9	1.8	0.07	0.03
Internal Cont.	0.6	0.0	39.4	46.7	0.2	0.1	1.2	0.2	45.6	50.7	0.9	0.7
	0.1	0.0	1.1	0.2	0.0	0.0	0.2	0.07	2.5	0.9	0.1	0.03
NPT-AD	1.0	**0.7**	67.2	53.2	0.8	0.2	1.7	0.6	66.2	53.5	1.2	0.6
	0.07	0.06	1.3	0.7	0.01	0.03	0.1	0.03	1.1	0.4	0.1	0.06

Metrics reported in table 5.1 are averaged over 10 runs, and we performed t-tests on the highest metrics to asses whether models obtained significantly different results. Among the AD approaches, we observe that the non-deep methods demonstrate the best performance, specifically ECOD, COPOD, and KNN. However, it is worth noting that the deep learning approach NPT-AD also yields comparable results. While the AD methods achieve satisfactory results regarding the AUROC, their per-

formances are consistently poor for both the F1-Score and AUPRC metrics across both countries and periods. In contrast, LightGBM exhibits significantly better performance across all metrics, consistently achieving the highest values, except for the F1-Score on the 2020 dataset of Country B. The overall poor performance of all models, including LightGBM, for the F1-Score and AUPRC, highlights the inherent challenges associated with fraud detection. However, a noteworthy observation is the substantial performance gap between LightGBM and the anomaly detection methods.

5.5 Discussion

5.5.1 Distribution Shift

The obtained results for both countries support the hypothesis that a distribution shift occurred between 2018 and 2020 in our dataset. Across most tested approaches, we observe a significant decrease in all metrics between the 2018 and 2020 datasets for both countries. Notably, the distribution shift appears more pronounced in country B, with metrics experiencing a more significant decline. The findings presented in Figure 5.1 further corroborate this statement as the dataset distance is higher between periods than for country A. The bi-dimensional representations also show less overlap between the periods than for country A. Although the AD methods display poor overall performance, they exhibit more resilience in the face of distribution shift than LightGBM. This trend is particularly evident for ECOD and COPOD, which maintained relatively similar metrics between the 2018 and 2020 datasets for both countries. Conversely, KNN and NPT-AD experienced a significant decline in performance across both periods and datasets compared to ECOD and COPOD. While LightGBM still achieves the highest metrics for most of the 2020 dataset, it suffers a substantial drop in performance between the two periods. This drop is especially pronounced in the AUPRC and F1-Score metrics. Notably, most AD methods outperform LightGBM by a significant margin in terms of the F1-Score for the 2020 dataset of country B.

Based on these findings from our dataset, it appears that LightGBM is a favorable choice in the fraud detection framework when labels are available and no distribution shift occurs. However, in the presence of a distribution shift, retraining LightGBM on an updated dataset becomes crucial to prevent a significant performance decline.

5.5.2 Anomaly Detection Methods for Ensembling

One potential advantage of anomaly detection (AD) methods is their ability to identify fraud cases that differ from those flagged by supervised approaches. If AD models can successfully detect fraud instances supervised models cannot identify, resorting to ensembling techniques could enhance overall fraud detection performance. To investigate this further, we focused on ECOD, one of the top-performing AD methods on the dataset consisting of payments in *country A*. We examined whether the fraud cases detected by ECOD differed from those identified by LightGBM. Across the 10 iterations, we observed that, on average, 3.28% of the fraud cases in the test set were detected by ECOD but not by LightGBM. Conversely, LightGBM detected 20.41% of the fraud cases in the test set that ECOD did not flag. Notably, the 3.28% represents 10.98% of the fraud cases detected by ECOD. In other words, 89.02% of the fraud cases detected by ECOD were also detected by LightGBM. As a result, en-

Table 5.2: Comparison of performance between datasets. We compare the performance of AD methods between the public dataset (Pozzolo et al., 2015) and our dataset in country A for the 2018 period.

	Public Dataset		Our Dataset (2018)	
	F1	AUROC	F1	AUROC
ECOD	38.6 ± 0.6	94.9 ± 0.1	0.5 ± 0.2	62.2 ± 0.6
COPOD	44.7 ± 0.9	94.7 ± 0.1	0.3 ± 0.2	64.8 ± 0.5
IForest	30.3 ± 3.7	94.8 ± 0.3	0.2 ± 0.1	64.1 ± 0.8
KNN	60.5 ± 1.5	96.6 ± 0.1	0.3 ± 0.1	68.9 ± 0.8
GOAD	53.1 ± 10.2	83.5 ± 2.7	0.1 ± 0.1	53.7 ± 1.4
NeuTraL-AD	$24.3 \pm 7.$	84.8 ± 4.7	0.6 ± 0.2	59.1 ± 3.6
Internal Cont.	57.9 ± 2.8	95.2 ± 0.5	0.6 ± 0.1	39.4 ± 1.1
NPT-AD	58.1 ± 3.2	95.7 ± 0.1	1.0 ± 0.1	67.2 ± 1.3

sembling models by combining AD methods with LightGBM to enhance fraud detection performance may prove ineffective.

5.5.3 Real-life Dataset vs Publicly Available Dataset

All tested AD methods obtain low-performance metrics for both periods on our dataset. In comparison, we also evaluated the different models on a publicly available fraud dataset (Pozzolo et al., 2015) to assess whether our dataset is particularly challenging to learn from or if anomaly detection methods are irrelevant for fraud detection. We display in table 5.2 the metrics obtained by the tested AD methods for the public dataset available on Kaggle and our dataset for country A for the 2018 period. We notice a sizeable gap for all methods between the obtained metrics for the public dataset and our dataset. This statement is particularly true for the F1-Score but is also valid for the AUROC. This observation supports the idea that our dataset is particularly complex and makes training AD models challenging.

Our dataset includes a significantly larger number of features than the public dataset, and Grinsztajn et al., 2022 have shown that including non-informative features may significantly harm deep models' capacity to learn. Moreover, some categorical features have a very large cardinality, which may hinder the models' learning capacity. Obtaining improved metrics necessitates costly feature engineering that is beyond our scope.

5.6 Conclusion

In conclusion, our study highlights several key findings concerning applying machine learning techniques for fraud detection. Our results demonstrate that LightGBM consistently outperforms the tested AD methods across various evaluation metrics, emphasizing its efficacy in fraud detection tasks compared to other methods. However, we also observed that LightGBM's performances are susceptible to degradation due to distribution shifts. This finding underscores the importance of retraining LightGBM on updated datasets when there is suspicion or evidence of a distribution shift. By adapting the model to the changing data distribution, it is possible to mitigate the drop in performance and maintain its effectiveness in fraud detection. Furthermore, our investigation revealed

that ensembling techniques with AD methods would not significantly improve overall fraud detection performance. Despite the potential for AD methods to detect frauds that may elude supervised approaches, our analysis showed that LightGBM also detected most of the frauds identified by AD methods. This finding suggests limited benefits in combining AD methods with LightGBM in our specific fraud detection framework. We believe these insights may contribute to advancing the field of fraud detection and inform practitioners in selecting appropriate models and strategies for robust and accurate fraud detection systems.

Future work may involve replicating our analysis on other credit card payment datasets to determine whether our obtained results can be generalized. Furthermore, enhancing the robustness of GBDT models against distribution shifts emerges as a critical direction for further exploration. Addressing this challenge is paramount for financial institutions, as it enables them to confidently embrace machine learning techniques in fraud detection systems.

Chapter 6

Conclusion

The present work focused on anomaly detection for tabular data, with a particular emphasis on the fraud detection application. We aimed to answer three crucial questions to advance the field of anomaly detection:

I- **Are sample-sample dependencies relevant to identifying anomalies within a dataset?**

II- **Can we leverage efficiently both dependencies to detect anomalies on tabular data ?**

III- **Are anomaly detection methods effective in identifying frauds on a real-life credit-card payment dataset?**

In chapter 3, we proposed a novel anomaly detection method that relies on influence measure and obtained encouraging results compared to existing methods found in the literature. This result emphasized that (I) **sample-sample dependencies are indeed relevant to identify anomalies on some datasets**. Indeed, given the terminology proposed in (Han et al., 2022) on anomalies in tabular data that separate anomalies into four groups, *dependency anomalies*, *global anomalies*, *local anomalies* and *clustered anomalies*, it appears as local anomalies require sample-sample dependencies to be detected and cannot be efficiently detected only relying on feature-feature relations. Moreover, this work opened the possibility to combine both feature-feature and sample-sample dependencies as our method could easily be combined with existing methods (as shown in section 3.6) to augment them.

In chapter 4, we put forward the first anomaly detection method for tabular data to combine feature-feature and sample-sample dependencies. Our proposed method either introduced both dependencies internally or allowed leveraging sample-sample dependencies by introducing a retrieval module. Our approach was also the first to explicitly rely on mask reconstruction for tabular data using attention mechanisms and stochastic masking. Through extensive experiments on a benchmark of 31 tabular datasets, we obtained SOTA performance compared to recent anomaly detection on tabular data. Moreover, our ablation studies demonstrated that combining the two types of dependencies is key to efficiently identifying anomalies within datasets, as solely relying on one of the two types yields significantly lower performances. Hence, we provide proof that (II) **one can leverage both types of dependencies to detect anomalies on tabular data, and this approach offers strong anomaly detection performance.**

In chapter 5, we experimented on a dataset made available to us by a prominent French bank. We tested whether anomaly detection approaches were competitive in detecting credit card pay-

ment fraud. Based on the financial industry's practice, we compared anomaly detection methods to gradient-boosted decision trees, the prevalent method for classification tasks on tabular data. Our experiment demonstrated that supervised anomaly detection performs significantly less than gradient-boosted decision trees. Our key findings are that (III) **existing supervised anomaly detection methods obtain significantly worse performance than gradient boosted decision trees on the fraud detection task, detect frauds that are also detected by gradient boosted decision trees while also being less computationally efficient.** Nevertheless, when evaluating anomaly detection methods on a public dataset, we observe a very significant difference between the obtained metric, which may mitigate our findings and highlight the *complexity* of our dataset.

Overall, this work investigated weakly supervised anomaly detection as a possible solution to extreme imbalance classification problems on tabular data. We proposed novel algorithms that obtained encouraging results on public datasets. Nevertheless, when confronted with real-life applications, this algorithm class proved to be less effective than the best-performing supervised approach for tabular data, namely gradient-boosted decision trees. While anomaly detection remains a possible solution to extremely imbalanced, more effective methods still need to be proposed for real-life applications in the industry.

Future Work While tabular data is the most widespread data modality used in the industry, current research on machine learning is still severely oriented towards unstructured data such as text or images. This heavy investment in developing ML for text and images has improved performance on a wide range of tasks, such as entity recognition, text generation, image generation, text classification, etc., by leveraging new models or training procedures. On the contrary, the best-performing models on most tasks on tabular data often involve gradient boosting (Friedman, 2000) that dates back to the late 1990s and 2000s. Given the ubiquity of deep approaches on other modalities, tabular data may also benefit from further research on efficiently adapting deep models for tabular data.

A promising path to leveraging deep models for tabular data is that of pre-trained models similar to foundation models trained on large text corpora and fine-tuned for specific tasks. Regarding tabular data, it has been well-discussed that the heterogeneity of features is one of the main challenges that deep models face when it comes to learning from tabular data. Recent work has nevertheless demonstrated the pertinence of recent architecture, e.g. transformers for several tasks on tabular data, such as anomaly detection, as demonstrated in chapter 4. Careful feature engineering is often crucial to obtaining satisfactory performance with deep models compared to tree-based methods that often require little preprocessing to obtain high performance on tabular data. Feature engineering still permits significant improvement for tree-based approaches, but overall, the *lazy* approach is more permissive on tree-based models that can perform well without lengthy data processing. To avoid such preprocessing that often requires expert knowledge, a possible solution would be to dispose of tabular foundation models to construct relevant representations for tabular datasets to be fine-tuned later.

In this line of work, Kim et al., 2024 have constructed graph-transformer-based approaches and leveraged a general-purpose knowledge database to pre-train a model that generates representations that facilitate learning from tabular data. Their approach, CART, can be fine-tuned for any downstream tasks on tabular data. Similarly, other works, such as TabPFN (Hollmann et al., 2023), also pave the way for future tabular foundation models that may enhance the overall performance

of deep models on tabular data. Nevertheless, these methods still incur significant computational overhead and are limited to small to medium-sized datasets.

Bibliography

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying support vector machines to imbalanced data sets. *Lecture Notes Artif. Intell.*, *3201*, 39–50. https://doi.org/10.1007/978-3-540-30115-8_7 (cited on pages 6, 103)

Alvarez-Melis, D., & Fusi, N. (2020). Geometric dataset distances via optimal transport. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Editors), *Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*. (Cited on page 80).

Angiulli, F., & Pizzuti, C. (2002). Fast outlier detection in high dimensional spaces. *European Conference on Principles of Data Mining and Knowledge Discovery*, 15–27 (cited on pages 78, 81).

Arik, S. Ö., & Pfister, T. (2021). Tabnet: Attentive interpretable tabular learning. *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, 6679–6687. https://doi.org/10.1609/aaai.v35i8.16826 (cited on pages 32, 49)

Assran, M., Duval, Q., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M., LeCun, Y., & Ballas, N. (2023). Self-supervised learning from images with a joint-embedding predictive architecture. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15619–15629 (cited on page 23).

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. https://doi.org/10.48550/ARXIV.1607.06450. (Cited on page 33)

Bahri, D., Jiang, H., Tay, Y., & Metzler, D. (2022). Scarf: Self-supervised contrastive learning using random feature corruption. *International Conference on Learning Representations* (cited on page 31).

Baldi, P., & Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural Netw.*, *2*(1), 53–58. https://doi.org/10.1016/0893-6080(89)90014-2 (cited on page 22)

Barshan, E., Brunet, M.-E., & Dziugaite, G. K. (2020). Relatif: Identifying explanatory training samples via relative influence. In S. Chiappa & R. Calandra (Editors), *Proceedings of the twenty third international conference on artificial intelligence and statistics* (Pages 1899–1909). PMLR. (Cited on page 39).

Basu, S., You, X., & Feizi, S. (2020). On second-order group influence functions for black-box predictions. In H. D. III & A. Singh (Editors), *Proceedings of the 37th international conference on machine learning* (Pages 715–724). PMLR. (Cited on page 39).

Bauder, R. A., Khoshgoftaar, T. M., & Hasanin, T. (2018). An empirical study on class rarity in big data. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 785–790. https://doi.org/10.1109/ICMLA.2018.00125 (cited on page 76)

Bergman, L., & Hoshen, Y. (2020). Classification-based anomaly detection for general data. *International Conference on Learning Representations* (cited on pages 10, 16, 23–26, 36, 39, 44, 45, 48, 52, 61, 65, 78, 81, 82, V, XI).

Blanchard, G., Lee, G., & Scott, C. (2011). Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in neural information processing systems 24: Proceedings of the 2011 conference*. (Cited on page 8).

Borisov, V., Sessler, K., Leemann, T., Pawelczyk, M., & Kasneci, G. (2023). Language models are realistic tabular data generators. *The Eleventh International Conference on Learning Representations* (cited on page 14).

Bourdonnaye, F. D. L., & Daniel, F. (2021). Evaluating categorical encoding methods on a real credit card fraud detection database. *CoRR*, *abs/2112.12024* (cited on page 81).

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*, 123–140 (cited on page 15).

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). Lof: Identifying density-based local outliers. *SIGMOD Rec.*, *29*(2), 93–104. https://doi.org/10.1145/335191.335388 (cited on pages 51, 78)

Carvajal, K., Chacon, M., Mery, D., & Acuña, G. (2004). Neural network method for failure detection with skewed class distribution. *Insight - Non-Destructive Testing and Condition Monitoring*, *46*. https://doi.org/10.1784/insi.46.7.399.55578 (cited on pages 6, 103)

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953 (cited on page 13)

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, *6*(1), 1–6. https://doi.org/10.1145/1007730.1007733 (cited on page 15)

Chawla, N. V., Lazarevic, A., Hall, L. O., & Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In N. Lavrač, D. Gamberger, L. Todorovski, & H. Blockeel (Editors), *Knowledge discovery in databases: Pkdd 2003* (Pages 107–119). Springer Berlin Heidelberg. (Cited on page 15).

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785 (cited on pages 15, 29, 79)

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In H. D. III & A. Singh (Editors), *Proceedings of the 37th international conference on machine learning* (Pages 1597–1607). PMLR. (Cited on pages 23, 26, I).

Chen, X., & Konukoglu, E. (2018). Unsupervised detection of lesions in brain MRI using constrained adversarial auto-encoders. *Medical Imaging with Deep Learning* (cited on page 51).

Chen, X., & He, K. (2020). Exploring simple siamese representation learning. *CoRR*, *abs/2011.10566* (cited on page 23).

Chong, P., Ruff, L., Kloft, M., & Binder, A. (2020). Simple and Effective Prevention of Mode Collapse in Deep One-Class Classification. *2020 International Joint Conference on Neural Networks (IJCNN)*, 1–9. https://doi.org/10/gjmrtt (cited on pages 21, 38)
ZSCC: 0000003

Dai, E., & Chen, J. (2022). Graph-augmented normalizing flows for anomaly detection of multiple time series. *International Conference on Learning Representations* (cited on page 16).

Davis, J., & Goadrich, M. (2006). The relationship between precision-recall and roc curves. *Proceedings of the 23rd International Conference on Machine Learning*, 233–240. https://doi.org/10.1145/1143844.1143874 (cited on page 81)

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Editors), *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, NAACL-HLT 2019, minneapolis, mn, usa, june 2-7, 2019, volume 1 (long and short papers)* (Pages 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/n19-1423. (Cited on page 52)

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552* (cited on page 30).

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations* (cited on page 52).

Dutta, H., Giannella, C., Borne, K., & Kargupta, H. (2007). Distributed top-k outlier detection from astronomy catalogs using the demac system. *Proceedings of the 7th SIAM International Conference on Data Mining*. https://doi.org/10.1137/1.9781611972771.47 (cited on pages 20, 78)

Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, *20*(1), 18–36. https://doi.org/https://doi.org/10.1111/j.0824-7935.2004.t01-1-00228.x (cited on pages 6, 103)

Falkner, S., Klein, A., & Hutter, F. (2018). Bohb: Robust and efficient hyperparameter optimization at scale. *International conference on machine learning*, 1437–1446 (cited on page 30).

Finke, T., Krämer, M., Morandini, A., Mück, A., & Oleksiyuk, I. (2021). Autoencoders for unsupervised anomaly detection in high energy physics. *Journal of High Energy Physics*, *2021*(6). https://doi.org/10.1007/jhep06(2021)161 (cited on pages 22, 78)

Fix, E., & Hodges, J. L. (1989). Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, *57*(3), 238–247 (cited on page 3).

Frery, J. (2019). *Ensemble Learning for Extremely Imbalced Data Flows* (Theses 2019LYSES034). Université de Lyon. (Cited on pages 76, 79).

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139. https://doi.org/https://doi.org/10.1006/jcss.1997.1504 (cited on page 15)

Friedman, J. (2000). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*. https://doi.org/10.1214/aos/1013203451 (cited on page 88)

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), 463–484. https://doi.org/10.1109/TSMCC.2011.2161285 (cited on page 15)

Garima, Liu, F., Kale, S., & Sundararajan, M. (2020). Estimating training data influence by tracing gradient descent. *Proceedings of the 34th International Conference on Neural Information Processing Systems* (cited on pages 36, 39, 41).

Golan, I., & El-Yaniv, R. (2018). Deep anomaly detection using geometric transformations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on page 16).

Gong, D., Liu, L., Le, V., Saha, B., Mansour, M. R., Venkatesh, S., & Hengel, A. v. d. (2019a). Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (cited on page 22).

Gong, D., Liu, L., Le, V., Saha, B., Mansour, M. R., Venkatesh, S., & Hengel, A. v. d. (2019b). Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (cited on page 78).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on pages 10, 13, 14).

Gopalan, P., Sharan, V., & Wieder, U. (2019). Pidforest: Anomaly detection and certification via partial identification. *Neural Information Processing Systems* (cited on pages 19, 52, 78).

Gorishniy, Y., Rubachev, I., Kartashev, N., Shlenskii, D., Kotelnikov, A., & Babenko, A. (2023). Tabr: Tabular deep learning meets nearest neighbors in 2023. (Cited on pages 10, 49, 53, 55, 79).

Gorishniy, Y., Rubachev, I., Khrulkov, V., & Babenko, A. (2021). Revisiting deep learning models for tabular data. In A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Editors), *Advances in neural information processing systems*. (Cited on pages 32, 79).

Goyal, S., Raghunathan, A., Jain, M., Simhadri, H. V., & Jain, P. (2020). Drocc: Deep robust one-class classification. In H. D. III & A. Singh (Editors), *Proceedings of the 37th international conference on machine learning* (Pages 3711–3721). PMLR. (Cited on pages 21, 38, 51, V, X).

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., & Valko, M. (2020). Bootstrap your own latent: A new approach to self-supervised learning. (Cited on page 23).

Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track* (cited on pages 10, 53, 77, 79, 84).

Gu, S., Ślusarczyk, B., Hajizada, S., Kovalyova, I., & Sakhbieva, A. (2021). Impact of the covid-19 pandemic on online consumer purchasing behavior. *Journal of Theoretical and Applied Electronic Commerce Research*, *16*(6), 2263–2281. https://doi.org/10.3390/jtaer16060125 (cited on page 80)

Guha, S., Mishra, N., Roy, G., & Schrijvers, O. (2016). Robust random cut forest based anomaly detection on streams. *International Conference on Machine Learning* (cited on pages 19, 52, 78).

Guolin, K., Qi, M., Thomas, F., Taifeng, W., Wei, C., Weidong, M., Qiwei, Y., & Tie-Yan, L. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*, 3146–3154 (cited on pages 15, 29, 77, 79, 81, 82).

Hajiramezanali, E., Diamant, N. L., Scalia, G., & Shen, M. W. (2022). Stab: Self-supervised learning for tabular data. *NeurIPS 2022 First Table Representation Workshop* (cited on page 31).

Han, S., Hu, X., Huang, H., Jiang, M., & Zhao, Y. (2022). ADBench: Anomaly detection benchmark. *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track* (cited on pages 49, 61, 87).

Hariri, S., Kind, M. C., & Brunner, R. J. (2021). Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering*, *33*(4), 1479–1489. https://doi.org/10.1109/TKDE.2019.2947676 (cited on pages 19, 52, 78)

Hartigan, J. A., & Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, *28*(1), 100–108 (cited on page 32).

Hawkins, D. M. (1974). The detection of errors in multivariate data using principal components. *Journal of the American Statistical Association*, *69*(346), 340–344 (cited on pages 38, 51).

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969 (cited on page 15)

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284. https://doi.org/10.1109/TKDE.2008.239 (cited on page 13)

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (cited on page 23).

Hendrycks, D., Mazeika, M., Kadavath, S., & Song, D. (2019). Using self-supervised learning can improve model robustness and uncertainty. In *Proceedings of the 33rd international conference on neural information processing systems*. Curran Associates Inc. (Cited on page 23).

Hilal, W., Gadsden, S. A., & Yawney, J. (2022). Financial fraud: A review of anomaly detection techniques and recent advances. *Expert Systems with Applications*, *193*, 116429. https://doi.org/https://doi.org/10.1016/j.eswa.2021.116429 (cited on page 48)

Hollmann, N., Müller, S., Eggensperger, K., & Hutter, F. (2023). TabPFN: A transformer that solves small tabular classification problems in a second. *The Eleventh International Conference on Learning Representations* (cited on pages 29, 88).

Hu, S., Liang, Y., Ma, L., & He, Y. (2009). Msmote: Improving classification performance when training data is imbalanced. *2009 Second International Workshop on Computer Science and Engineering*, *2*, 13–17. https://doi.org/10.1109/WCSE.2009.756 (cited on page 14)

Huang, L., Nguyen, X., Garofalakis, M. N., Jordan, M. I., Joseph, A. D., & Taft, N. (2006). In-network PCA and anomaly detection. In B. Schölkopf, J. C. Platt, & T. Hofmann (Editors), *Advances in neural information processing systems 19, proceedings of the twentieth annual conference on neural information processing systems, vancouver, british columbia, canada, december 4-7, 2006* (Pages 617–624). MIT Press. (Cited on page 78).

Hui Han, B. M., Wenyuan Wang. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. *ICIC* (cited on page 14).

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intell. Data Anal.*, *6*(5), 429–449 (cited on pages 6, 76, 103).

Jeffares, A., Liu, T., Crabbé, J., Imrie, F., & van der Schaar, M. (2023). TANGOS: Regularizing tabular neural networks through gradient orthogonalization and specialization. *The Eleventh International Conference on Learning Representations* (cited on page 53).

Kadra, A., Lindauer, M., Hutter, F., & Grabocka, J. (2021). Well-tuned simple nets excel on tabular datasets. *Thirty-Fifth Conference on Neural Information Processing Systems* (cited on pages 30, 53, 79).

Kasmi, G., Dubus, L., Blanc, P., & Saint-Drenan, Y.-M. (2022). Towards unsupervised assessment with open-source data of the accuracy of deep learning-based distributed PV mapping. *Workshop on Machine Learning for Earth Observation (MACLEAN), in Conjunction with the ECML/PKDD 2022* (cited on page 3).

Kim, K. H., Shim, S., Lim, Y., Jeon, J., Choi, J., Kim, B., & Yoon, A. S. (2020). Rapp: Novelty detection with reconstruction along projection pathway. *International Conference on Learning Representations* (cited on pages 16, 22, 38, 48, 51, 78).

Kim, M. J., Grinsztajn, L., & Varoquaux, G. (2024). Carte: Pretraining and transfer for tabular learning. (Cited on page 88).

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. (Cited on pages 10, 14, 21, 36, 41, 42).

Koh, P. W., & Liang, P. (2017). Understanding black-box predictions via influence functions. In D. Precup & Y. W. Teh (Editors), *Proceedings of the 34th international conference on machine learning* (Pages 1885–1894). PMLR. (Cited on page 39).

Kong, Z., & Chaudhuri, K. (2021). Understanding instance-based interpretability of variational auto-encoders. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Editors), *Advances in neural information processing systems* (Pages 2400–2412). Curran Associates, Inc. (Cited on pages 36, 39, 43).

Kossen, J., Band, N., Lyle, C., Gomez, A., Rainforth, T., & Gal, Y. (2021). Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. In A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Editors), *Advances in neural information processing systems*. (Cited on pages 10, 11, 32–34, 49, 52, 53, 55, 58, 62, 104).

Kotelnikov, A., Baranchuk, D., Rubachev, I., & Babenko, A. (2023). TabDDPM: Modelling tabular data with diffusion models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett

(Editors), *Proceedings of the 40th international conference on machine learning* (Pages 17564–17579). PMLR. (Cited on page 14).

Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, *5*, 221–232 (cited on pages 6, 13, 14).

Krönert, E., Hattab, D., & Celisse, A. (2024). Breakpoint based online anomaly detection. (Cited on page 16).

Laurikkala, J., Juhola, M., & Kentala, E. (2000). Informal identification of outliers in medical data (cited on page 16).

Li, Z., Zhao, Y., Botta, N., Ionescu, C., & Hu, X. (2020). COPOD: Copula-based outlier detection. *2020 IEEE International Conference on Data Mining (ICDM)*. https://doi.org/10.1109/icdm50108.2020.00135 (cited on pages 17, 51, 81)

Li, Z., Zhao, Y., Hu, X., Botta, N., Ionescu, C., & Chen, G. H. (2022). ECOD: unsupervised outlier detection using empirical cumulative distribution functions. *CoRR*, *abs/2201.00382* (cited on pages 78, 81).

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. *2008 Eighth IEEE International Conference on Data Mining*, 413–422. https://doi.org/10.1109/ICDM.2008.17 (cited on pages 19, 48, 52, 78, 81)

Liu, T., Qian, Z., Berrevoets, J., & van der Schaar, M. (2023). GOGGLE: Generative modelling for tabular data by learning relational structure. *The Eleventh International Conference on Learning Representations* (cited on page 14).

Liu, X.-Y., Wu, J., & Zhou, Z.-H. (2008). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *39*(2), 539–550 (cited on page 15).

Liu, X.-Y., Wu, J., & Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *39*(2), 539–550. https://doi.org/10.1109/TSMCB.2008.2007853 (cited on page 13)

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (cited on page 34).

Liznerski, P., Ruff, L., Vandermeulen, R. A., Franks, B. J., Kloft, M., & Muller, K. R. (2021). Explainable deep one-class classification. *International Conference on Learning Representations* (cited on page 48).

Malaiya, R. K., Kwon, D., Kim, J., Suh, S. C., Kim, H., & Kim, I. (2018). An empirical evaluation of deep learning for network anomaly detection. *2018 International Conference on Computing, Networking and Communications (ICNC)*, 893–898. https://doi.org/10.1109/ICCNC.2018.8390278 (cited on page 48)

McInnes, L., Healy, J., & Melville, J. (2020). Umap: Uniform manifold approximation and projection for dimension reduction. (Cited on page 80).

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V., & Herrera, F. (2012). A unifying view on dataset shift in classification. *Pattern Recognition*, *45*(1), 521–530. https://doi.org/https://doi.org/10.1016/j.patcog.2011.06.019 (cited on page 7)

Moya, M. M., & Hush, D. R. (1996). Network constraints and multi-objective optimization for one-class classification. *Neural Netw.*, *9*(3), 463–474. https://doi.org/10.1016/0893-6080(95)00120-4 (cited on pages 10, 51)

Müller, S., Hollmann, N., Arango, S. P., Grabocka, J., & Hutter, F. (2022). Transformers can do bayesian inference. *International Conference on Learning Representations* (cited on page 29).

Parmar, N. J., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., & Tran, D. (2018). Image transformer. *International Conference on Machine Learning (ICML)* (cited on page 52).

Parzen, E. (1962). On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, *33*(3), 1065–1076. https://doi.org/10.1214/aoms/1177704472 (cited on pages 17, 51)

Pol, A. A., Berger, V., Cerminara, G., Germain, C., & Pierini, M. (2019). Anomaly Detection With Conditional Variational Autoencoders. *ICMLA 2019 - 18th IEEE International Conference on Machine Learning and Applications* (cited on pages 22, 78).

Pozzolo, A. D., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. *2015 IEEE Symposium Series on Computational Intelligence*, 159–166. https://doi.org/10.1109/SSCI.2015.33 (cited on pages 11, 84)

Principi, E., Vesperini, F., Squartini, S., & Piazza, F. (2017). Acoustic novelty detection with adversarial autoencoders. *2017 International Joint Conference on Neural Networks (IJCNN)*, 3324–3330. https://doi.org/10.1109/IJCNN.2017.7966273 (cited on page 51)

Qiu, C., Pfrommer, T., Kloft, M., Mandt, S., & Rudolph, M. (2021). Neural transformation learning for deep anomaly detection beyond images. In M. Meila & T. Zhang (Editors), *Proceedings of the 38th international conference on machine learning, ICML 2021, 18-24 july 2021, virtual event* (Pages 8703–8714). PMLR. (Cited on pages 10, 16, 23, 25, 36, 39, 45, 48, 52, 54, 65, 78, 81, 82, V, XII).

Quiñonero-Candela, J. (Editor). (2009). *Dataset shift in machine learning*. MIT Press. (Cited on page 7) OCLC: ocn227205909.

Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In W. Chen, J. F. Naughton, & P. A. Bernstein (Editors), *Proceedings of the 2000 ACM SIGMOD international conference on management of data, may 16-18, 2000, dallas, texas, USA* (Pages 427–438). ACM. https://doi.org/10.1145/342009.335437. (Cited on pages 51, 78, 81)

Reiss, T., Cohen, N., Bergman, L., & Hoshen, Y. (2021). Panda: Adapting pretrained features for anomaly detection and segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2806–2814 (cited on pages 23, 26).

Reiss, T., & Hoshen, Y. (2023). Mean-shifted contrastive loss for anomaly detection. *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*. https://doi.org/10.1609/aaai.v37i2.25309 (cited on pages 23, 27, 28, 52)

Reyes, E., & Estévez, P. A. (2020). Transformation based deep anomaly detection in astronomical images. *2020 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN48605.2020.9206997 (cited on page 48)

Roberts, S., & Tarassenko, L. (1994a). A probabilistic resource allocating network for novelty detection. *Neural Computation*, *6*(2), 270–284. https://doi.org/10.1162/neco.1994.6.2.270 (cited on page 17)

Roberts, S., & Tarassenko, L. (1994b). A probabilistic resource allocating network for novelty detection. *Neural Computation*, *6*(2), 270–284. https://doi.org/10.1162/neco.1994.6.2.270 (cited on page 51)

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10684–10695 (cited on page 3).

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519 (cited on page 3)

Roy, T. S., & Vijay, A. H. (2020). A robust anomaly finder based on autoencoders. (Cited on page 22).

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., Müller, E., & Kloft, M. (2018). Deep one-class classification. *Proceedings of the 35th International Conference on Machine Learning*, *80*, 4393–4402 (cited on pages 10, 20, 21, 27, 36, 38, 45, 48, 51, 78).

Ruff, L., Vandermeulen, R. A., Görnitz, N., Binder, A., Müller, E., Müller, K.-R., & Kloft, M. (2020). Deep semi-supervised anomaly detection. *International Conference on Learning Representations* (cited on pages 10, 18, 21, 38, 78).

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., & Müller, K.-R. (2021). A Unifying Review of Deep and Shallow Anomaly Detection. *Proceedings*

of the IEEE, *109*(5), 756–795. https://doi.org/10/gjmk3g (cited on pages 17, 20, 35)
ZSCC: 0000035

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, *5*, 197–227 (cited on page 15).

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., & Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In M. Niethammer, M. Styner, S. Aylward, H. Zhu, I. Oguz, P.-T. Yap, & D. Shen (Editors), *Information processing in medical imaging* (Pages 146–157). Springer International Publishing. (Cited on pages 10, 16, 51).

Schölkopf, B., Platt, J., & Hofmann, T. (2007). In-network pca and anomaly detection. In *Advances in neural information processing systems 19: Proceedings of the 2006 conference* (Pages 617–624). (Cited on page 20).

Schölkopf, B., Williamson, R., Smola, A., Shawe-Taylor, J., & Platt, J. (1999). Support vector method for novelty detection. *Proceedings of the 12th International Conference on Neural Information Processing Systems*, 582–588 (cited on pages 10, 19, 38, 48, 51, 78).

Sehwag, V., Chiang, M., & Mittal, P. (2021). SSD: A unified framework for self-supervised outlier detection. *International Conference on Learning Representations* (cited on page 23).

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Napolitano, A. (2009). Rusboost: A hybrid approach to alleviating class imbalance. *IEEE transactions on systems, man, and cybernetics-part A: systems and humans*, *40*(1), 185–197 (cited on page 15).

Sendera, M., Śmieja, M., Maziarka, Ł., Struski, Ł., Spurek, P., & Tabor, J. (2021). Flow-based SVDD for anomaly detection. *arXiv:2108.04907 [cs]* (cited on page 21).

Sharan, V., Gopalan, P., & Wieder, U. (2018). Efficient anomaly detection via matrix sketching. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on page 38).

Shavitt, I., & Segal, E. (2018). Regularization learning networks: Deep learning for tabular datasets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on pages 29, 30, 49, 53).

Shen, L., Li, Z., & Kwok, J. (2020). Timeseries anomaly detection using temporal hierarchical one-class network. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Editors), *Advances in neural information processing systems* (Pages 13016–13026). Curran Associates, Inc. (Cited on page 16).

Shenkar, T., & Wolf, L. (2022). Anomaly detection for tabular data with internal contrastive learning. *International Conference on Learning Representations* (cited on pages 10, 16, 26, 36, 39, 45, 48, 52, 54, 61, 64, 65, 79, 81, 82).

Shwartz-Ziv, R., & Armon, A. (2021). Tabular data: Deep learning is not all you need. *8th ICML Workshop on Automated Machine Learning (AutoML)* (cited on pages 10, 53).

Silva, J. V. V., Lopez, M. A., & Mattos, D. M. F. (2020). Attackers are not stealthy: Statistical analysis of the well-known and infamous kdd network security dataset. *2020 4th Conference on Cloud and Internet of Things (CIoT)*, 1–8. https://doi.org/10.1109/CIoT50422.2020.9244289 (cited on page 61)

Sklar, M. (1959). Fonctions de repartition an dimensions et leurs marges. *Publ. inst. statist. univ. Paris*, *8*, 229–231 (cited on page 17).

Sohn, K., Li, C.-L., Yoon, J., Jin, M., & Pfister, T. (2021). Learning and evaluating representations for deep one-class classification. *International Conference on Learning Representations* (cited on pages 23, 24, 27, 39, 52).

Somepalli, G., Goldblum, M., Schwarzschild, A., Bruss, C. B., & Goldstein, T. (2021). SAINT: improved neural networks for tabular data via row attention and contrastive pre-training. *CoRR*, *abs/2106.01342* (cited on pages 10, 30, 34, 49, 52, 53, 79).

Tack, J., Mo, S., Jeong, J., & Shin, J. (2020). Csi: Novelty detection via contrastive learning on distributionally shifted instances. *NeurIPS* (cited on pages 23, 26, 78).

Tax, D., & Duin, R. (2004). Support vector data description. *Machine Learning*, *54*, 45–66. https://doi.org/10.1023/B:MACH.0000008084.60811.49 (cited on pages 10, 18, 36, 38, 48, 51, 78)

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024a). Beyond individual input for deep anomaly detection on tabular data. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, & F. Berkenkamp (Editors), *Proceedings of the 41st international conference on machine learning* (Pages 48097–48123). PMLR. (Cited on pages 78, 81, 82).

Thimonier, H., Popineau, F., Rimmel, A., & Doan, B.-L. (2024b). Retrieval augmented deep anomaly detection for tabular data. *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA*. https://doi.org/https://doi.org/10.1145/3627673.3679559

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2022). TracInAD: Measuring influence for anomaly detection. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–6. https://doi.org/10.1109/IJCNN55064.2022.9892058 (cited on pages 48, 78)

Thimonier, H., Popineau, F., Rimmel, A., Doan, B.-L., & Daniel, F. (2023). Comparative evaluation of anomaly detection methods for fraud detection in online credit card payments.

Tian, Y., Krishnan, D., & Isola, P. (2019). Contrastive multiview coding. *CoRR*, *abs/1906.05849* (cited on page 23).

Tian, Y., Chen, X., & Ganguli, S. (2021). Understanding self-supervised learning dynamics without contrastive pairs. *CoRR*, *abs/2102.06810* (cited on page 23).

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., … Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. (Cited on page 3).

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605 (cited on page 80).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on pages 11, 32, 49, 52, 65).

Verma, V., Luong, T., Kawaguchi, K., Pham, H., & Le, Q. V. (2021). Towards domain-agnostic contrastive learning. *ICML*, 10530–10541 (cited on page I).

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, *11*(110), 3371–3408 (cited on page 21).

Wang, T., & Isola, P. (2020). Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *ICML* (cited on pages 23, 24).

Wei, W., Li, J., Cao, L., Ou, Y., & Chen, J. (2013). Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, *16*. https://doi.org/10.1007/s11280-012-0178-0 (cited on page 6)

Weiss, G., & Provost, F. (2001). The effect of class distribution on classifier learning: An empirical study. *Tech Rep* (cited on pages 6, 103).

Wu, G., & Chang, E. Y. (2003). Class-boundary alignment for imbalanced dataset learning. *In ICML 2003 Workshop on Learning from Imbalanced Data Sets*, 49–56 (cited on pages 6, 103).

Wu, J., Chen, S., Zhao, Q., Sergazinov, R., Li, C., Liu, S., Zhao, C., Xie, T., Guo, H., Ji, C., Cociorva, D., & Brunzel, H. (2024). Switchtab: Switched autoencoders are effective tabular learners. (Cited on page 32).

Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019). Modeling tabular data using conditional gan. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Gar-

nett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on page 14).

Yan, J., Zheng, B., Xu, H., Zhu, Y., Chen, D., Sun, J., Wu, J., & Chen, J. (2024). Making pre-trained language models great on tabular prediction. *The Twelfth International Conference on Learning Representations* (cited on page 34).

Yang, Y., Wang, Y., Liu, G., Wu, L., & Liu, Q. (2023). Unitabe: Pretraining a unified tabular encoder for heterogeneous tabular data. *arXiv preprint arXiv:2307.09249* (cited on page 32).

Yanmin, S., Wong, A., & Kamel, M. S. (2011). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*, 687–719. https://doi.org/10.1142/S0218001409007326 (cited on pages 6, 48, 76, 103)

Ye, H., Fan, W., Song, X., Zheng, S., Zhao, H., dan Guo, D., & Chang, Y. (2024). PTaRL: Prototype-based tabular representation learning via space calibration. *The Twelfth International Conference on Learning Representations* (cited on page 32).

Yeh, C.-K., Kim, J. S., Yen, I. E., & Ravikumar, P. (2018). Representer point selection for explaining deep neural networks. *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 9311–9321 (cited on page 39).

You, C., Zhao, R., Liu, F., Dong, S., Chinchali, S., Topcu, U., Staib, L., & Duncan, J. (2022). Class-aware adversarial transformers for medical image segmentation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Editors), *Advances in neural information processing systems* (Pages 29582–29596). Curran Associates, Inc. (Cited on page 3).

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., & Hsieh, C.-J. (2020). Large batch optimization for deep learning: Training bert in 76 minutes. *International Conference on Learning Representations* (cited on pages 61, 64).

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. *Proceedings of the IEEE/CVF international conference on computer vision*, 6023–6032 (cited on page 30).

Zhang, H., Zhang, J., Shen, Z., Srinivasan, B., Qin, X., Faloutsos, C., Rangwala, H., & Karypis, G. (2024). Mixed-type tabular data synthesis with score-based diffusion in latent space. *The Twelfth International Conference on Learning Representations* (cited on page 14).

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). Mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412* (cited on page 30).

Zhang, J., & Mani, I. (2003). KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets* (cited on pages 6, 13, 103).

Zhang, M., Lucas, J., Ba, J., & Hinton, G. E. (2019). Lookahead optimizer: K steps forward, 1 step back. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Editors), *Advances in neural information processing systems*. Curran Associates, Inc. (Cited on page 61).

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. *CVPR* (cited on page 26).

Zhao, Y., Nasrullah, Z., & Li, Z. (2019). Pyod: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, *20*(96), 1–7 (cited on page 82).

Zhao, Z., Kunar, A., Birke, R., & Chen, L. Y. (2021). Ctab-gan: Effective table data synthesizing. In V. N. Balasubramanian & I. Tsang (Editors), *Proceedings of the 13th asian conference on machine learning* (Pages 97–112). PMLR. (Cited on page 14).

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., & Loy, C. C. (2022). Domain Generalization: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–20. https://doi.org/10.1109/TPAMI.2022.3195549 (cited on page 8)

Zhu, B., Shi, X., Erickson, N., Li, M., Karypis, G., & Shoaran, M. (2023). XTab: Cross-table pretraining for tabular transformers. *Proceedings of the 40th International Conference on Machine Learning* (cited on page 32).

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., & Chen, H. (2018). Deep autoencoding gaussian mixture model for unsupervised anomaly detection. *International Conference on Learning Representations* (cited on pages 43, 44, 61).

Résumé

L'utilisation de l'apprentissage machine pour la détection de fraudes au paiement par carte bancaire est cruciale du point de vue des institutions bancaires et financières. En effet, selon le rapport Nilson[1], on estime le coût annuel des fraudes à 28.58 milliard de dollars en 2021 ce qui représente approximativement le PIB de pays comme la Guinée Equatoriale,le Bénin ou encore le Rwanda.

Si les institutions bancaires se sont jusqu'alors appuyées sur des modèles déterministes reposant sur des systèmes de règles pour détecter des fraudes, l'essor relativement récent des méthodes d'apprentissage statistiques sur un large panel d'applications a poussé ces institutions à se tourner vers ce type d'approche pour renforcer leurs systèmes de détection.

L'utilisation de l'apprentissage statistique pour la détection de fraude n'est néanmoins pas triviale. Deux caractéristiques de cette application rendent les approches standards difficilement applicables. D'une part, la base de données d'entrainement est extrêmement déséquilibrées: la part de paiements non frauduleux est largement majoritaire ($> 99\%$) par rapport à la part de fraudes. Par ailleurs, les paiements par carte bancaire sont particulièrement sujets au changement de distribution au cours du temps: les comportements de paiement d'une période ne sont pas ceux d'une autre. En témoignent notamment les changements d'habitude de paiements engendrés par les confinements successifs liés à la pandémie de Covid-19. Par un effet d'hystérèse, bien que la pandémie soit derrière nous, les changements des habitudes de paiements se sont maintenues.

Ainsi, comme la littérature l'a montré (Akbani et al., 2004; Carvajal et al., 2004; Estabrooks et al., 2004; Japkowicz & Stephen, 2002; Weiss & Provost, 2001; Wu & Chang, 2003; Yanmin et al., 2011; Zhang & Mani, 2003) les méthodes standards supervisées peinent à obtenir des performances satisfaisantes sur des tâches de classification et de régression dans le cas d'un tel déséquilibre. De même, en case de *distribution shift*, la plupart des propriétés statistiques des méthodes d'apprentissage statistiques ne sont plus satisfaites. Dès lors, une classe d'algorithme supervisé semble particulièrement pertinente pour contrecarrer ces problèmes: les méthodes de détection d'anomalies.

Les méthodes de détection d'anomalies semi-supervisées diffèrent des méthodes classiques supervisées en ce qu'elles utilisent le label de manière indirecte. Le label est utilisé pour construire un jeu de données composé uniquement d'observations appartenant à une seule classe dite *normale*. Ces méthodes caractérisent alors la distribution *normale* et considèrent les samples non-normaux comme des anomalies. Ce faisant, le déséquilibre n'est plus un problème puisque seule la classe majoritaire sert à l'apprentissage. De plus, les nouveaux types d'anomalies pouvant apparaître devraient être détectés puisque n'appartenant pas aux samples normaux.

Dans ce cadre là, nous avons investigué la question des méthodes de détection d'anomalies pour données structurées en proposant deux nouveaux algorithmes.

[1] https://nilsonreport.com/

Le premier, `TracInAD` repose sur une mesure d'influence utilisée originellement dans la littérature de l'explicabilité. `TracIn` sert en effet à mesurer la contribution d'une observation $\mathbf{x}$ à la diminution de la fonction de perte $\ell(.,.)$ d'une autre observation $\mathbf{z}$ au cours de l'entraînement. Notre approche consiste à entrainer un modèle de détection d'anomalie à base de réseau neuronaux et à mesurer l'influence moyenne d'un échantillon de la base d'entrainement sur chaque sample de la base de test. Notre approche est agnostique au modèle de détection d'anomalie utilisé, et démontre des résultats encourageants en comparaison des méthodes existantes. Cette approche permet de mesurer à quel point une observation est une anomalie via des dépendances inter-observations, c'est à dire via les relations qu'elle a avec les autres samples.

L'autre algorithme proposé permet quant à lui de combiner les dépendances inter-observations précédemment évoquées, avec les dépendances inter-features de chacune de ces observations. Cette approche consiste à entrainer un transformer augmenté d'un module de *retrieval*. Ce dernier permet de sélectionner des observations au sein de la base d'entrainement, permettant pour *chacune* des observations de réduire sa fonction de perte. Les modules de retrieval peuvent être *externes* au transformer et reposer sur des modules d'attention ou bien à base de k plus proches voisins, ou bien être *internes* au modèle via l'utilisation de transformer non paramétriques (Kossen et al., 2021). Cette approche à base de reconstruction de masques, consiste à entrainer un transformer à prédire les valeurs des features masquées d'observations normales uniquement. En inférence, on demandera à ces modèles entrainés de prédire les features masquées des observations de test pour un ensemble de masques: plus l'erreur de reconstruction sera importante, plus la probabilité de n'avoir pas été généré par la distribution *normale* sera importante. Augmenter les transformers de module permettant de recourir à des dépendances inter-samples a permis d'augmenter significativement les performances de détection d'anomalie, et semble montrer qu'inclure les deux types de dépendances sont clefs pour détecter efficacement tous les types d'anomalies.

Enfin, nous avons tester différentes méthodes de détection d'anomalie sur une dataset de paiements par carte bancaire mis à disposition par une grande banque française. Nous avons comparé les performances obtenues par les différents aux performances de l'approche supervisée classiquement utilisée sur les données tabulaire: les *gradient boosted decisions trees*. Nos expériences ont montré une différence de performance significative en faveur des gradient boosted decisions trees qui surpassent toutes les méthodes de détection d'anomalie testées d'une marge importante. Ainsi, ces résultats semblent indiquer que les méthodes de détection d'anomalie semblent encore peu compétitives en comparaison des gradient boosted decisions trees.

Summary

Using Machine Learning (ML) models for fraud detection on credit card payments is of critical importance for banks and financial institutions. Until recently, they had relied on deterministic rule-based models constructed thanks to expert knowledge. However, with the recent success of ML in a broad range of applications, banks have turned to this type of approach to augment their existing detection systems.

However, using ML for fraud detection is far from trivial. Indeed, two characteristics of this application make using standard supervised methods challenging. First, the datasets are inherently imbalanced as most payments are non-fraudulent ($> 99\%$). Second, payment behaviors tend to change over time and generate a distribution shift: the data used to train a model may be generated from a different distribution than the test set.

One class of weakly supervised algorithms appears particularly relevant to overcoming these two problems: weakly supervised anomaly detection (AD) methods.

Weakly-supervised AD methods differ from standard supervised methods as they *indirecty* use the labels to construct a dataset solely composed of one class, considered the *normal* class. These methods characterize the *normal* distribution while considering low-probable samples under this distribution as *anomalies*. Thus, the imbalance does not negatively impact these approaches; moreover, as anomalies are defined in opposition to normal samples, the distribution shift of fraudulent payment should not have any impact on the models' performance.

Thus, we investigated using AD methods for structured data and fraud detection by proposing two novel algorithms.

First, we put forward `TracInAD`, which relies on an influence measure originally proposed for explicability purposes. `TracIn` serves to measure how much a sample x contributed to reducing the loss of another sample z in the course of training. Our approach consists in training a Deep AD model and measuring the average influence of a subsample of the training set on each sample of the test set. This approach relies on inter-sample dependencies to identify anomalies.

Second, we put forward an approach combining inter-feature and inter-sample dependencies to identify frauds within tabular datasets. This approach consists in training a transformer augmented by a retrieval module, for the self-supervised task of mask reconstruction. In inference, the model's ability to reconstruct a sample's masked features serves to measure how likely the sample is to be an anomaly. Our experiments suggest that including both dependencies is critical to efficiently detecting anomalies in tabular data.

Finally, we experimented with a private real-life online payment dataset made available by a prominent French bank. We compared the performance of AD methods to Gradient Boosted Decision Trees (GBDT) and found that GBDT significantly outperformed all tested AD methods. Our

experiments suggest that AD methods need further improvement to compete with GBDT for fraud detection.

Appendix A

Self-Supervised Learning

A.1 Contrastive Learning

To compute the contrastive loss, let $x \in \mathbb{R}^d$ be a sample while $\{x_+\}$ and $\{x_-\}$ are a set of positive and negative samples respectively. Then, the contrastive loss can be expressed as

$$\mathscr{L}_{cont}(\mathbf{x}, \{\mathbf{x}_+\}, \{\mathbf{x}_-\}) = \frac{1}{|\{\mathbf{x}_+\}|} \log \frac{\sum_{\mathbf{x}' \in \{\mathbf{x}_+\}} \exp(sim(\phi(\mathbf{x}), \phi(\mathbf{x}'))/\tau)}{\sum_{\mathbf{x}' \in \{\mathbf{x}_+\} \cup \{\mathbf{x}_-\}} \exp(sim(\phi(\mathbf{x}), \phi(\mathbf{x}'))/\tau)} \tag{1}$$

where $sim(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y} / \|\mathbf{x}\| \|\mathbf{y}\|$ is the cosine similarity, $|\{\mathbf{x}_+\}|$ is the cardinality of the set $\{\mathbf{x}_+\}$, and $\phi(\mathbf{x}) = f(\mathbf{x})/\|f(\mathbf{x})\|_2$ is the ℓ_2-normalized constrastive representation of sample $\mathbf{x}$.

Both contrastive and non-contrastive self-supervised learning involve using transformation as a mean of creating positive pairs (as well as negative pairs for the case of contrastive learning). The type of transformations to apply for self-supervised learning to yield meaningful data representation has been well-studied for the case of image datasets. For instance, Chen et al. (2020) have discussed how random crops, blurring or color distortion should be the preferred transformation to apply to create from the original sample $\mathbf{x} \in \mathscr{X}$ the positive pair $(\mathbf{x}_1, \mathbf{x}_2)$ while other transformations such as four-fold rotations could be harmful to contrastive learning. Regarding other types of data, especially tabular datasets, the question of data augmentation is unclear. Recently, Verma et al. (2021) have proposed an augmentation methodology that makes contrastive learning possible for any data domain $\mathscr{X}$ without any specific knowledge required. They propose to use mixup as a mean of combining existing samples to obtain the augmented versions of the original. Their approach makes use of hidden representations of data samples x instead of directly modifying $\mathbf{x}$. For instance consider an encoder $f : \mathscr{X} \to \mathscr{Z}$, and $h, \tilde{h}$ fixed-sized hidden representations from an intermediate layer of f for samples $\mathbf{x}, \tilde{\mathbf{x}} \in \mathscr{X}$. Then positive samples for $\mathbf{x}$ can be constructed as

$$\hat{h} = \lambda h + (1 - \lambda)\tilde{h},$$

with $\lambda \sim \mathrm{U}(\alpha, 1)$ and α close to 1.

A.2 SimCLR

Let us briefly discuss how SimCLR functions, consider for each sample $\mathbf{x}_i$ two transformation $\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}$, where $\mathbf{x}_i^{(1)} := \mathrm{T}_1(\mathbf{x})$ where $\mathrm{T} \in \mathscr{T}$ a family of transformations. Then, the SimCLR objective can simply

be described as a contrastive objective as in equation (1), where the pairs $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$ and $\mathbf{x}_i^{(2)}, \mathbf{x}_i^{(1)}$ are seen as positive examples while any other examples are negative. For instance, consider a batch $\mathscr{B} := \{\mathbf{x}_i\}_{i=1}^{\mathrm{B}}$, then denoting $\tilde{\mathscr{B}} = \{\mathbf{x}_i^{(1)}\}_{i=1}^{\mathrm{B}} \bigcup \{\mathbf{x}_i^{(2)}\}_{i=1}^{\mathrm{B}}$ and $\tilde{\mathscr{B}_{-i}} = \{\mathbf{x}_j^{(1)}\}_{j \neq i} \bigcup \{\mathbf{x}_j^{(2)}\}_{j \neq i}$, then the SimCLR loss can be expressed using the contrastive loss described in equation (1) as

$$\mathscr{L}_{\mathrm{SimCLR}}(\mathscr{B}; \mathscr{T}) := \frac{1}{2\mathrm{B}} \sum_{i=1}^{\mathrm{B}} \mathscr{L}_{cont}(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, \tilde{\mathscr{B}_{-i}}) + \mathscr{L}_{cont}(\mathbf{x}_i^{(2)}, \mathbf{x}_i^{(1)}, \tilde{\mathscr{B}_{-i}}) \tag{2}$$

Appendix B

Detailed Experimental Results

B.1 Experimental settings

Table B.1: Datasets hyperparameters. When the batch size is -1 it refers to a full pass over the training set before an update of the weights.

Dataset	epoch	batch size	lr	p_{mask}^{train}	r	m	e
Wine	1000	−1	0.001	0.15	1	13	8
Lympho	100	−1	0.01	0.15	4	3078	16
Glass	1000	−1	0.01	0.15	4	255	16
Vertebral	2000	−1	0.001	0.15	1	6	8
WBC	100	−1	0.01	0.15	3	4525	16
Ecoli	100	−1	0.01	0.15	3	63	16
Ionosphere	100	−1	0.001	0.15	2	561	16
Arrhythmia	100	−1	0.01	0.15	1	274	16
BreastW	500	−1	0.01	0.15	3	129	16
Pima	500	−1	0.01	0.15	4	162	16
Vowels	1000	−1	0.01	0.15	2	78	16
Letter Recognition	1000	−1	0.01	0.15	1	32	16
Cardio	100	−1	0.01	0.15	2	231	16
Seismic	100	−1	0.01	0.15	2	276	16
Musk	100	−1	0.01	0.15	2	166	16
Speech	1000	512	0.001	0.15	1	400	8
Thyroid	5000	−1	0.01	0.1	2	21	16
Abalone	1000	−1	0.0001	0.15	4	162	16
Optdigits	500	−1	0.01	0.2	1	64	16
Satimage-2	100	−1	0.01	0.2	1	36	16
Satellite	100	−1	0.01	0.2	1	36	16
Pendigits	1000	−1	0.01	0.25	2	136	16
Annthyroid	400	−1	0.01	0.15	1	6	16
Mnist	1000	−1	0.001	0.15	1	100	32
Mammography	200	−1	0.01	0.25	4	56	16
Shuttle	100	4096	0.01	0.25	3	129	64
Mulcross	100	4096	0.001	0.15	2	10	16
ForestCover	100	4096	0.01	0.15	2	55	16
Campaign	100	4096	0.001	0.15	1	62	16
Fraud	100	4096	0.001	0.2	1	29	32
Backdoor	1000	256	0.001	0.2	1	196	32

B.2 Additional results

In this section, we display the metrics for each of the experiments we performed. This includes the F1-score for all tested approaches in tables B.2 and B.3, the AUROC for the approaches for which it is relevant to compute it; displayed in tables B.4 and B.5, the F1-score for each architecture discussed in the original papers of NeuTraL-AD Qiu et al. (2021) in table B.8, GOAD Bergman and Hoshen (2020) in table B.7, and DROCC Goyal et al. (2020) in table B.6. For each of these tables, we highlight in bold the highest metric in the table.

Table B.2: Anomaly detection F1-score (↑) for deep models. We perform 5% T-test to test whether the difference between the highest metrics for each dataset is statistically significant.

Method	DROCC (abalone)	GOAD (thyroid)	NeuTraL. (arrhy.)	Inter.Cont.	Transformer	NPT-AD
Wine	63.0±20.0	67.0±9.4	78.2±4.5	90.0±6.3	23.5±7.9	72.5±7.7
Lympho	65.0±5.0	68.3±13.0	20.0±18.7	86.7±6.0	88.3±7.6	**94.2**±7.9
Glass	14.5±11.1	12.7±3.9	9.0±4.4	**27.2**±10.6	14.4±6.1	**26.2**±10.9
Vertebral	9.3±6.1	16.3±9.6	3.8±1.2	**26.0**±7.7	12.3±5.2	20.3±4.8
Wbc	9.0±6.2	66.2±2.9	60.9±5.6	**67.6**±3.6	66.4±3.2	**67.3**±1.7
Ecoli	N/A	61.4±31.7	7.0±7.1	70.0±7.8	75.0±9.9	**77.7**±0.1
Ionosphere	76.9±2.8	83.4±2.6	90.6±2.4	**93.2**±1.3	88.1±2.8	**92.7**±0.6
Arrhyth.	37.1±6.8	52.0±2.3	59.5±2.6	**61.8**±1.8	59.8±2.2	60.4±1.4
Breastw	93.0±3.7	96.0±0.6	91.8±1.3	96.1±0.7	**96.7**±0.3	95.7±0.3
Pima	66.0±4.1	66.0±3.1	60.3±1.4	59.1±2.2	65.6±2.0	**68.8**±0.6
Vowels	66.2±8.8	31.1±4.2	10.0±6.2	**90.8**±1.6	28.7±8.0	88.7±1.6
Letter	55.6±3.6	20.7±1.7	5.7±0.8	62.8±2.4	41.5±6.2	**71.4**±1.9
Cardio	49.8±3.2	**78.6**±2.5	45.5±4.3	71.0±2.4	68.8±2.8	**78.1**±0.1
Seismic	19.1±0.9	24.1±1.0	11.8±4.3	20.7±1.9	19.1±5.7	**26.2**±0.7
Musk	99.4±1.5	**100**±0.0	99.0±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Speech	4.3±2.0	4.8±2.3	4.7±1.4	5.2±1.2	6.8±1.9	**9.3**±0.8
Thyroid	72.7±3.1	72.5±2.8	69.4±1.4	**76.8**±1.2	55.5±4.8	**77.0**±0.6
Abalone	17.9±1.3	57.6±2.2	53.2±4.0	**68.7**±2.3	42.5±7.8	59.7±0.1
Optdig.	30.5±5.2	0.3±0.3	16.2±7.3	**66.3**±10.1	61.1±4.7	62.0±2.7
Satim.	4.8±1.6	90.7±0.7	92.3±1.9	92.4±0.7	89.0±4.1	**94.8**±0.8
Satellite	52.2±1.5	64.2±0.8	71.6±0.6	73.2±1.6	65.6±3.3	**74.6**±0.7
Pendig.	11.0±2.6	40.1±5.0	69.8±8.7	82.3±4.5	35.4±10.9	**92.5**±1.3
Annthyr.	64.2±3.3	50.3±6.3	44.1±2.3	45.4±1.8	29.9±1.5	**57.7**±0.6
Mnist	N/A	66.9±1.3	84.8±0.5	85.9±0.0	56.7±5.7	**71.8**±0.3
Mammo.	32.6±2.1	33.7±6.1	19.2±2.4	29.4±1.4	17.4±2.2	**43.6**±0.5
Shuttle	N/A	73.5±5.1	97.9±0.2	**98.4**±0.1	85.3±9.8	**98.2**±0.3
Mullcr.	N/A	99.7±0.8	96.3±10.5	**100**±0.0	**100**±0.0	**100**±0.0
Forest	N/A	0.1±0.2	51.6±8.2	44.0±4.1	21.3±3.1	58.0±10
Camp.	N/A	16.2±1.8	42.1±1.7	46.8±1.4	47.0±1.9	**49.8**±0.3
Fraud	N/A	53.1±10.2	24.3±7.8	**57.9**±2.8	54.3±5.2	**58.1**±3.2
Backd.	N/A	12.7±2.9	84.4±1.8	**86.6**±0.1	85.8±0.6	84.1±0.1
mean	32.7	51.0	50.8	67.2	56.2	**68.8**
mean std	3.4	4.4	4.0	2.9	4.3	2.0
mean rk	11.5	8.4	9.6	3.7	7.5	**3.1**

Table B.3: Anomaly detection F1-score (↑) for non-deep models. We perform 5% T-test to test whether the difference between the highest metrics for each dataset is statistically significant.

Method	COPOD	IForest	KNN	PIDForest	RRCF	NPT-AD
Wine	60.0±4.5	64.0±12.8	**94.0**±4.9	50.0±6.4	69.0±11.4	**72.5**±7.7
Lympho	85.0±5.0	71.7±7.6	80.0±11.7	70.0±0.0	36.7±18.0	**94.2**±7.9
Glass	11.1±0.0	11.1±0.0	11.1±9.7	8.9±6.0	15.6±13.3	**26.2**±10.9
Vertebral	1.7±1.7	13.0±3.8	10.0±4.5	12.0±5.2	8.0±4.8	**20.3**±4.8
Wbc	**71.4**±0.0	70.0±3.7	63.8±2.3	65.7±3.7	54.8±6.1	67.3±1.7
Ecoli	25.6±11.2	58.9±22.2	**77.8**±3.3	25.6±11.2	28.9±11.3	**77.7**±0.1
Ionosphere	70.8±1.8	80.8±2.1	88.6±1.6	67.1±3.9	72.0±1.8	**92.7**±0.6
Arrhythmia	58.2±1.4	60.9±3.3	**61.8**±2.2	22.7±2.5	50.6±3.3	60.4±1.4
Breastw	96.4±0.6	**97.2**±0.5	96.0±0.7	70.6±7.6	63.0±1.8	95.7±0.3
Pima	62.3±1.1	**69.6**±1.2	65.3±1.0	65.9±2.9	55.4±1.7	**68.8**±0.6
Vowels	4.8±1.0	25.8±4.7	64.4±3.7	23.2±3.2	18.0±4.6	**88.7**±1.6
Letter	12.9±0.7	15.6±3.3	45.0±2.6	14.2±2.3	17.4±2.2	**71.4**±1.9
Cardio	65.0±1.4	73.5±4.1	67.6±0.9	43.0±2.5	43.9±2.7	**78.1**±0.1
Seismic	29.2±1.3	**73.9**±1.5	30.6±1.4	29.2±1.6	24.1±3.2	26.2±0.7
Musk	49.6±1.2	52.0±15.3	**100**±0.0	35.4±0.0	38.4±6.5	**100**±0.0
Speech	3.3±0.0	4.9±1.9	5.1±1.0	2.0±1.9	3.9±2.8	**9.3**±0.8
Thyroid	30.8±0.5	**78.9**±2.7	57.3±1.3	72.0±3.2	31.9±4.7	77.0±0.6
Abalone	50.3±6.4	53.4±1.7	43.4±4.8	58.6±1.6	36.9±6.4	**59.7**±0.1
Optdigit	3.0±0.3	15.8±4.3	**90.0**±1.2	22.5±16.8	1.3±0.7	62.0±2.7
Satimage	77.9±0.9	86.5±1.7	93.8±1.2	35.5±0.4	47.9±3.4	**94.8**±0.8
Satellite	56.7±0.2	69.6±0.5	**76.3**±0.4	46.9±3.7	55.4±1.3	74.6±0.7
Pendigit	34.9±0.6	52.1±6.4	91.0±1.4	44.6±5.3	16.3±2.6	**92.5**±1.3
Annthyroid	31.5±0.5	57.3±1.3	37.8±0.6	**65.4**±2.7	32.1±0.8	57.7±0.6
Mnist	38.5±0.4	51.2±2.5	69.4±0.9	32.6±5.7	33.5±1.7	**71.8**±0.3
Mammo.	**53.4**±0.9	39.0±3.3	38.8±1.5	28.1±4.3	27.1±1.9	43.6±0.5
Shuttle	96.0±0.0	96.4±0.8	97.3±0.2	70.7±1.0	32.0±2.2	**98.2**±0.3
Mullcr.	66.0±0.1	99.1±0.5	**100**±0.0	67.4±2.1	**100**±0.0	**100**±0.0
Forest	18.2±0.2	11.1±1.6	**92.1**±0.3	8.1±2.8	9.9±1.5	58.0±10
Campaign	**49.5**±0.1	42.4±1.0	41.6±0.4	42.4±0.2	36.6±0.1	**49.8**±0.3
Fraud	44.7±0.9	30.3±3.7	**60.5**±1.5	41.0±0.9	17.1±0.4	58.1±3.2
Backdoor	13.4±0.4	3.8±1.2	**88.5**±0.1	3.4±0.2	24.5±0.1	84.1±0.1
mean	44.2	52.6	65.8	39.8	37.0	**68.8**
mean std	**1.5**	3,9	2.2	3.6	4.0	2.0
mean rk	10.5	7.6	5.2	11.5	12.6	**3.1**

Table B.4: Anomaly detection AUROC(↑). We perform 5% T-test to test whether the differences between the highest metrics for each dataset are statistically significant.

Method	DROCC (Thyroid)	DROCC (Arrhyth.)	DROCC (Abalone)	GOAD (Thyroid)	GOAD (kddrev)	GOAD (kdd)	Internal Cont.	NPT-AD (Ours)
Wine	53.5±22	60.1±32	90.9±8.2	95.2±1.9	97.3±1.7	86.3±9.5	**99.5**±0.6	96.6±0.5
Lympho	6.4±5.2	58.6±30	83.7±12	94.8±5.6	79.7±11	59.9±15	99.5±0.3	**99.9**±0.1
Glass	63.5±9.1	55.5±22	75.4±8.9	62.2±14	85.5±7	82.1±6.3	**88.1**±5.0	82.8±2.4
Vertebral	**55.0**±5.1	**58.0**±15	41.2±10	47.0±13	52.2±3.9	49.4±4.2	51.1±3.2	**54.6**±3.9
WBC	6.8±1.8	41.3±25	35.4±13	95.4±0.7	66.1±12	86.6±2.9	95.4±1.1	**96.3**±0.3
Ecoli	N/A	N/A	N/A	82.7±8.4	87.2±3.3	84.7±6.8	86.5±1.2	**88.7**±1.6
Ionosph.	19.6±5.8	83.5±5.6	80.0±2.8	92.4±1.3	96.3±1.1	96.5±1.1	**98.1**±0.4	97.4±1.7
Arrhyth.	53.2±7.0	52.7±8.6	51.2±8.1	80.0±1.9	73.3±5.1	64.3±8.8	**81.7**±0.6	80.1±0.0
Breastw	7.7±8.6	64.4±33	96.6±3.3	98.7±0.8	80.8±9.5	97.7±0.8	**99.1**±0.3	98.6±0.1
Pima	36.2±4.6	54.9±11	69.1±4.9	68.7±3.9	59.3±2.2	63.2±2.3	59.4±2.8	**73.4**±0.4
Vowels	79.4±9.5	72.0±12	95.3±2.1	81.0±2.4	98.5±0.3	97.6±0.5	**99.7**±0.1	99.3±0.1
Letter	77.6±3.3	73.3±5.4	90.0±1.2	60.9±0.7	89.9±0.5	87.6±0.9	92.8±0.9	**96.1**±0.2
Cardio	84.3±4.0	73.8±12	73.5±3.2	**94.8**±1.7	81.3±4.5	84.6±3.0	92.7±0.8	**94.7**±0.2
Seismic	58.2±2.8	60.3±4.5	56.7±1.3	**69.5**±1.5	67.2±1.2	67.9±1.2	62.9±1.0	**69.8**±0.3
Musk	2.3±5.1	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Speech	51.2±5.6	50.5±4.0	52.6±3.4	47.1±1.3	**65.3**±3.2	54.1±4.4	58.9±2.7	54.3±0.3
Thyroid	95.6±0.9	96.1±2.5	98.1±0.3	94.5±1.5	77.1±8.8	89.2±3.0	**98.5**±0.1	97.8±0.1
Abalone	82.4±14	52.9±26	70.6±9.7	89.2±0.9	46.0±3.7	54.3±7.8	**94.3**±0.6	91.6±1.2
Optdig.	84.2±4.6	89.0±4.6	89.5±2.1	66.9±3.3	95.7±0.5	93.1±1.9	**97.5**±1.5	**97.5**±0.3
Satimage	19.1±1.4	87.5±8.8	11.5±1.2	99.1±0.1	86.5±7.1	93.2±1.7	99.8±0.1	**99.9**±0.0
Satellite	64.6±8.9	73.1±1.3	50.2±2.2	69.1±0.8	76.3±1.0	78.2±0.9	**80.6**±1.7	**80.3**±0.9
Pendigits	58.9±7.6	50.8±15	76.6±5.4	87.5±3.9	89.2±2.9	85.1±3.4	99.5±0.1	**99.9**±0.0
Annthyr.	92.9±2.3	86.5±3.6	**93.4**±1.3	76.1±6.5	89.6±4.9	93.2±0.9	80.5±1.3	86.7±0.6
Mnist	N/A	N/A	N/A	90.9±0.4	89.4±0.7	87.7±1.0	**98.2**±0.0	94.4±0.1
Mammo.	81.0±1.3	85.0±2.1	82.0±1.5	66.3±6.4	57.2±1.9	54.5±2.3	81.1±2.0	88.6±0.3
Mullcross	N/A	N/A	N/A	100±0	N/A	51.3±16	**100**±0	**100**±0
Shuttle	N/A	N/A	N/A	88.4±5.5	N/A	99.9±0.0	**100**±0	99.8±0.1
Forest	N/A	N/A	N/A	15.9±6.6	N/A	76.0±5.3	**96.2**±0.6	**95.8**±7.9
Campaign	N/A	N/A	N/A	38.4±2.0	N/A	50.9±2.5	73.7±1.5	**79.1**±0.5
Fraud	N/A	N/A	N/A	83.5±2.7	N/A	86.3±0.8	95.2±0.5	**95.7**±0.1
Backdoor	N/A	N/A	N/A	61.3±10.2	N/A	88.9±1.1	92.6±0.6	**95.2**±0.1
mean	39.8	50.9	53.7	77.3	64.1	78.8	88.8	**89.8**
mean std	4.5	9.1	3.4	3.5	3.2	3.8	**1.0**	**0.8**
mean rk	10.3	10.0	8.7	7.4	7.7	7.2	2.9	**2.5**

Table B.5: Anomaly detection AUROC(↑). We perform 5% T-test to test whether the differences between the highest metrics for each dataset are statistically significant.

Method	NeuTraL-AD (thyroid)	NeuTraL-AD (arrhythmia)	NeuTraL-AD (kddrev)	NeuTraL-AD (kdd)	COPOD	Transformer
Wine	82.9±13.1	**95.4**±1.9	86.0±10.3	85.2±12.9	87.5±1.7	61.4±6.7
Lympho	90.7±5.1	80.9±7.5	93.1±4.1	83.1±9.9	**99.4**±0.4	**99.5**±0.4
Glass	62.2±3.0	62.9±1.2	62.5±4.6	**65.1**±2.9	63.7±3.3	61.2±7.0
Vertebral	32.8±3.5	25.8±4.5	51.9±14.9	**54.4**±16.3	32.6±1.2	44.8±5.2
Wbc	80.4±5.0	92.6±2.2	62.4±8.2	32.5±11.6	**96.3**±0.5	95.0±1.1
Ecoli	43.2±9.3	53.8±9.9	52.5±10.6	49.9±10.9	81.0±1.2	**84.8**±1.6
Iono.	87.9±2.9	**95.7**±1.7	92.9±0.9	87.2±2.7	80.3±2.1	**95.4**±1.9
Arrhyt.	76.0±1.5	80.2±1.6	77.9±1.8	76.4±2.7	80.5±1.3	**81.7**±1.1
Breastw	86.0±2.2	96.2±1.1	95.9±1.6	91.3±5.1	**99.4**±0.2	**99.6**±0.1
Pima	55.1±1.9	60.6±1.2	57.3±2.1	57.3±3.7	65.2±0.7	**67.2**±2.4
Vowels	72.3±3.1	69.7±3.8	62.2±6.1	56.1±8.0	49.6±1.0	**78.4**±9.2
Letter	40.4±3.9	35.4±0.8	36.1±2.9	37.4±4.2	50.1±0.8	**80.5**±4.8
Cardio	74.6±2.7	74.3±3.1	47.7±4.2	28.9±6.3	92.2±0.3	**93.5**±1.3
Seismic	42.7±4.2	39.9±5.5	40.6±9.4	41.0±13.1	**70.8**±0.4	58.2±7.9
Musk	99.5±0.2	99.4±0.2	98.2±1.0	91.9±3.6	94.5±0.2	**100**±0.0
Speech	46.7±1.6	48.0±1.6	52.7±3.2	**54.3**±2.3	49.1±0.5	47.2±0.7
Thyroid	**97.5**±0.2	97.1±0.2	95.8±1.6	79.4±10.4	94.1±0.2	93.8±1.2
Abalone	**92.9**±1.0	92.7±0.8	80.8±2.4	73.9±4.6	86.3±0.3	88.3±2.0
Optdigits	72.3±7.1	82.6±5.4	84.1±4.3	78.4±7.2	68.0±0.4	**96.4**±4.7
Satimage	99.6±0.4	**99.8**±0.1	**99.8**±0.1	**99.7**±0.1	97.4±0.1	**99.7**±0.1
Satellite	80.7±0.4	**81.0**±0.4	76.8±0.6	74.0±1.6	63.5±0.2	73.8±2.5
Pendigits	88.6±5.7	**98.6**±0.8	97.5±0.9	93.0±3.1	90.4±0.2	93.8±2.6
Annthyr.	**87.7**±4.2	80.2±2.9	64.9±1.6	63.7±3.0	77.4±0.4	65.4±1.4
Mnist	**97.5**±0.3	**97.8**±0.2	93.4±0.7	89.9±1.1	77.2±0.2	87.4±3.2
Mammo.	67.6±3.3	73.8±2.5	65.4±3.3	69.6±3.3	**90.5**±0.1	77.6±1.0
Mullcross	98.5±2.1	99.5±1.5	99.6±0.5	99.2±1.5	93.2±0.0	**100**±0.0
Shuttle	99.8±0.3	**99.9**±0.1	**100**±0.0	**99.9**±0.1	99.4±0.0	97.2±2.2
Forestc.	**96.8**±1.5	**96.7**±1.1	84.4±7.0	82.6±6.2	88.4±0.0	95.1±0.8
Campaign	75.4±4.8	70.0±2.1	76.5±0.7	76.4±0.5	**78.3**±0.1	75.3±2.1
Fraud	81.6±7.4	84.8±4.7	93.1±0.6	93.3±0.4	**94.7**±0.1	**94.7**±0.4
Backdoor	91.6±5.8	92.5±7.5	92.9±0.4	92.9±0.3	78.9±0.1	**95.1**±0.2
mean	77.5	79.3	76.6	72.8	79.7	**83.4**
mean std	3.5	2.5	3.6	5.1	**0.6**	2.4
mean rank	8.2	7.1	7.8	8.7	7.5	**6.1**

Table B.6: DROCC Goyal et al. (2020): F1-score (↑) between architecture. The mean rank was computed including all architectures of all models.

Method	DROCC (thyroid)	DROCC (arrhyth.)	DROCC (abalone)
Wine	20.0±19.0	32.0±35.4	**63.0**±20.0
Lympho	0.0±0.0	38.3±23.6	**65.0**±5.0
Glass	**22.2**±17.2	13.3±12.0	14.5±11.1
Vertebral	25.7±5.4	**27.0**±15.9	9.3±6.1
Wbc	0.0±0.0	**18.6**±16.0	9.0±6.2
Ecoli	N/A	N/A	N/A
Ionosph.	29.9±6.8	76.3±6.4	**76.9**±2.8
Arrhyth.	**38.8**±6.2	37.9±8.0	37.1±6.8
Breastw	15.3±7.7	63.8±29.3	**93.0**±3.7
Pima	40.6±3.3	55.2±8.0	**66.0**±4.1
Vowels	33.0±16.4	20.4±15.0	**66.2**±8.8
Letter	39.0±4.8	31.3±6.5	**55.6**±3.6
Cardio	**62.6**±6.1	53.3±12.9	49.8±3.2
Seismic	17.7±2.5	17.9±2.7	**19.1**±0.9
Musk	1.3±3.3	**99.7**±0.9	99.4±1.5
Speech	3.4±2.4	2.1±1.9	**4.3**±2.0
Thyroid	68.4±3.2	69.7±5.7	**72.7**±3.1
Abalone	**44.3**±17.6	11.6±10.5	17.9±1.3
Optdigits	18.4±5.4	26.5±12.6	**30.5**±5.2
Satimage2	10.2±2.5	**33.7**±19.6	4.8±1.6
Satellite	61.3±6.3	**68.1**±0.7	52.2±1.5
Pendigits	7.9±2.9	10.6±7.9	**11.0**±2.6
Annthyr.	63.8±4.7	55.6±5.2	**64.2**±3.3
Mnist	N/A	N/A	N/A
Mammo.	**34.1**±2.2	31.5±6.2	32.6±2.1
Shuttle	N/A	N/A	N/A
Mullcross	N/A	N/A	N/A
Forest	N/A	N/A	N/A
Campaign	N/A	N/A	N/A
Fraud	N/A	N/A	N/A
Backdoor	N/A	N/A	N/A
mean	21.2	28.9	**32.7**
mean std	4.7	8.5	3.4
mean rank	12.6	11.9	10.8

Table B.7: GOAD Bergman and Hoshen (2020): F1-score (↑) between architecture. The mean rank was computed including all architectures of all models.

Method	GOAD (thyroid)	GOAD (kddrev)	GOAD (kdd)
Wine	67.0±9.4	**76.0**±10.8	42.2±26.9
Lympho	**68.3**±13.0	67.7±7.8	46.0±21.5
Glass	12.7±3.9	**25.7**±12	24.0±15.1
Vertebral	16.3±9.6	**26.9**±5.2	25.5±4.7
Wbc	**66.2**±2.9	16.8±16.1	57.2±6.9
Ecoli	61.4±31.7	**69.3**±23.7	66.1±27.8
Ionosph.	83.4±2.6	88.1±2.3	**88.7**±2.7
Arrhyth.	**52.0**±2.3	51.6±4.0	45.2±7.6
Breastw	**96.0**±0.6	73.5±9.4	94.8±1.0
Pima	**66.0**±3.1	57.3±1.9	60.2±2.0
Vowels	31.1±4.2	**78.6**±2.9	72.6±4.5
Letter	20.7±1.7	**53.8**±2.2	48.6±3.0
Cardio	**78.6**±2.5	48.9±5.8	58.4±4.8
Seismic	**24.1**±1.0	18.6±1.9	19.4±2.6
Musk	**100**±0	**100**±0	**100**±0
Speech	4.8±2.3	**8.9**±2.9	4.4±2.4
Thyroid	**72.5**±2.8	17.2±9.4	32.9±9.9
Abalone	**57.6**±2.2	6.2±1.4	6.6±1.0
Optdigits	0.3±0.3	**45.8**±2.6	36.5±9.9
Satimage2	**90.7**±0.7	20.4±10.5	21.7±2.2
Satellite	64.2±0.4	67.9±2.0	**70.1**±0.8
Pendigits	**40.1**±5.0	25.1±3.6	19.4±4.5
Annthyr.	50.3±6.3	61.4±7.8	**68.0**±3.7
Mnist	66.9±1.3	**67.5**±1.2	66.2±1.5
Mammo.	**33.7**±6.1	16.5±1.3	16.0±1.5
Shuttle	73.5±5.1	N/A	**98.4**±0.2
Mullcross	**99.7**±0.8	N/A	36.4±17.0
Forest	0.1±0.2	N/A	**15.0**±4.3
Campaign	16.2±1.8	N/A	**25.6**±2.1
Fraud	**53.1**±10.2	N/A	**53.7**±2.0
Backdoor	12.7±2.9	N/A	**39.9**±3.2
mean	**50.9**	38.3	47.1
mean std	4.4	4.8	6.4
mean rank	7.8	9.5	8.4

Table B.8: NeuTraL-AD Qiu et al. (2021): F1-score (↑) between architecture. The mean rank was computed including all architectures of all models.

Method	NeuTraL-AD (thyroid)	NeuTraL-AD (arrhythmia)	NeuTraL-AD (kddrev)	NeuTraL-AD (kdd)
Wine	51.4±26.2	**78.2**±4.5	62.3±26.9	62.3±28.9
Lympho	46.7±17.9	20±18.7	**54.2**±15.7	34.2±15.3
Glass	7.5±6.2	9.0±4.4	9.5±5.9	**13.0**±7.8
Vertebral	9.2±3.4	3.8±1.2	**23.3**±9.8	13.0±7.8
Wbc	40.7±10	**60.9**±5.6	21.1±10	13.0±7.8
Ecoli	4.0±5.8	7.0±7.1	6.5±11.1	**8.0**±12.5
Ionosph.	79.2±2.8	**90.6**±2.4	86.9±1.4	**79.4**±4.0
Arrhyth.	54.9±3.4	**59.5**±2.6	57.7±1.6	57.2±3.1
Breastw	80.2±2.0	**91.8**±1.3	89.6±2.9	85.6±5.6
Pima	55.4±1.7	**60.3**±1.4	57.2±1.9	56.8±2.5
Vowels	**13.2**±6.3	10±6.2	5.0±3.8	3.9±3.4
Letter	4.9±1.7	**5.7**±0.8	3.6±1.2	4.8±2.8
Cardio	**46.9**±3.9	45.5±4.3	14.7±5.0	3.8±2.7
Seismic	**12.8**±1.3	11.8±4.3	8.7±4.4	10.7±7.9
Musk	98.9±0	**99.0**±0	79.3±11.2	43.4±16.5
Speech	5.3±1.8	4.7±1.4	4.3±2.4	**6.1**±2.7
Thyroid	**75.6**±2.3	69.4±1.4	61.4±8.4	26.6±17.0
Abalone	**60.8**±4.2	53.2±4.0	45.6±8.6	47.1±8.3
Optdigits	11.9±7.0	16.2±7.3	**18.5**±9.6	17.1±9.4
Satimage2	85.8±9.0	92.3±1.9	**92.4**±1.4	91.3±0.9
Satellite	**72.7**±0.3	71.6±0.6	70.4±0.6	66.9±2.1
Pendigits	32.4±14.3	**69.8**±8.7	58.4±8.9	42.2±13.2
Annthyr.	**53.5**±5.1	44.1±2.3	33.2±2.1	29.1±4.3
Mnist	82.8±0.9	**84.8**±0.5	68.8±2.5	60.8±2.8
Mammo.	11.3±1.7	19.2±2.4	**20.8**±3.7	19.1±4.9
Shuttle	97.1±0.4	**97.9**±0.2	97.6±0.1	97.5±0.1
Mullcross	88.9±12.2	**96.3**±10.5	96.2±3.6	92.9±9.4
Forest	**64.6**±9.9	51.6±8.2	12.2±11.4	8.7±7.6
Campaign	**52.0**±4.1	42.1±1.7	51.6±0.1	51.2±0.8
Fraud	24.7±7.8	24.3±7.8	**61.0**±5.2	55.9±4.2
Backdoor	84.9±5.0	84.4±1.8	**87.3**±0.2	86.8±0.3
mean	48.7	**50.8**	47.0	41.7
mean std	5,8	4.0	5.9	7.0
mean rank	9.8	9.0	9.4	10.7

B.3 Dataset classification

Table B.9: Necessary dependencies to take into account for AD by mask-reconstruction. Given the obtained hierarchy between NPT-AD, Mask-KNN and the transformer on each dataset, we display the necessary dependencies to identify efficiently anomalies.

Dependencies	feature-feature	sample-sample
Wine	✓	✓
Lympho	✓	×
Glass	×	✓
Vertebral	×	✓
Wbc	✓	✓
Ecoli	✓	×
Ionosph.	✓	✓
Arrhyth.	✓	✓
Breastw	✓	✓
Pima	✓	✓
Vowels	×	✓
Letter	✓	✓
Cardio	✓	✓
Seismic	×	✓
Musk	✓	✓
Speech	✓	✓
Thyroid	✓	✓
Abalone	✓	✓
Optdigits	×	✓
Satimage	✓	✓
Satellite	×	✓
Pendigits	×	✓
Annthyr.	✓	✓
Mnist	×	✓
Mammo.	×	✓
Shuttle	✓	✓
Mulcross	✓	✓
Forest	✓	×
Campaign	✓	✓
Fraud	✓	×
Backdoor	✓	×

B.4 External Retrieval Modules

Table B.10: Anomaly detection F1-score (↑). We perform 5% T-test to test whether the difference between the highest metrics for each dataset is statistically significant.

Method	Transformer	+KNN	+v-att.	+att-bsim	+att-bsim -bval
Wine	23.5±7.9	24.9±5.9	26.5±7.3	**29.0**±7.7	27.0±5.6
Lympho	88.3±7.6	89.2±7.9	88.3±9.3	**90.0**±8.1	89.1±7.9
Glass	**14.4**±6.1	12.8±3.9	12.2±3.3	12.8±3.9	11.1±0.1
Verteb.	12.3±5.2	**15.7**±2.8	12.7±4.1	14.3±2.6	13.5±5.1
Wbc	66.4±3.2	65.2±4.0	66.2±5.2	65.5±4.4	**67.9**±4.2
Ecoli	75.0±9.9	**75.6**±7.5	**75.6**±9.7	73.8±8.0	75.0±9.7
Ionosph.	88.1±2.8	85.7±3.4	86.0±4.7	**91.7**±2.1	79.3±1.6
Arrhyth.	59.8±2.2	60.3±2.2	60.2±2.7	**61.2**±2.1	**61.1**±2.8
Breastw	96.7±0.3	96.7±0.3	**96.8**±0.3	96.7±0.3	96.7±0.3
Pima	65.6±2.0	64.7±3.1	64.0±3.3	64.3±2.4	**67.0**±1.5
Vowels	28.7±8.0	40.0±10.0	49.1±11.1	44.5±10.5	**58.0**±11.2
Letter	41.5±6.2	32.9±11.8	41.8±11.5	**43.7**±10.3	28.5±7.1
Cardio	**68.8**±2.8	65.6±3.6	62.8±6.9	67.7±3.7	68.3±4.5
Seismic	**19.1**±5.7	17.4±5.5	**19.5**±6.3	16.7±5.5	17.5±5.4
Musk	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Speech	**6.8**±1.9	6.3±1.4	5.7±1.7	5.9±1.5	5.9±1.7
Thyroid	55.5±4.8	55.5±4.9	**56.0**±5.9	**55.8**±6.3	55.3±6.6
Abalone	42.5±7.8	42.5±9.5	49.8±6.5	**53.0**±5.7	43.2±9.1
Optdig.	61.1±4.7	**70.7**±16.5	51.5±7.6	62.6±6.5	22.8±5
Satimage2	89.0±4.1	86.8±0.4	90.7±2.6	**93.2**±1.7	64.2±7.2
Satellite	65.6±3.3	58.6±2.9	57.3±3.0	**71.9**±1.5	53.7±3.3
Pendig.	35.4±10.9	**52.1**±9.0	39.0±14.5	**53.4**±9.8	34.2±12.2
Annthyr.	29.9±1.5	**30.4**±1.9	**30.3**±1.5	**30.3**±1.6	**30.5**±1.4
Mnist	56.7±5.7	**64.2**±3.7	61.7±1.0	61.6±1.0	56.7±1.9
Mammo.	17.4±2.2	17.3±2.4	15.5±2.5	17.2±3.0	**17.7**±2.7
Shuttle	85.3±9.8	90.8±2.9	67.7±13.7	87.8±3.7	**95.6**±1.8
Mullcr.	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Forest	21.3±3.1	18.6±4.6	21.0±5.9	**24.9**±6.5	11.1±4.1
Camp.	47.0±1.9	48.5±2.1	43.3±2.3	49.7±1.2	49.1±1.1
Fraud	53.4±4.4	56.4±2,1	56.3±2.1	57.1±2.1	55.2±1.8
Backd.	85.8±0.6	**86.1**± 0.6	85.2± 0.7	85.3± 0.6	82.4± 1.3
mean	56.2	56.1	55.7	**58.6**	53.9
mean std	4.4	4.6	5.0	**3.7**	**3.7**

Table B.11: Anomaly detection AUROC(↑). We perform 5% T-test to test whether the differences between the highest metrics for each dataset are statistically significant.

Method	Transformer	+KNN	+v-att.	+att-bsim	+att-bsim -bval
Wine	61.4±6.7	60.4±5.4	62.1±6.4	**63.5**±7.8	**64.5**±5.6
Lympho	99.5±0.4	99.6±0.4	99.6±0.5	**99.7**±0.3	**99.7**±0.3
Glass	61.2±7.0	61.2±5.0	**62.1**±7.0	59.3±6.9	59.1±5.8
Vertebral	44.8±5.2	**46.7**±4.1	45.3±7.1	45.4±3.7	45.4±4.7
WBC	95.0±1.1	94.3±1.5	94.3±1.6	94.2±1.1	**95.5**±1.6
Ecoli	84.8±1.6	84.8±1.8	85.2±2.7	**87.4**±1.8	85.4±2.3
Ionosph.	95.4±1.9	93.7±2.7	93.6±4.0	**97.5**±0.1	87.2±2.3
Arrhyth.	81.7±1.1	81.9±0.9	81.8±0.9	**82.3**±0.7	**82.1**±0.9
Breastw	**99.6**±0.1	**99.6**±0.1	**99.6**±0.1	**99.6**±0.1	**99.6**±0.1
Pima	67.2±2.4	66.0±3.8	65.4±3.8	65.8±2.9	**68.7**±1.4
Vowels	78.4±9.2	86.1±5.2	90.4±4.7	88.3±4.5	**94.3**±2.8
Letter	80.5±4.8	73.5±9.6	**81.0**±8.7	**81.5**±6.8	69.1±7.7
Cardio	**93.5**±1.3	92.0±1.7	89.9±4.2	**93.3**±1.7	**93.7**±1.3
Seismic	**58.2**±7.9	56.8±8.4	57.9±7.6	**58.0**±6.7	54.8±6.2
Musk	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Speech	47.2±0.7	**47.3**±0.8	**47.3**±0.8	**47.3**±0.8	47.0±0.5
Thyroid	**93.8**±1.2	**93.8**±1.2	93.6±5.9	93.7±1.5	93.6±1.8
Abalone	**88.3**±2.0	86.1±3.6	**88.0**±3.5	**87.9**±3.7	86.6±2.9
Optdig.	96.4±4.7	96.2±9.8	94.9±1.7	**96.7**±1.1	83.4±3.2
Satimage	99.7±0.1	99.5±0.2	99.6±0.2	**99.8**±0.1	96.8±1.9
Satellite	73.8±2.5	68.9±2.0	67.8±2.5	**79.5**±1.9	62.0±2.9
Pendigits	93.8±2.6	96.5±1.4	94.1±2.8	**97.1**±1.2	89.4±7.1
Annthyr.	65.4±1.4	66.0±1.7	**66.2**±1.3	**66.2**±1.6	66.0±1.1
Mnist	87.4±3.2	**90.3**±2.2	89.9±0.4	**90.0**±0.5	87.3±1.0
Mammo.	77.6±1.0	76.8±2.4	75.2±2.9	78.4±1.6	**79.8**±1.8
Mullcross	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0	**100**±0.0
Shuttle	97.2±2.2	98.1±0.5	90.2±6.1	97.7±0.9	**98.9**±0.3
Forest	95.1±0.8	94.5±0.7	95.0±1.0	**95.4**±0.9	92.7±0.7
Campaign	75.3±2.1	75.7±1.9	69.3±2.0	76.1±2.0	75.9±2.1
Fraud	94.7±0.4	95.1±0.4	95.2±0.4	**95.8**±0.4	94.7±0.4
Backdoor	**95.1**±0.2	**95.2**±0.3	94.5±0.2	94.7±0.1	91.7±0.3
mean	83.4	83.1	83.1	**84.4**	82.1
mean std	2.4	2.8	2.6	2.0	2.3