# Novel class discovery in tabular data : an application to network fault diagnosis

Colin Troisemaine

COLLEGES BRETAGNE LOIRE
SCIENCES POUR L'INGENIEUR ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

# THÈSE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPÉRIEURE
MINES-TÉLÉCOM ATLANTIQUE BRETAGNE
PAYS DE LA LOIRE – IMT ATLANTIQUE

ÉCOLE DOCTORALE N° 648
*Sciences pour l'Ingénieur et le Numérique*
Spécialité : *Informatique*

Par
Colin TROISEMAINE

# Novel Class Discovery in Tabular Data : an Application to Network Fault Diagnosis

**Thèse présentée et soutenue à IMT Atlantique, Brest, le 4 octobre 2024**
**Unité de recherche : Lab-STICC**
**Thèse N° : 2024IMTA0422**

## Rapporteuses avant soutenance :

| | |
|---|---|
| Michèle SEBAG | Professeure à Université Paris Saclay |
| Pascale KUNTZ | Professeure à Université de Nantes |

## Composition du Jury :

| | | |
|---|---|---|
| Président : | Eric FABRE | Directeur de Recherches à INRIA, Rennes |
| Examinatrices : | Michèle SEBAG | Directrice de Recherche au CNRS |
| | Pascale KUNTZ | Professeure à Université de Nantes |
| | Catherine LEPERS | Professeure à Telecom SudParis |
| Dir. de thèse : | Sandrine VATON | Professeure à IMT Atlantique, Brest |
| Encadrants : | Stéphane GOSSELIN | Ingénieur de recherche à Orange Innovation, Lannion |
| | Alexandre REIFFERS-MASSON | Maître de conférences à IMT Atlantique, Brest |

## Invités :

| | |
|---|---|
| Vincent LEMAIRE | Ingénieur de recherche à Orange Innovation, HDR, Lannion |
| Joachim FLOCON-CHOLET | Ingénieur de recherche à Orange Innovation, Lannion |

# Remerciements

Je tiens d'abord à exprimer ma gratitude à mes encadrants : Stéphane Gosselin pour la qualité de nos échanges et la pertinence de ses observations; Alexandre Reiffers-Masson pour sa curiosité et son habileté à me faire découvrir de nouveaux concepts; Sandrine Vaton pour son encadrement bienveillant et ses conseils éclairés; et Joachim Flocon-Cholet pour sa chaleur humaine et son expertise. Leur maîtrise des divers sujets de cette thèse, leurs encouragements et leur soutien continu ont été essentiels à la réalisation de ce travail. Je leur suis particulièrement reconnaissant pour la confiance qu'ils m'ont témoignée et pour l'équilibre entre liberté et encadrement qu'ils m'ont offert. Leur disponibilité et leurs précieux conseils ont grandement contribué à la qualité de cette thèse.

Je voudrais ensuite remercier Michèle Sebag et Pascale Kuntz d'avoir accepté d'être les rapporteuses de cette thèse, ainsi que Catherine Lepers et Eric Fabre pour leur participation au jury. Je leur suis reconnaissant pour l'intérêt qu'ils portent à mes travaux.

Je souhaite exprimer ma profonde gratitude à Vincent Lemaire, qui m'a non seulement initié au monde de la recherche et encouragé à entreprendre cette thèse, mais qui m'a aussi accompagné tout au long de celle-ci. J'ai tiré beaucoup d'enseignements de nos nombreux échanges, où il a fait preuve d'une grande patience et pédagogie. Vincent restera pour moi un exemple par sa générosité, son honnêteté et sa maturité. Cette thèse n'aurait pas été aussi aboutie sans ses conseils avisés et son soutien indéfectible.

Je tiens à ne pas oublier les collègues avec qui j'ai eu la chance de partager mon quotidien, en particulier Mina Rafla, Xihui Wang, Victor Guyomard, Thibault Cordier et Charbel Kindji. Leur présence et leur précieuse amitié ont été pour moi des sources de motivation et de réconfort tout au long de ce parcours.

Je souhaite également remercier l'équipe ADN, en particulier Emmanuelle Cressan pour son implication dans l'organisation des différentes missions, ainsi que Aleksandra, Jean-Michel, Aurélien et Jean-Luc pour leur accueil chaleureux. Enfin, je suis reconnaissant envers l'équipe PROF, notamment à Françoise, Marc, Hugo, Nathan, Tanguy, Franck et Nicolas pour m'avoir régulièrement reçu à leur table et pour toutes les discussions enrichissantes et passionnées que nous avons pu avoir.

# Table of Contents

# NOTATIONS AND METRICS

To ensure clarity and consistency throughout this thesis, a standardized set of notations will be employed. In Table 1, we provide a list of the notations commonly used throughout this manuscript, along with their definitions.

Table 1 – Frequently used notations and their meanings.

| **Symbol** | **Meaning** |
|---|---|
| $D^l$ and $D^u$ | labeled and unlabeled sets, composed of a set of samples and their corresponding class labels. |
| $N$ and $M$ | number of samples in $D^l$ and $D^u$. |
| $X^l$ and $X^u$ | labeled and unlabeled data sets in $\mathbb{R}^{d\times N}$ and $\mathbb{R}^{d\times M}$. |
| $x^l$ and $x^u$ | data samples in $\mathbb{R}^d$. |
| $C^l$ and $C^u$ | number of known and novel classes. |
| $\mathcal{Y}^l$ and $\mathcal{Y}^u$ | target spaces in $\mathbb{R}^{C^l}$ and $\mathbb{R}^{C^u}$. |
| $Y^l$ and $Y^u$ | corresponding class labels of $X^l$ and $X^u$ in $\mathcal{Y}^l/\mathcal{Y}^u$. |
| $\mathbb{1}$[condition] | the indicator function evaluates to 1 if the condition is true and 0 otherwise. |
| $z$ | the projection of a sample $x$ in the latent space of an encoder. |

The algorithms explored in this thesis will be evaluated with three main metrics:

• **The Clustering Accuracy** (ACC) [1] measures the extent to which the predicted labels match the ground truth labels. Since the cluster label numbers are random and may not correspond to the ground truth class labels, a mapping must be found to optimally align them. This mapping can be obtained using the Hungarian algorithm [2] (also known as the Kuhn-Munkres algorithm).

$$ACC = \frac{1}{M}\sum_{i=1}^{M}\mathbb{1}[y_i^u = \text{map}(\hat{y}_i^u)] \tag{1}$$

where $\text{map}(\hat{y}_i^u)$ is the mapping of the predicted label $\hat{y}^u$ for sample $x_i^u$. In other words, the clustering accuracy is just the usual accuracy computed after finding the optimal alignment between the randomly associated cluster numbers and the ground truth labels.

• **The Normalized Mutual Information** (NMI) indicates the correspondence between the predicted and ground truth labels, and is invariant to permutations.

$$NMI = \frac{I(\hat{y}^u, y^u)}{\sqrt{H(\hat{y}^u)H(y^u)}} \tag{2}$$

where $I(\hat{y}^u, y^u)$ is the mutual information between $\hat{y}^u$ and $y^u$ and $H(y^u)$ and $H(\hat{y}^u)$ are the entropies of the empirical distributions of $y^u$ and $\hat{y}^u$ respectively. The mutual information is computed as:

$$I(\hat{y}^u, y^u) = \sum_{\hat{y}^u, y^u} P(\hat{y}^u, y^u) \log \left( \frac{P(\hat{y}^u, y^u)}{P(\hat{y}^u)\, P(y^u)} \right) \tag{3}$$

• **The Adjusted Rand Index** (ARI) evaluates the similarity between two sets of labels by taking into account the similarity that would be expected by random chance. It adjusts the Rand Index (RI) for the expected RI of random labeling.

$$ARI = \frac{RI - Expected\_RI}{max(RI) - Expected\_RI} \tag{4}$$

where $RI = \frac{a+b}{\binom{n}{2}}$, $a$ is the number of pairs of elements grouped in $y$ that are also grouped in $\hat{y}$, $b$ is the number of pairs of elements that are in different groups in $y$ and in $\hat{y}$, and $\binom{n}{2}$ is the total number of possible pairs in the dataset.

These three metrics are symmetric, i.e. $metric(a, b) = metric(b, a)$, and range between 0 and 1 [1], with values closer to 1 indicating better agreement between the cluster and ground truth labels.

1. The ARI actually ranges from -1 to 1, with negative values indicating a performance worse than random.

Chapter 1

# Introduction

## Contents

## 1.1 The value of reliable telecommunication networks

Driven by technological advancements, the global economy has become increasingly reliant on internet services. Instant communication, project management, marketing or e-commerce have all become integral to modern business operations, facilitating global reach and operational efficiency. Optical fiber has played an especially pivotal role in these advancements by vastly improving internet speeds. According to the ARCEP (an independent French agency responsible for regulating telecommunications), as of March 31, 2018, 10.9 million French premises were eligible for Fiber-To-The-Home (FTTH) offers. And by September 30, 2023, this number had grown to 38.9 million premises [3], covering more than 84% of the fixed internet access market. Optical fiber has enabled the integration of high-speed, low-latency applications such as video streaming, online gaming, cloud computing, telemedicine and smart city infrastructure into everyday life. And this evolution is set to continue as more and more bandwidth-eager devices and services are connected. For instance, from Cisco's estimations [4], the number of 4K televisions in homes quadrupled between 2018 and 2023.

As the primary entities responsible for providing and maintaining internet connectivity, Internet Service Providers (ISPs) face many challenges from this rapid growth of the FTTH network. ISPs build and maintain the network infrastructure by laying optical fiber, installing routers, switches, data centers and other equipment necessary for data transmission [1]. They also manage and optimize the quality of service, ensuring that users receive reliable and high-speed internet connections. And with an ever-changing network consisting of a wide variety of equipment, it is becoming increasingly difficult to provide high quality services with optimal uptime. Even with reliable hardware and software, faults will inevitably occur, affecting the end-user experience. According to a 2023 study by the Uptime Institute [5], 70% of IT and data centers outages cost businesses more than $100,000 per outage. Sometimes faults can be spectacular, for example in 2016 a single router failure caused Delta Airlines to cancel 2,300 flights and lose $177 million. The quick resolution of such faults is imperative, as several ISPs usually compete for subscribers, and users may switch to alternative providers. Therefore, effective fault management is a key aspect of operating telecommunication networks, and especially in access network.

1. In some cases, ISPs do not build their own network and instead lease the infrastructure of commercial operators, rather than building and maintaining their own. This model is often referred to as a "virtual network operator" (VNO).

## 1.2 The fault management process

Fault Management (FM) is the process of locating, analyzing and resolving network problems (such as a cut fiber, network capacity overload or cyberattack) [6]. Figure 1.1 shows the FM process from detection to recovery.

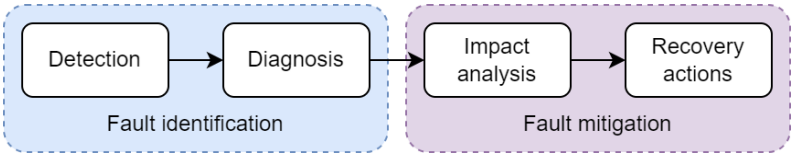



Figure 1.1 – The Fault Management process.

Fault identification involves two complementary tasks: detection and diagnosis [7]. The purpose of fault **detection** is to decide whether the network is operating normally or not. It can be triggered after observing a problem or proactively to anticipate upcoming problems. And in the event of a fault, the **diagnosis** step (also called *root cause analysis*) seeks to deduce the original cause of a fault or group of faults. It is sometimes divided into two sub-tasks: localization and identification. The purpose of the mitigation process is to determine the necessary actions to definitively resolve the fault. It starts with **impact analysis**, where the nature of the fault (criticality, duration, scope, etc.) is determined. This step helps to understand the risks associated with the fault (process or product failures) and allows the operator to prioritize the most serious faults. Finally, the **recovery actions** often consist of dispatching technicians to repair the previously identified software or hardware causing the fault.

In the context of this research, we consider that it is the role of the ISPs to collect the data describing the state of their network and to detect if it is subject to a fault. The impact analysis and recovery actions can arguably only be carried out by network experts, so we are only interested in the diagnosis step. The production of a diagnosis is often considered the most difficult step in automatic fault management [8], as a good diagnosis must allow the exact location of the fault to be determined and the repair to be definitive.

In the past, fault diagnosis was performed manually by network experts who examined each fault individually using a limited number of descriptors. However, the evolution of telecommunications networks over the last two decades has significantly increased their size and complexity, to the point where network diagnosis has become too complicated for human experts. For this reason, fault diagnosis in the real world is now mostly conducted

through expert systems (often rule-based algorithms) [9]. Similar to how an expert studies a specific case, these systems analyze data from a large number of performance indicators before reaching a conclusion on the nature of the fault. Rule-based expert systems encapsulate the knowledge of the experts of the network in the form of rules. While their decisions are easy to interpret, they have a number of drawbacks. Firstly, these solutions are highly specific and cannot be transferred to different domains of application or network architectures. Then, they still require manual maintenance by network experts and, as the volume of data and the number of potential sources of faults increase, they are becoming increasingly difficult to manage. Finally, because the rules may not cover all possible cases, some faults may not match any of the rules and go undiagnosed. These undiagnosed faults will require additional attention and investigation time, increasing the operating costs.

*Orange*, the company where I conducted my thesis, is an ISP who is rapidly expanding its Fiber-To-The-Home services. Similarly to most internet providers, *Orange France* has developed its own expert diagnosis system called DELC (for *Diagnostic Expert de la Ligne Client*, or *expert diagnosis of the customer's line*). It is able to provide a diagnosis for the majority of the faults, but some remain undiagnosed and often require a costly investigation by a technician. With millions of interventions by technicians every year, this is a topic of crucial importance for *Orange*.

Thus, the goal of this thesis is to discover new diagnoses for the faults for which the DELC system was unable to determine the root cause. In this manuscript, we will consider that we have a set of diagnosed faults predefined by network experts (the *known classes*) and a set of undiagnosed faults (the *novel classes*) that we need to explore. To design a system that can be used with other network architectures or problems, and to avoid creating a system that is overly specific to DELC, we will attempt to create a method that can be applied to partition any set of unknown (or *novel*) classes, given a set of known classes.

## 1.3 Automatic network diagnosis systems

Before introducing the task of network diagnosis as a machine learning problem, we briefly discuss existing diagnosis systems. Because large FTTH networks consist of devices from many different vendors, with a plethora of protocols and management software, automating network diagnosis is not an easy process. Network equipment vendors such as

Cisco, Nokia or Huawei offer solutions that can only monitor their own devices. To achieve full coverage of their network, ISPs must deploy management solutions from multiple vendors, each focused on specific aspects of the network [10]. However, as each vendor uses different performance indicators, metrics and data structures, this results in a significant amount of effort to link the different pieces of software together to gain a global view of the network. There is currently no holistic solution proposed to provide an overview of such heterogeneous networks, which makes diagnosis a complex task and is the crux of the problem we address in this thesis.

Recently, there has been a growing interest in statistical and data-driven approaches. Machine learning has been particularly popular to leverage the large amounts of data collected by ISPs [11]. Indeed, it is a much more attractive solution to query each equipment for its individual information before merging all the collected variables into a single unified dataset describing the network. In this way, the same methodology can be applied on datasets from different ISPs, as long as the objective is clearly defined. In the fault diagnosis literature, methodologies are generally categorized in three groups [12, 13, 14]: **Model-based** methods aim at representing processes based on expert knowledge before applying mathematical models. They can lead to very accurate diagnoses, but are inherently difficult to scale to complex systems. **Signal-based** approaches rely on a fundamental understanding of the process physics to identify possible abnormalities and faults by comparing detected signals to previous measures of normal operation. Primarily employed with physical machinery, such as production lines, these approaches are less applicable to our case. Lastly, **data-driven** approaches leverage large amounts of historical data and are well suited for complex industrial systems such as FTTH networks. However, many data-driven methods stop at the fault detection step and do not attempt to create groups of similar faults, let alone diagnosing the root cause.

It should be noted here that most of the literature assumes that all possible root causes of failures are known in advance, often reducing the diagnosis to a classification task.

At *Orange*, two theses have already been carried out with the aim of improving the coverage of the diagnosis system. First, in [15], Serge Romaric Tembo Mouafo examined the applicability of Bayesian networks, which have been extensively researched for the diagnosis task. These models represent the variables of the network as nodes, with edges representing the conditional dependencies between the nodes. In theory, the parameters of these models can be automatically estimated from the data. However, practical im-

plementation revealed that Bayesian networks are highly application-specific and require manual tuning of weights by an expert of the network to reflect reality. Maintenance becomes time-consuming when the network changes or new faults arise, especially in large access networks like *Orange*'s. This is the main drawback of model-based methodologies, to which S. Tembo's work [15] belongs.

And in [16], Amine Echraibi developed probabilistic graphical models based on Dirichlet process mixtures. Interestingly, these models are capable of identifying clusters of similar faults without prior knowledge of the number of clusters. However, in the Dirichlet process, the size of the clusters decreases geometrically with their number, which is not necessarily true in real-world scenarios. Additionally, it was chosen in this work to approximate the posterior distribution of the target variable through the mean field technique, which makes the assumption that instances of a dataset are independant. With network data, this assumption is clearly violated as multiple clients can be affected by the same fault. Finally, as the authors note, it is very difficult to create a clean dataset from the large number of heterogeneous measures extracted from the access network. This was the main obstacle to validate experimentally the efficacy of their methods, which we tackle in Chapter 6. A. Echraibi's work falls under the data-driven category, but distinguishes itself by attempting to interpret the clusters of faults it creates by studying their most descriptive features.

## 1.4 Network diagnosis as a machine learning problem

In this thesis, we approach the problem of automatic fault diagnosis from a data-centric point of view. As we have seen before, fully modeling the network of an ISP is not feasible in practice, as it is a monumental task that requires continuous maintenance. Instead, we assume that for each fault, the ISP has collected a number of features describing the network at the time of the failure. Then, over the course of a few weeks or months, the faults are compiled into a single dataset.

*Orange*'s rule-based expert diagnosis system, DELC, diagnoses between 80% and 90% of the FTTH faults. And undiagnosed faults are labeled as DNIs (for *Défauts Non Identifiés*, or *unidentified faults*). Thus, we can consider that we have two distinct sets of data: one made up of the *known* diagnoses where all faults have been labeled by the expert system, and the other made up of the unlabeled DNIs. For the time being, we assume that the rules for known diagnoses in the expert system have complete coverage, and therefore

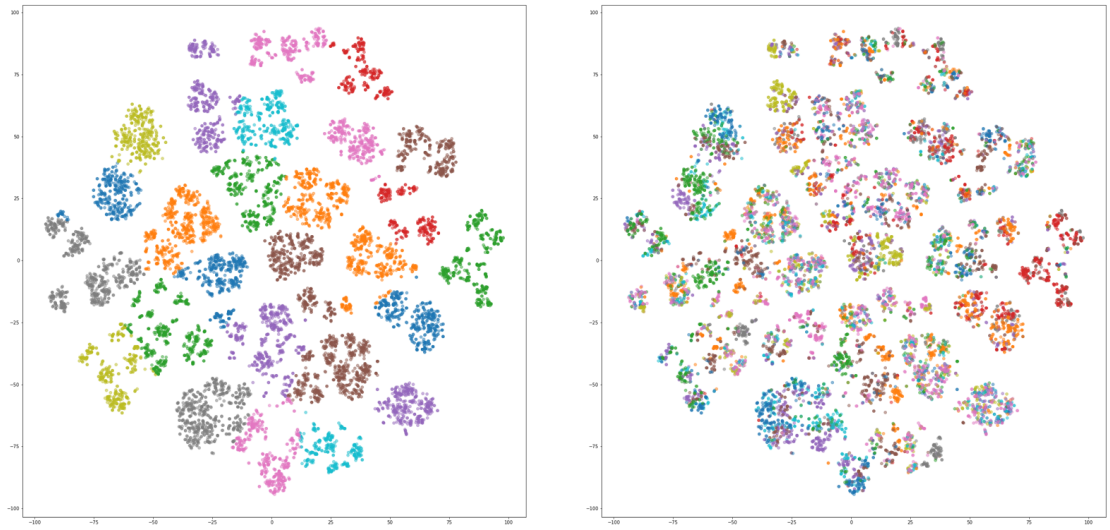
that the unlabeled faults do not contain instances of the known diagnoses. Indeed, we are only interested in discovering new diagnoses. Updating the definition of known diagnoses could be an interesting future development of our work. The collected data is comprised of a large number of heterogeneous features such as signals strengths, software versions, timestamps, equipment statuses, etc. After preprocessing, the data can be shaped into a table, with each row representing a fault and each column a feature.

While exploring the sub-domains of *open-world* Machine Learning (see Chapter 2), we came across the nascent field of Novel Class Discovery (NCD). Very similar to our problem, we are given during training a labeled set of known classes and an unlabeled set of different and unknown classes. And the goal is to discover the underlying classes within the unlabeled data. The first groundbreaking paper was published in 2021 [17], and NCD has remained largely focused on image data ever since. As there is a strong spatial correlation between the pixels of an image, powerful techniques such as convolution, data augmentation or Self-Supervised Learning [18] can be applied to improve performance. But since there is no spatial correlation between the heterogeneous features of tabular data, these techniques cannot be used and state-of-the-art NCD methods cannot be directly employed. However, we believe that the general philosophy and concepts behind NCD can still be transferred to our case. For these reasons, we consider in this thesis that our fault diagnosis problem is a Novel Class Discovery problem in tabular data, and we make both theoretical and practical contributions to this domain.

## 1.5 The limits of unsupervised clustering and the need for NCD

Before we proceed, we would like to quickly address a question that often arises when introducing the NCD problem: "*Why cannot unsupervised clustering methods solve the NCD problem?*" While simply clustering the unlabeled set of unknown classes could work with some datasets, the more complex cases prove difficult for unsupervised methods. We illustrate this idea in Figure 1.2 with a dataset comprised of real FTTH faults, where the classes are the known diagnoses of the DELC system. In Figure 1.2(a), the prediction of a $k$-means algorithm is the typical result expected from an unsupervised clustering method. The clusters' edges are clearly defined, and visually isolated groups of points have been grouped together. However, the reality of the ground truth in Figure 1.2(b) tells another

story. The obvious clusters of points shown by the t-SNE [2] and confirmed by $k$-means are completely different to the underlying classes we are trying to discover. This means that the features relevant to the diagnosis problem do not have enough weight in the original feature space.



(a) Points colored with the prediction of $k$-means in the original feature space.

(b) Points colored with the ground truth labels.

Figure 1.2 – t-SNE visualization of the instances of the 80 most represented classes of DELC.

As with most NCD problems, clusters found by purely unsupervised means are rarely of interest to domain experts who have already drawn the most obvious conclusions. For example, we observe that clustering algorithms tend to group faults according to the model of home modem, which is not useful for diagnosis. So in NCD, the problems are generally too complex for unsupervised clustering, and a labeled set of known classes is used to extract a general idea of what constitutes an interesting class. These known classes are a translation of the experts' knowledge and will guide the clustering process. Some works even define the goal of NCD as extracting the distinctive high-level features of the known classes into a representation where the novel classes are easily separable [19]. In this thesis, we will inject the experts' knowledge into a latent space using a supervised objective that predicts the known classes.

2. t-distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction technique used for visualizing high-dimensional data in lower dimensions while preserving local structure.

## 1.6 Contributions and organization of the document

This thesis is organized in 5 main chapters which we describe in this section.

We begin in Chapter 2 with a comprehensive survey of the state-of-the-art Novel Class Discovery methods. After formally defining the NCD problem, we give an overview of the different families of approaches. For each family, we describe their general principle and detail a few representative methods. We also present some common tools and techniques used in NCD, such as pseudo-labeling, self-supervised learning and contrastive learning. Finally, to help readers unfamiliar with the NCD problem differentiate it from other closely related domains, we summarize some of the closest areas of research and discuss their main differences. The contents of this chapter have been submitted to an international machine learning journal and are available as a preprint in the meantime:

[20] **Novel Class Discovery: an Introduction and Key Concepts.** Colin Troisemaine, Vincent Lemaire, Stéphane Gosselin, Alexandre Reiffers-Masson, Joachim Flocon-Cholet and Sandrine Vaton. In: ArXiv preprints, 2023.

In Chapter 3, we propose a first method for solving the NCD problem in tabular data, which we call *TabularNCD*. We extract knowledge from already known classes into a latent space to guide the discovery process of novel classes. We propose a new pseudo-labeling process and follow recent findings in Multi-Task Learning to optimize a joint objective function. Extensive experiments are conducted to evaluate our method and demonstrate its effectiveness against 3 competitors on 7 diverse public classification datasets. This method has been published at an international conference:

[21] **A Method for Discovering Novel Classes in Tabular Data.** Colin Troisemaine, Joachim Flocon-Cholet, Stéphane Gosselin, Sandrine Vaton, Alexandre Reiffers-Masson and Vincent Lemaire. In: IEEE International Conference on Knowledge Graph (ICKG), 2022.

Its French translation has been presented at the following national conference:

[22] **Découvrir de nouvelles classes dans des données tabulaires.** Colin Troisemaine, Joachim Flocon-Cholet, Stéphane Gosselin, Sandrine Vaton, Alexandre Reiffers-Masson and Vincent Lemaire. In: Extraction et Gestion des Connaissances (EGC), 2023.

Chapter 4 is dedicated to solving the NCD problem under more realistic conditions. In particular, we consider that no prior knowledge of the novel classes is available. To this end, we propose to tune the hyperparameters by adapting the $k$-fold cross-validation process. Since we have found that methods with too many hyperparameters are likely to overfit the known classes, we propose a new model focused on simplicity. Furthermore, we find that the latent space of this method can be used to reliably estimate the number of novel classes. In addition, we adapt two unsupervised clustering algorithms ($k$-means and Spectral Clustering) to leverage the knowledge of the known classes. The contents of this chapter were presented in the following paper, which is part of the journal track of the *ECML/PKDD* international conference:

[23] **A Practical Approach to Novel Class Discovery in Tabular Data.** Colin Troisemaine, Alexandre Reiffers-Masson, Stéphane Gosselin, Vincent Lemaire and Sandrine Vaton. In: Data Mining and Knowledge Discovery, 2024.

Chapter 5 presents an interactive interface for interpreting the results of clustering or NCD algorithms. The interpretation of the domain- and application-specific attributes of tabular data is difficult and often requires a domain expert. Therefore, this interface allows a domain expert to easily run state-of-the-art NCD and clustering algorithms on tabular data. Without writing any code, and with minimal knowledge of data science, interpretable results can be generated. This work was presented in a demo paper at an international conference:

[24] **An Interactive Interface for Novel Class Discovery in Tabular Data.** Colin Troisemaine, Joachim Flocon-Cholet, Stéphane Gosselin, Alexandre Reiffers-Masson, Sandrine Vaton and Vincent Lemaire. In: Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), Applied Data Science and Demo Track, 2023.

In Chapter 6, we validate the methods developed in the previous chapters on real operational faults of *Orange*'s FTTH network. We start by detailing the data wrangling process that had to be followed to transform the raw data used by DELC into a format usable by machine learning algorithms. In the experiments, 7 clustering and NCD algorithms are evaluated against each other on this dataset. The results are presented in terms of performance metrics and with a confusion matrix. Finally, we give an example of a decision tree that can be automatically derived from the discovered clusters, and

compare it to the true rules in DELC's software.

Finally, Chapter 7 summarizes the contributions made throughout the dissertation, highlights their limitations and explores potential future research directions. In these future research directions, we discuss promising topics such as cluster interpretation, human-in-the-loop, Generalized Category Discovery (GCD) and hyperbolic neural networks for hierarchical data representation.

The chapters are organized around the individual papers that have been produced during the course of this thesis, reflecting the scientific approach that has been taken. Each chapter is designed to be self-contained, allowing for independant understanding without the need to read the others. As a result, there may be repetitions, especially in the introductions and related works of Chapters 3 and 4. We recommend that the reader follow the chapters in their intended order and skip those sections that may be redundant.

Chapter 2

# Novel Class Discovery: an Introduction and Key Concepts

## Contents




This chapter is currently under review at an international journal and is available as a preprint: [20] Colin Troisemaine, Vincent Lemaire, Stéphane Gosselin, Alexandre Reiffers-Masson, Joachim Flocon-Cholet and Sandrine Vaton. Novel Class Discovery: an Introduction and Key Concepts. In: *ArXiv preprints*, 2023.

## 2.1 Introduction

In the past decade of machine learning research, many classification models have relied heavily on the availability of large amounts of labeled data for all relevant classes. The recent success of these models is due in part to the abundance of labeled data. However, it is not always possible to have labeled data for all classes of interest, leading researchers to consider scenarios where unlabeled data is available. This "open-world" assumption is becoming increasingly more common in practical applications, where instances outside the initial set of classes may emerge [25]. To illustrate, let's examine the scenario of Figure 2.1. Here, instances from classes never seen during training appear at test time. An ideal model should not only be able to classify the known classes (parrots and cats), but also to discover the new ones (tigers and horses).

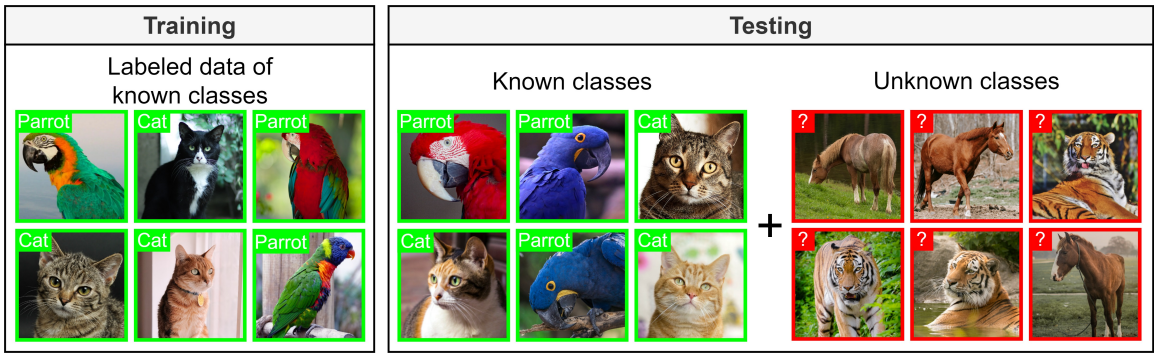



Figure 2.1 – The open-world scenario, where new classes appear during inference.

*What is the issue?* - In this example, a standard classification model is likely to incorrectly classify instances that fall outside the known classes as belonging to one of the known classes. This is a well-known phenomenon of neural networks, where they can produce overconfident incorrect predictions, even in the case of semantically related inputs [26]. Here, a tiger would be classified as a parrot or a cat. For this reason, researchers are now exploring scenarios where unlabeled data is also available [27, 28]. In this chapter, we will focus on one such scenario, where a labeled set of known classes and an unlabeled set of novel classes are given during training. The goal is to learn to categorize the unlabeled data into the appropriate classes. This is referred to as "Novel Class Discovery (NCD)" [1] [29].

1. In this chapter, we use the term "Novel Class Discovery" to refer to the specific domain and not to the *act of discovering novel classes*. This name is becoming gradually more popular in the literature, but it can be confusing due to its general meaning. It is also sometimes called "Novel Category Discovery".

*What is the usual setup of NCD?* - Illustrated in Figure 2.2, the training data in NCD consists of two sets of samples: one from known classes and one from novel classes. The test set is comprised solely of samples from novel classes. The NCD scenario belongs to Weakly Supervised Learning [27, 28], where methods that require all the classes to be known in advance can be distinguished from those that are able to manage classes that have never appeared during training. As an example, in Open-World Learning (OWL) [25], methods seek to accurately label samples of classes seen during training, while identifying samples from novel classes. However, the methods in OWL are generally not tasked with clustering the novel classes and unlabeled data is left unused. Another example is Zero-Shot Learning (ZSL) [30], where the models are designed to accurately predict classes that have never appeared during training. But some kind of description of these novel classes is needed to be able to recognize them. On the other hand, NCD has recently gained significant attention due to its practicality and real-world applications.

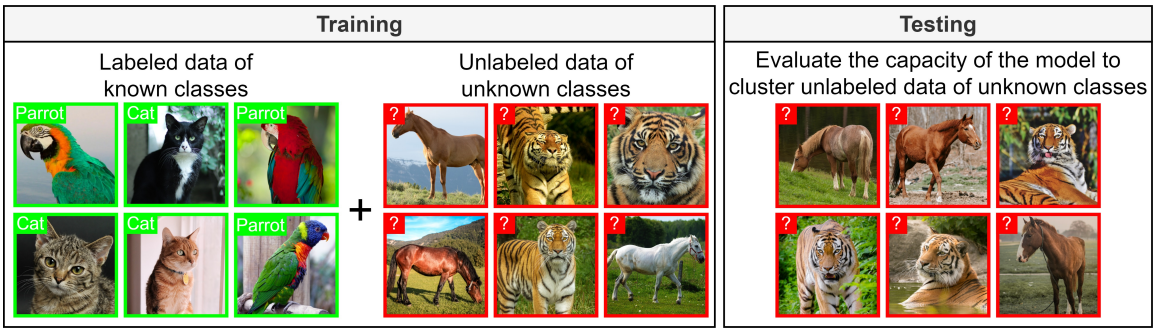



Figure 2.2 – The Novel Class Discovery scenario, where both labeled data of known classes and unlabeled data of novel classes are available during training.

*Why does clustering alone fail to produce good results?* - Albeit naive, unsupervised clustering is a direct solution to the NCD problem as it can sometimes be sufficient for discovering classes in unlabeled data. For example, many clustering methods have obtained an accuracy larger than 90% on the MNIST dataset [31, 32, 33]. But in the case of complex datasets, the literature shows that clustering fails [34, 35] compared to more sophisticated approaches. Clustering can fail for many reasons due to the assumptions that the methods make: spherical clusters, mixture of Gaussian distributions, shape of the data, similarity measure, etc. Thus, the partitioning produced could be incoherent with the data or with the semantic classes; i.e. unsupervised learning is not enough in some cases. We attempt to illustrate this idea in Figure 2.3: If the similarity measure

used is highly influenced by the color of images, the clusters that are generated will likely group images based on their dominant color. Although the clusters formed in this manner will be statistically accurate (with high similarity within the cluster and low similarity between clusters), the semantic categories will not be revealed.


Figure 2.3 – Example of naive solution that could be found with unsupervised clustering. The images are grouped by dominant color and not by semantic class such as bird, flower, fish, ...

As real-world datasets vary widely in nature and the desired clusters can have very different definitions, it seems impossible to create a clustering algorithm that fits all data types. Therefore, there is a need for more refined techniques that can extract from known classes a relevant representation of a class in order to improve the clustering process.

*To fill these gaps* - the Novel Class Discovery domain has been proposed: it attempts to identify new classes in unlabeled data by exploiting prior knowledge from known classes. The idea behind NCD is that by having a set of known classes, a suitable method should be able to improve its performance by extracting a general concept of what constitutes a good class. This can, for example, take the form of a specialized similarity function or a latent space containing domain-specific features. It is assumed that the model does not need to be able to distinguish the known from the novel classes. If this assumption is not made, this becomes a *Generalized Category Discovery* (GCD) [36] problem. Some solutions have been proposed for the NCD problem in the context of computer vision and have displayed promising results [17, 37, 38, 39].

In most of the literature, the difficulty of a NCD problem is set by varying the number of known/novel classes, and increasing the number of known classes is considered as a way of making the problem easier. In [40], the authors explore the influence of the semantic similarity between the classes of the labeled and unlabeled sets. Their assumption is that if the labeled set has a high semantic similarity to the unlabeled set, the NCD problem

will be easier to solve. Intuitively, if the task is to distinguish different animal species in the unlabeled set, a set of other known animals will be beneficial, while a set of cars will not. They prove the validity of this assumption through their experiments and find that a labeled set with low semantic similarity can even have a negative impact on the performance.

*Contributions and organization of this chapter* - We provide a detailed overview of Novel Class Discovery and its formulation, as well as its positioning with respect to related domains. We outline the key components present in most NCD methods, in the form of general workflows and a study of some representative methods, organized by the way they transfer knowledge from the labeled to the unlabeled set. Additionally, we situate related works in the context of NCD. The remaining sections of this chapter are organized as follows: Section 2.2 introduces relevant general knowledge and an overview of domains related to NCD. Section 2.3 presents a taxonomy of current NCD methods and describes some representative methods. Section 2.4 provides a brief overview of new domains derived from NCD. Since certain techniques and tools are frequently found in NCD methods, Section 2.5 offers a concise description of them. Finally, Section 2.6 highlights links and differences with related research fields before concluding.

## 2.2 Preliminaries

In this section, we introduce some general knowledge useful to understand most of the NCD works. We start by briefly summarizing the history of NCD in the literature, before giving a formal definition that follows the widely used mathematical notations of [39] and [41]. And we present the usual evaluation protocol and the metrics used in NCD. Please note that the notations that will be utilized throughout this chapter can be found in the unnumbered *Notations* section just before the general introduction of this manuscript.

**A brief history of NCD:** The 2018 article of Hsu et al. [29] can be considered the first to solve the Novel Class Discovery problem. The authors position their work as a transfer learning task where the labels of the target set are not available and must be inferred. Their methods, KCL [29] and MCL [42], are still regularly used as competitors in NCD articles. The term "Novel Category Discovery" was initially used by Han et al. [41] in 2020 and is another popular term to designate the NCD problem. Building on this

work, Zhong et al. defined "Novel Class Discovery" as a new specific setting in 2021 [39].

**A formal definition of NCD:** During training, the data is provided in two distinct sets: a labeled set $D^l = \{(x_i^l, y_i^l)\}_{i=1}^N$ where each sample $x_i^l$ in $X^l \in \mathbb{R}^{d \times N}$ has a corresponding class labels $y_i^l \in \mathcal{Y}^l = \{1, \ldots, C^l\}$. And an unlabeled set $D^u = \{x_i^u\}_{i=1}^M$ where only the samples $x_u^l$ in $X^u \in \mathbb{R}^{d \times M}$ are available. The goal is to use both $D^l$ and $D^u$ to discover the $C^u$ novel classes, and this is usually done by partitioning $D^u$ into $C^u$ clusters and associating labels $y_i^u \in \mathcal{Y}^u = \{1, \ldots, C^u\}$ to the data in $D^u$.

In the specific setup of NCD, there is no overlap between the classes of $\mathcal{Y}^l$ and $\mathcal{Y}^u$, so we have $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$. We are not concerned with the accuracy on the classes of $D^l$, this set is only here to provide a form of knowledge on what constitutes a relevant class. In all the works reviewed in this chapter, the number $C^u$ of novel classes is assumed to be known a priori, although we will see that some works attempt to estimate this number.

**Positioning and key concepts of NCD:** Novel Class Discovery is a nascent problem with a setup that can be challenging to differentiate from other fields. To provide an overview of the domains explored in this chapter, we propose Figure 2.4. By comparing NCD with these related domains and highlighting the key differences, we aim to offer the reader a clear and comprehensive understanding of the NCD domain. Please refer to Section 2.6 for further details and discussions. Note that in Figure 2.4, the domains are differentiated only by their setup, and while they may be similar, they dot not solve exactly the same problems. Additionally, Open-World Learning is reviewed in Section 2.6.4 but does not appear in this figure. This is due to its broad definition and the multitude of domains it encompasses, which would cause it to appear in several branches of Figure 2.4.

**Evaluation protocol and metrics in NCD:** To evaluate a NCD method on a given dataset, the typical procedure [37] is to hold out (or *hide*) during the training phase a portion of the classes from a fully labeled dataset to act as novel classes and form the unlabeled dataset $D^u$. For example, in most articles evaluated on MNIST, the authors consider the first 5 digits as known classes and the last 5 as novel classes whose labels are not used during training. The performance metrics are only computed on $D^u$, as NCD is only concerned with the performance on the novel classes.

The primary metric used to evaluate the performance of models in NCD is the clustering accuracy (ACC). First introduced by [1], it requires to optimally map the predicted

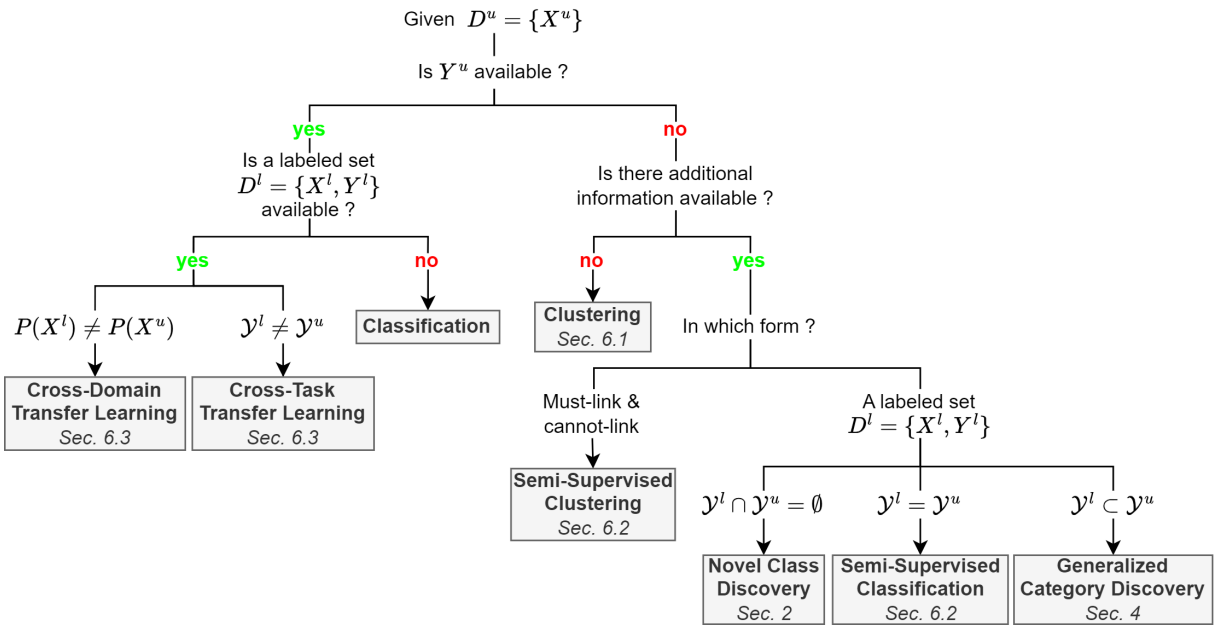



Figure 2.4 – Overview of the domains related to Novel Class Discovery.

labels to the ground truth labels, as the cluster numbers will not necessarily match the class numbers. The mapping can be obtained with the Hungarian algorithm [2] (also known as the Kuhn-Munkres algorithm). The ACC is defined as:

$$ACC = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}[y_i^u = \text{map}(\hat{y}_i^u)] \tag{2.1}$$

where $\text{map}(\hat{y}_i^u)$ is the mapping of the predicted label for sample $x_i^u$ and $M$ is the number of samples in the unlabeled set $D^u$.

Another popular metric is the normalized mutual information (NMI). It measures the correspondence between the predicted and ground truth labels and is invariant to permutations. It is defined as:

$$NMI = \frac{I(\hat{y}^u, y^u)}{\sqrt{H(\hat{y}^u)H(y^u)}} \tag{2.2}$$

where $I(\hat{y}^u, y^u)$ is the mutual information between $\hat{y}^u$ and $y^u$ and $H(y^u)$ and $H(\hat{y}^u)$ are the marginal entropies of the empirical distributions of $y^u$ and $\hat{y}^u$ respectively.

Both metrics range between 0 and 1, with values closer to 1 indicating a better agreement to the ground truth labels. Other metrics that can be found in NCD articles include

the Balanced Accuracy (BACC) and the Adjusted Rand Index (ARI). In the case of imbalanced class distribution, the BACC provides a more representative evaluation of the performance of a model compared to the simple accuracy. It is calculated as the average of sensitivity and specificity. And the ARI gives a normalized measure of agreement between the predicted clusters and the ground truth. Unlike the other metrics, it ranges from -1 to 1, with higher values also indicating better agreement between the two clusterings. A score of 0 indicates random clustering, while negative scores indicate a performance worse than random.

## 2.3 Taxonomy of Novel Class Discovery methods

Table 2.1 – Main contributions of the works in NCD, organized by the method of knowledge transfer from $D^l$ to $D^u$.

| **Knowledge transfer method** | | **Article** | **Main contributions** |
|---|---|---|---|
| Two-stage methods | Similarity function learned on $D^l$ | CCN [29] | The first article to define and solve the NCD problem. |
| | | MCL [42] | Improvement of [29] and introduction of the modified binary cross-entropy with inner product. |
| | Latent space learned on $D^l$ | DTC [37] | Adaptation of a deep clustering method [43] for NCD. |
| | | MM/MP [19] | Formalization of the assumptions behind NCD. Solving NCD with a limited quantity of unlabeled data. |
| One-stage methods | Joint objective on $D^l$ and $D^u$ | AutoNovel [17, 41] | Using SSL to pre-train using all the data. The RankStats method for pseudo-labeling. Joint objective of classification on $D^l$ and clustering on $D^u$. |
| | | CD-KNet-Exp [38] | Using the Hilbert Schmidt Independence Criterion to bridge supervised and unsupervised information. |
| | | *Unnamed* [44] | Insertion of the pre-training objective in the joint loss. |
| | | OpenMix [45] | Creating synthetic samples with mixed known and novel classes to produce robust pseudo-labels. |
| | | NCL [39] | Adapting contrastive learning to the NCD setting, along with NCD-specific hard-negative generation. |
| | | WTA [46] | A solution for NCD in multi-modal video data, using WTA hashing [47] for pseudo-labeling. |
| | | DualRS [48] | Automatic extraction of both global and local features of images to define robust pseudo-labels. |
| | | Spacing loss [49] | Learning an easily separable representation with spaced-out spherical clusters. |
| | | TabularNCD [21] | Solving the NCD problem for tabular datasets. |

In this section, NCD works are organized by the way in which they transfer knowledge from the labeled set $D^l$ to the unlabeled set $D^u$. Also identified by [50], and [49], NCD

methods adopt either a *one-* or *two-stage* approach. An overview of the methods that are studied in this section is provided in Table 2.1, along with a brief description of their contributions.

The first NCD works published were generally two-stage approaches, so they are described here first. They tackle the NCD problem in a way similar to cross-task Transfer Learning (TL) methods. They first focus on $D^l$ only (like a source dataset in TL) before exploring $D^u$ (similarly to a target dataset without labels in TL). Within this category, two families of methods can be distinguished: one uses $D^l$ to learn a similarity function, while the other incorporates the features relevant to the classes of $D^l$ into a latent representation.

More recent methods adopt one-stage approaches and process $D^l$ and $D^u$ simultaneously through a shared objective function. All the one-stage methods reviewed here work in a similar manner, where a latent space shared by $D^l$ and $D^u$ is trained by two classification networks with different objectives. These objectives usually include clustering the unlabeled data and maintaining good classification accuracy on the labeled data.

### 2.3.1 Two-stage methods

#### Learned-similarity–based

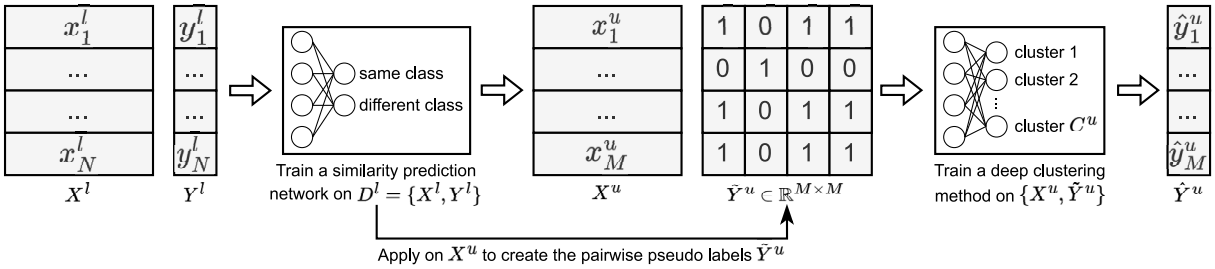



Figure 2.5 – General workflow of learned-similarity–based methods.

The general workflow of learned-similarity–based methods is illustrated in Figure 2.5. Learned-similarity–based methods start by learning on $D^l$ a function that is also applicable on $D^u$ and determines if pairs of instances belong to the same class or not. As the numbers $C^l$ and $C^u$ of classes can be different, a *binary* classification network is generally trained by deriving supervised pairwise labels from the existing class labels $Y^l$. The learned binary classifier is then applied on each unique pair of instances in the unlabeled set $D^u = \{X^u\}$

to form a pairwise pseudo-label matrix $\tilde{Y}^u$. This matrix is used as a target to train a classifier on $D^u$ and make the final class prediction.

In this section, we review two of the main learned-similarity—based methods of the literature. CCN [29] is the first to tackle the very specific problem of NCD, and MCL [42] makes improvements to CCN and defines a loss function used in many subsequent NCD works.

• **Constrained Clustering Network (CCN)** [29] tackles the cross-domain Transfer Learning (TL) problem which is outside of the scope of this review, as well as a cross-task TL problem that corresponds to NCD. In the latter, the method seeks to cluster $D^u$ by using the knowledge of a network trained on $D^l$. In the first stage, a similarity prediction network is trained on $D^l$ to distinguish if pairs of instances belong to the same class or not. This network is then applied on $D^u$ to create a matrix of pairwise pseudo-labels $\tilde{Y}^u$ (similarly to must-link and cannot-link constraints). In the second stage, a new classification network is defined with $C^u$ output neurons with the objective of partitioning $D^u$. It is trained on $D^u$ by comparing the previously defined pseudo-labels to the KL-divergence between pairs of its cluster assignments. In other words, if for two samples $x_i$ and $x_j$ the value in the pseudo-labels matrix is 1 (i.e. $\tilde{Y}^u_{i,j} = 1$), the two cluster assignments of the classification network must match according to the KL-divergence. The idea behind this approach is that if a pair of instances is similar, then their output distribution should be similar (and vice-versa), resulting in clusters of similar instances according to the similarity network.

• **Meta Classification Likelihood (MCL)** [42] is a continuation of CCN [29] by the same authors. They also consider multiple scenarios, one of them being "unsupervised cross-task transfer learning", which corresponds to the NCD setting. Similarly to CCN [29], pairwise pseudo-labels are constructed on $D^u$ by a similarity prediction network trained on $D^l$. A classification network with $C^u$ output neurons is also defined to partition $D^u$. But this time, the KL-divergence is not used to determine if two instances were assigned to the same class. Instead, they use the inner product of the prediction $p_{i,j} = \hat{y}_i^T \cdot \hat{y}_j$. This $p_{i,j}$ will be close to 1 when the predicted distributions $\hat{y}_i$ and $\hat{y}_j$ are sharply peaked at the same output node and close to 0 otherwise. This is a simple yet effective idea that can be directly compared to the pairwise pseudo-labels $\tilde{y}_{i,j} \in \{0, 1\}$

and enables the use of the usual binary cross-entropy (BCE) as a loss function:

$$L_{BCE} = -\sum_{i,j} \tilde{y}_{i,j} \log(\hat{y}_i^T \cdot \hat{y}_j) + (1 - \tilde{y}_{i,j}) \log(1 - \hat{y}_i^T \cdot \hat{y}_j) \tag{2.3}$$

This is an important formalization of the classification problem with pairwise labels that has been used in many subsequent NCD papers.

### Latent-space–based

The general workflow of latent-space–based methods is illustrated in Figure 2.6. These methods start by training with $D^l = \{X^l, Y^l\}$ a latent representation that incorporates the important characteristics of the known classes $\mathcal{Y}^l$. This is usually done by defining a deep classifier with several hidden layers. After training with cross-entropy, the output and softmax layers are discarded, and the last hidden layer is now regarded as the output of an encoder. These methods make the assumption that the high-level features of the known classes are shared by the novel classes. As the latent space highlights these features, $X^u$ is then projected inside, and any off-the-shelf clustering method can be applied to discover the novel classes.

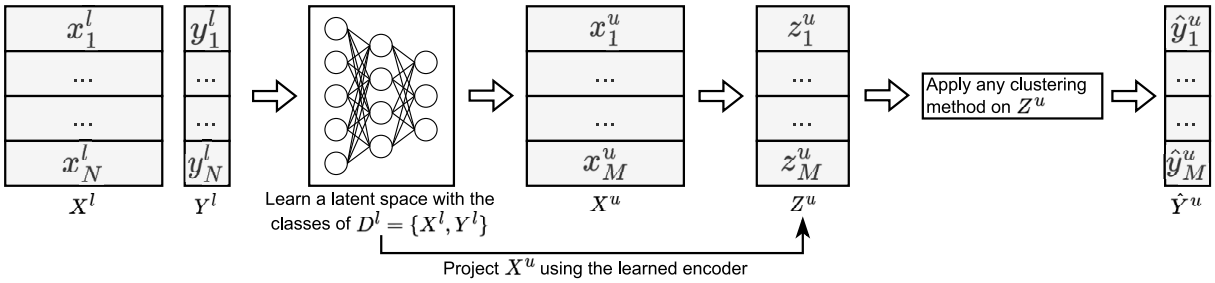



Figure 2.6 – General workflow of latent-space–based methods

Two relevant latent-space–based methods are summarized below. DTC [37] extends to the NCD setting a deep clustering method, which is very suitable for the NCD problem. MM [19] formalizes the assumptions behind NCD and proposes to train a set of expert classifiers to cluster the unlabeled data.

• **Deep Transfer Clustering (DTC)** [37] is based on an unsupervised deep clustering method, DEC [43], which clusters the data while learning a good representation at the same time. Unlike many deep clustering methods, DEC does not rely on pairwise

pseudo-labels. Instead, it maintains a list of class prototypes that represent the cluster centers and assigns instances to the closest prototype. To adapt DEC to the NCD setting, DTC initializes a representation by training a classifier with cross-entropy on $D^l$ using the ground truth labels. The embedding of $D^u$ is then obtained by projecting through the classifier whose last layer was removed. An intuitive conclusion for DEC is that if the classes $Y^l$ and $Y^u$ share similar semantic features, DEC should perform better on the embedding of $D^u$ produced this way.

After projection of $D^u$, DTC applies DEC with some improvements. Namely, the clusters are slowly annealed to prevent collapsing the representation to the closest cluster centers, and they find that further reducing the dimension of the learned representation with Principal Component Analysis (PCA) leads to an improved performance.

• **Meta Discovery with MAML (MM)** [19] proposes a new method along with theoretical contributions to the field of NCD, by defining a set of conditions that must be met so that NCD is theoretically solvable. In simple terms, they state that: (1) known and novel classes must be disjoint (2) it must be meaningful to separate observations from $X^l$ and $X^u$ (3) good high-level features must exist for $X^l$ or $X^u$ and based on these features, it must be easy to separate $X^l$ or $X^u$ (4) these high-level features are shared by $X^l$ and $X^u$. These four conditions are worthy of consideration when the NCD problem is addressed for a new dataset. The reader may find more details in the original article.

Based on the assumption that $X^l$ and $X^u$ share high-level features where the partitioning is easy, the authors suggest that it is possible to cluster $D^u$ based on the features learned on $D^l$. Therefore, they propose a two-stage approach that starts by training a number of "expert" classifiers on $D^l$ with a shared feature extractor. These classifiers are constrained to be orthogonal to each other to ensure that they each learn to recognize unique features of the labeled data. The resulting latent space should reveal these high-level features, shared by the labeled and unlabeled data, and should be sufficient to cluster $D^u$. The expert classifiers are then fine-tuned on the unlabeled data $D^u$ with the BCE of Equation (2.3) by defining pseudo-labels based on the similarity of instances in the latent representation learned on $D^l$. The output of the classifiers after fine-tuning is used as the final prediction for the unlabeled data.

This paper also makes experiments given a limited quantity of unlabeled data, and shows that its method is more robust than the competitors in this case.

### 2.3.2 One-stage methods

#### Introduction

The general workflow of one-stage methods is illustrated in Figure 2.7. In opposition to two-stage methods, one-stage methods exploit both sets $D^l$ and $D^u$ simultaneously. Some of these methods still have multiple steps (such as pre-training on $D^l$), but they are characterized by their joint use of $D^l$ and $D^u$ during the clustering phase. Among two-stage approaches, both similarity (see Section 2.3.1.A) and latent-space based (see Section 2.3.1.B) are negatively impacted when the relevant high-level features are not completely shared by the known and novel classes, as shown in [40]. But by handling data from both sets of classes, one-stage methods will inherently obtain a better latent representation less biased towards the known classes.

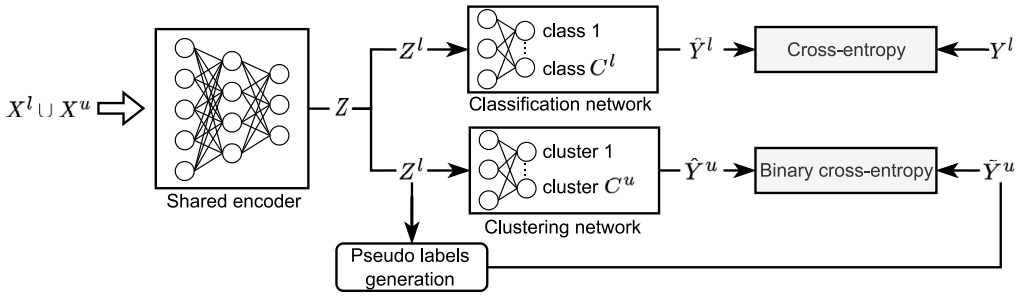



Figure 2.7 – General workflow of one-stage methods. The regularization loss is omitted for the sake of clarity.

Most one-stage methods jointly train two classification networks (see Figure 2.7). One predicts the labels of $D^l$ and introduces the relevant features of the known classes, and the other partitions $D^u$ using pseudo-labels usually defined with similarity measures. By training both networks on the same latent space, they share knowledge with other. In this chapter, the classification network trained on $D^u$ will be referred to as a "clustering" network, since it is trained with unlabeled data.

One-stage methods define a multi-objective loss function which typically has 3 components: cross-entropy ($\mathcal{L}_{CE}$), binary cross-entropy ($\mathcal{L}_{BCE}$) and regularization ($\mathcal{L}_{MSE}$). The cross-entropy loss is simply used to train the classification network with the ground truth labels. The binary cross-entropy loss compares the prediction of the clustering network to pseudo-labels (see Equation (2.3)). And the regularisation loss ensures that the model generalizes to a good solution. This is usually done by encouraging both networks to predict the same class for an instance and its randomly augmented counterpart (see

column "Data Augmentation" in Table 2.2).

While Section 2.3.1 was, to the best of our knowledge, an exhaustive list of the two-stage methods, there is a larger (and fast growing) number of papers that follow a one-stage approach. For this reason, only four methods representative of the literature are first detailed, and a few other methods are described more concisely in the last section.

#### AutoNovel

AutoNovel [17, 41] is the first one-stage method proposed to solve the NCD problem. It introduced the architecture illustrated in Figure 2.7 and inspired many subsequent works [39, 49, 45, 46, 48]. AutoNovel starts by carefully initializing its encoder using the RotNet [51] Self-Supervised Learning (SSL) method to train on both labeled and unlabeled data. As SSL does not leverage the labels of known classes, the learned features will not be biased towards the known classes. At this point, the authors consider that the features learned by the encoder will be representative of all data and will be useful for any given task, so they *freeze* all but the last layer of the encoder. Finally, the labeled data is used to train for a few epochs the classifier and fine-tune the last layer of the encoder. This concludes the initialization of the representation (the shared encoder in Figure 2.7), which is crucial as the next step involves determining pseudo-labels in the latent space based on pairwise similarity measures.

To realize the joint learning on $D^l$ and $D^u$, the two classification networks that can be seen in Figure 2.7 are added on top of the encoder. The three components of the model (shared encoder, classification network and clustering network) are then trained using a loss composed of the three components described in the introduction of this section:

$$\mathcal{L}_{AutoNovel} = \mathcal{L}_{CE} + \mathcal{L}_{BCE} + \mathcal{L}_{MSE} \tag{2.4}$$

As AutoNovel uses the BCE of Equation (2.3), the inner products of the clustering network predictions are compared to the pairwise pseudo-labels defined by their original RankStats (for *ranking statistics*) method (see Section 2.5.2).

#### Class Discovery Kernel Network with Expansion (CD-KNet-Exp)

CD-KNet-Exp [38] is a multi-stage method that constructs a latent representation using $D^l$ and $D^u$ that is suitable, after training, to the discovery of the novel classes by a $k$-means. It starts by pre-training a representation with a "deep" classifier on $D^l$ only.

Since this embedding could be highly biased towards the known classes, and may not generalize well to $D^u$, the representation is then fine-tuned with both $D^l$ and $D^u$. In this second stage, they optimize the following objective:

$$\max_{U,\theta} \mathbb{H}(f_\theta(X), U) + \lambda \mathbb{H}(f_\theta(X^l), Y^l) \tag{2.5}$$

$f$ is the feature extractor (or encoder) of parameter $\theta$. $\mathbb{H}(P, Q)$ is the Hilbert Schmidt Independence Criterion (HSIC). It measures the dependence between distributions $P$ and $Q$. And $U$ is the spectral embedding of $X$. Intuitively, the first term encourages the separation of all classes (old and new) by performing something similar to spectral clustering. And the second term introduces the supervised information from the known classes by maximizing the dependence between the embedding of $X^l$ and its labels $Y^l$.

This second step produces a latent space that should have incorporated the information from both known and novel classes and be easily separable. For this reason, the embedding of the data is finally $f_\theta(X^u)$ partitioned with $k$-means clustering.

### OpenMix

The principle of OpenMix [45] is to exploit the labeled data to generate more robust pseudo-labels for the unlabeled data. It relies on MixUp [52], which is widely used in supervised and semi-supervised learning. As MixUp requires labeled samples for every class of interest, applying it directly on the unlabeled data would still produce unreliable pseudo-labels. Instead, OpenMix generates new training samples by mixing both labeled and unlabeled samples.

First, a latent representation is initialized using the known classes only. Then, a clustering network is defined to discover the new classes using a joint loss on $D^l$ and $D^u$. The model is trained with synthetic data that are a mix of a sample from a *known class* and a sample from a *novel class*. The synthetic data points are generated with MixUp, while the labels are a combination of the ground truth labels of the labeled samples and the pseudo-labels determined using cosine similarity for the unlabeled samples (see Figure 2.8). The authors argue that the overall uncertainty of the resulting pseudo-labels will be reduced, as the labeled counterpart does not belong to any new class and its label distribution is exactly true.

These synthetic labels are compared to the prediction of the model: (i) the classification network predicts the known part and (ii) the clustering network the novel part (see

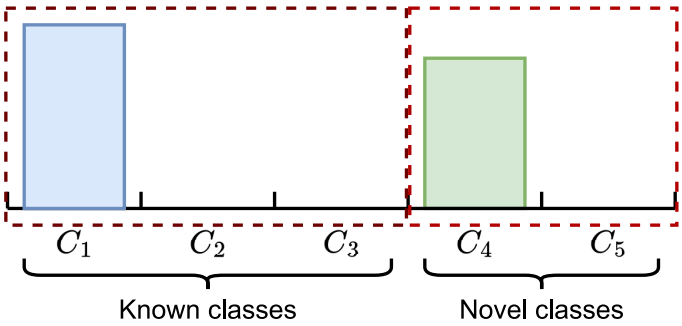



Figure 2.8 – Example of synthetic label generated by Openmix [45]. Here, it is a mix of a labeled sample of class $C_1$ and an unlabeled sample with pseudo-label $C_4$.

Figure 2.7) of the full label space.

The authors observe that the clustering network has good accuracy on the samples that it predicted with high-confidence. Based on this observation, they regard these samples as *reliable anchors* that are further integrated with unlabeled samples to generate even more combinations with MixUp.

### Neighborhood Contrastive Learning (NCL)

NCL [39] is inspired by AutoNovel [17] as it uses the same architecture (see Figure 2.7) and pre-trains its representation in the same way. Its main contribution is the addition of 2 contrastive learning terms to the loss of AutoNovel (see Equation (2.4)) to improve the learning of discriminative representations. The first one is the supervised contrastive learning term from [53] applied to the labeled data using the ground truth labels. The second term is applied on the unlabeled data and adapts the original unsupervised contrastive learning loss to the NCD problem to exploit both labeled and unlabeled data.

For this second term, the authors maintain a queue $M^u$ of samples from past training steps, and consider for any instance in a batch that the $k$ most similar instances from the queue are most likely from the same class. The contrastive loss, for these *positive* pairs is defined for the embedding $z_i^u$ of an instance $x_i^u$ as:

$$l(z_i^u, \rho_k) = -\frac{1}{k} \sum_{\bar{z}_j^u \in \rho_k} \log \frac{e^{\delta(z_i^u, \bar{z}_j^u)/\tau}}{e^{\delta(z_i^u, \hat{z}_i^u)/\tau} + \sum_{m=1}^{|M^u|} e^{\delta(z_i^u, \bar{z}_m^u)/\tau}} \tag{2.6}$$

with $\rho_k$ the $k$ instances most similar to $z_i^u$ in the unlabeled queue $M^u$, $\delta$ the similarity function and $\tau$ a temperature parameter.

Additionally, synthetic positive pairs $(z^u, \hat{z}^u)$ are generated by randomly augmenting

each instance. The contrastive loss for positive pairs is written as:

$$l(z^u, \hat{z}^u) = -\log \frac{e^{\delta(z^u, \hat{z}^u)/\tau}}{e^{\delta(z^u, \hat{z}^u)/\tau} + \sum_{m=1}^{|M^u|} e^{\delta(z^u, \bar{z}_m^u)/\tau}} \tag{2.7}$$

Finally, "hard negatives" are introduced in the queue $M^u$ to further improve the learning process. Hard negatives refer to similar samples that belong to a different class and are an important concept in contrastive learning. Selecting hard negatives in $D^u$ can be difficult since there are no class labels available. Therefore, the authors take advantage of the fact that the classes of $D^l$ and $D^u$ are necessarily disjoint and create new hard negative samples by interpolating easy negatives from the unlabeled set (i.e. instances that are most likely true negatives) with hard negatives from the labeled set.

To summarize, the overall loss that is optimized by the model is:

$$\mathcal{L}_{NCL} = \mathcal{L}_{AutoNovel} + l_{scl} + \alpha l(z_i^u, \rho_k) + (1 - \alpha) l(z^u, \hat{z}^u) \tag{2.8}$$

where $l_{scl}$ is the supervised contrastive loss term for the labeled samples of $D^l$ and $\alpha$ is a trade-off parameter.

### Other methods

We briefly describe a few other one-stage NCD works here. In [44], the SSL objective of RotNet [51] and joint objective of Equation (2.4) are merged in a single loss function. The shared encoder is therefore influenced by the classification network, the clustering network and a linear layer that predicts the random rotations of images. The authors argue that the self-supervised signals will provide a strong regularization that will alleviate the performance degradation caused by the noisy pseudo-labels.

The method proposed in [46] is able to process multi-modal data, composed of both video and audio. Two feature encoders are trained with Noise Contrastive Estimation (NCE) [54], and the latent representations are concatenated before being fed to either a classification or clustering network. The Winner-Take-All hash [47] is used to measure the similarity between each pair of unlabeled samples during the definition of pseudo-labels required to train the clustering network. The authors argue that WTA is more robust to noise and effectively captures the structural relationships among the objects (see [46] for more details).

The Dual Ranking Statistics (DualRS) [48] method trains two framework branches

on a shared latent representation. Both branches have a classifier trained to predict the known classes and a clustering network trained with pseudo-labels and Equation (2.3). One branch is tasked to extract global features, as pseudo-labels are defined by measuring the similarity between *whole* images. The other branch focuses on individual local details, and pairwise similarities are computed using only part of each image. The authors argue that these branches are complementary to each other, as they focus on different granularity of the data. The global branch may easily find similarities and introduce more false positives and have high recall (but low precision), while the local-part branch will be more "strict" and have high precision (but low recall). To make the two branches communicate, agreement between the similarity score distributions of unlabeled data is encouraged.

Similarly to [38], the Spacing Loss [49] method shapes a latent space where the novel classes are easily separable. During training, the representation is slowly guided to have spaced-out clusters that are equidistant to each other. Each epoch alternates between learning with pseudo-labels derived from the closest cluster centers and modifying the cluster centers themselves. During inference, a $k$-means is run in the learned latent representation to discover the novel categories.

Finally, to the best of our knowledge, a single method has attempted to solve NCD in the context of tabular data [21]. It pre-trains a simple encoder of dense layers with the VIME [55] self-supervised learning method and adopts the two heads architecture of Figure 2.7. Similar to other one-stage methods, known classes are classified jointly with clustering on the unlabeled data, and pseudo-labels are defined based on pairwise cosine similarity.

### 2.3.3 Estimating the number of novel classes

The assumption that the number $C^u$ of novel classes in the unlabeled set $D^u$ is known can be unrealistic in some scenarios. For this reason, a few methods were proposed to automatically estimate this number $C^u$.

A method used in [29, 42, 48, 56], consists in setting the number of output neurons of the clustering network to a large number (e.g. 100). In doing so, we rely on the clustering network to use only the necessary number of clusters and to leave the other output neurons unused. Clusters are counted if they contain more instances than a certain threshold. This approach is surprisingly simple, but displays stable results in the different articles that experimented it.

In [36, 57], a $k$-means is performed on the entire dataset $D^l \cup D^u$. The number of novel

classes $C^u$ is estimated to be the $k$ that maximized the Hungarian clustering accuracy (see Section 2.2): a $k$ too high will result in clusters assigned to the null set and a number too low will have clusters composed of multiple classes, both cases will be considered as being assigned incorrectly.

Finally, another popular idea is to make use of the known classes [37, 41, 17, 58]. This process is illustrated in Figure 2.9. The known classes of $D^l$ are first split into a *probe* subset $D^l_r$ and a training subset $D^l \backslash D^l_r$ containing the remaining classes. The set $D^l \backslash D^l_r$ is used for supervised feature representation learning, while the probe set $D^l_r$ is combined with the unlabeled set $D^u$. Now, a constrained $k$-means is run on $D^l_r \cup D^u$. Part of the classes of $D^l_r$ are used for the clusters initialization, while the rest are used to compute 2 cluster quality indices (average clustering accuracy and cluster validity index, see [37]). Note that this can be difficult to use when the number of known classes is small, since it involves many class splits.

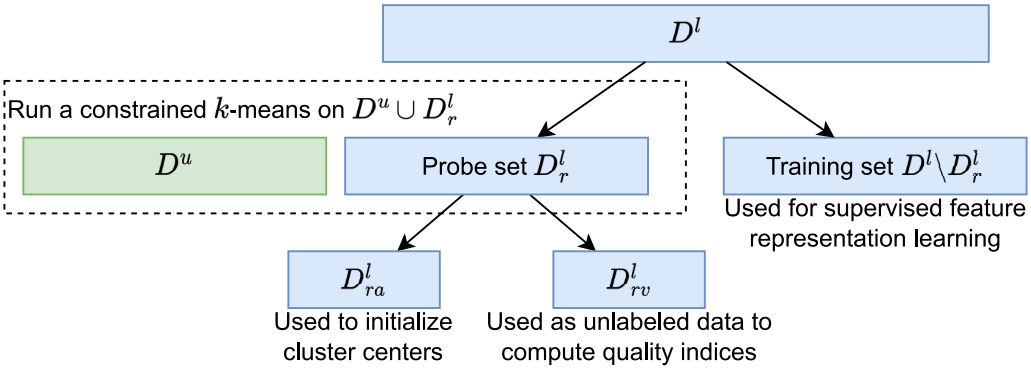



Figure 2.9 – Number of novel classes estimation process from DTC [37].

### 2.3.4 Methods summary

Table 2.2 summarizes the important characteristics of the methods that were described in this section. These characteristics include the type of data processed, the method of defining pairwise pseudo-labels and, if applicable, the method of estimating the number of novel classes $C^u$. From column "Unknown $C^u$", it is evident that all the works reviewed here assume knowledge of the number of novel classes. Moreover, this table highlights the popularity of pairwise pseudo-labeling as a means of training classification networks on unlabeled data, with only DTC [37] and CD-KNet-Exp [38] relying on different processes.

Table 2.2 – Overview of the characteristics of NCD methods.

| | **Method** | **Data Type** | **Backbone architecture** | **Pairwise pseudo-labels** | **Pre-training** | **Data Augmentation** | **Unknown $C_u$** |
|---|---|---|---|---|---|---|---|
| Two-stage methods | CCN [29] | Image | ResNet18 | From learned classifier | ✗ | ✗ | ✗ + Estimated ($k = 100$) |
| | MCL [42] | Image | LeNet, VGG8 and ResNet | From learned classifier | ✗ | Crop and flip | ✗ + Estimated ($k = 100$) |
| | DTC [37] | Image | ResNet18 and VGG | ✗ (class prototypes) | CE on $D^l$ | Crop and flip | ✗ + Estimated (probe classes) |
| | MM/MP [19] | Image | ResNet18 and VGG16 | RankStats [17] | CE on $D^l$ | ✗ | ✗ |
| One-stage methods | AutoNovel [17, 41] | Image | VGG and ResNet18 | RankStats [17] | RotNet [51] on $D^l \cup D^u$ | Crop and flip | ✗ + Estimated (probe classes) |
| | CD-KNet-Exp [38] | Image | Custom CNN | ✗ | CE on $D^l$ | ✗ | ✗ |
| | *Unnamed* [44] | Image | ResNet18 | Threshold on SNE | ✗ | Yes, unspecified | ✗ |
| | OpenMix [45] | Image | VGG and ResNet18 | Threshold cosine similarity | CE on $D^l$ | Crop and flip | ✗ |
| | NCL [39] | Image | ResNet18 | Threshold cosine similarity | RotNet [51] on $D^l \cup D^u$ | Crop and flip | ✗ |
| | WTA [46] | Image & Video | R3D-18 and ResNet18 | WTA hash [47] | ✗ | Crop, resize, flip, color distortion and blur | ✗ |
| | DualRS [48] | Image | RestNet18 | Dual ranking statistics | RotNet [51] on $D^l \cup D^u$ | Crop and flip | ✗ + method from DTC |
| | Spacing Loss [49] | Image | ResNet18 | Threshold cosine sim. + class prototypes | CE on $D^l$ | Crop and flip | ✗ |
| | TabularNCD [21] | Tabular | Custom DNN | Number of most similar | VIME [55] on $D^l \cup D^u$ | SMOTE [59] | ✗ |

## 2.4 New domains derived from NCD

As the number of NCD works increases, new domains closely related to it are emerging. Researchers are designing scenarios where they relax some of the hypotheses or define new tasks inspired by NCD. This section will provide a brief overview of some of the most important of these domains. Given their similarity in settings, Table 2.3 highlights some of the key differences among them.

Table 2.3 – Distinctions between the related domains.

| | NCD | GCD | NCDwF |
|---|---|---|---|
| test data $\in \mathcal{Y}^l \cup \mathcal{Y}^u$ | ✗ | ✓ | ✓ |
| $D^l$ and $D^u$ are available simultaneously | ✓ | ✓ | ✗ |

**Generalized Category Discovery (GCD)** [36] is a setting that is gaining traction from the community, with some very recent articles published [36, 58, 60, 57]. GCD was designed to be a less constrained and more realistic setting of Novel Class Discovery, as it does not assume that samples during inference will only belong to the novel classes. As the test data can belong to either known or novel classes, the task at inference becomes to (i) accurately classify samples from known classes and (ii) find the clusters of samples from novel classes. Compared to NCD, this poses a greater challenge for designing an efficient model. Methods in this domain are thus evaluated for both their classification and clustering performance. Note that this setting is close to Open-World Learning, but still different as the training data is still composed of two separate sets ($D^l$ and $D^u$).

This problem was first solved in 2021 by [61], but it was not immediately recognized as a setting distinct from NCD. Later, as multiple articles were published simultaneously, different names were used and problem was presented in varying ways. Some of these names include *Generalized Novel Class Discovery* [60], *Open Set Domain Adaptation* [62] and *Open-World Semi-Supervised Learning* [63], however, they all ultimately aimed to solve the same task.

In the first article that formalizes the GCD problem [36], the authors find that existing NCD methods are prone to overfitting on the known classes. Instead of using a parametric classifier, which was seemingly the cause of the overfitting, they use contrastive learning and a semi-supervised $k$-means to recognize images.

Another method of interest is XCon [57]. In this case, the authors focus on fine-grained Generalized Category Discovery, where different classes have very close high-level

features (e.g. two different species of birds where only the beak is different). They propose to partition the data into $k$ sub-datasets that share irrelevant cues (e.g. background and object pose) to force the method to focus on important discriminative information.

**Note:** GCD and its links to Open-World Learning are discussed in Section 2.6.4.

**Novel Class Discovery without Forgetting** (NCDwF) [64] is another domain that relaxes some of the assumptions behind NCD. In NCDwF, $D^l$ and $D^u$ are not available simultaneously. Instead, during training, we are first given $D^l$ to train the standard supervised task of discriminating known classes. Then, $D^l$ becomes unavailable and we are given $D^u$ with the goal of discovering the novel classes. At inference time, the learned model is evaluated for its performance on instances from a mix known and novel classes. This task also poses a greater challenge than NCD as it needs to recognize instances from the full class distribution $\mathcal{Y}^l \cup \mathcal{Y}^u$. And it is more challenging than GCD as the two training sets $D^l$ and $D^u$ are not available at the same time. This means that the partitioning of $D^u$ must be learned while avoiding *catastrophic forgetting* on known classes (hence the name). This domain can be applied if, for example, a model that was previously trained to identify some classes in a dataset that is no longer accessible, and we need to detect new classes while maintaining accuracy on the previously learned categories.

ResTune [50] is the first to solve NCDwF. This article examines three distinct test cases, with NCD and NCDwF among them. This two-stage method starts with pre-training using the labeled data $D^l$ and a simple cross-entropy loss. Then, during the training on $D^u$ only, the previously learned representation and classifier are frozen to avoid both forgetting of known classes and overfitting on the unlabeled data. The partitioning is done by adapting DEC [43] to the NCDwF setting.

In [65], this problem is referred to as *class-incremental novel class discovery* (class-iNCD). Given the NCDwF setting, a two-stage method that seeks to define a classifier capable of predicting in the full label space $\mathcal{Y}^l \cup \mathcal{Y}^u$ is proposed. Similarly to ResTune [50], an encoder and a classifier are first trained with supervision on the labeled set $D^l$. Then, during the exploration of the unlabeled set $D^u$, the previously learned classifier is extended with $C^u$ new output neurons. Additionally, a classification network is added on the shared latent space to partition the unlabeled samples. It is trained with the unsupervised BCE objective of Equation (2.3) and pseudo-labels defined by the RankStats method [17]. The classes predicted by this network are used as targets for the full classification network.

Finally, [64] introduces the name NCDwF. To avoid the forgetting, it proposes a

method to generate synthetic samples that are representative of each known class and act as a proxy for the no longer available labeled data. Furthermore, the authors propose a mutual-information based regularizer which improves the partitioning of novel categories, and a Known Class Identifier that helps generalize inference when the test data includes instances from both known and novel classes.

**Novel Class Discovery in Semantic Segmentation** (NCDSS) is a task defined in [66] which consists in segmenting images that contain novel classes, given a set of labeled images with known foreground and background classes. Since the pixels of multiple categories within a single image must be correctly classified, it is more challenging than NCD. Similarly to NCD, the condition that $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$ is respected, meaning that no image in the unlabeled set contains an object from the known classes. The framework they propose has three stages: base training, clustering with pseudo-labels, and novel fine-tuning. In the base training stage, the model is trained with labeled base data, which is then used in novel images to filter out salient base pixels and assign base labels. In the clustering stage, novel images are fed into the model to obtain novel foreground pixels, which are then used for clustering and assigning novel labels. To address the issue of noisy clustering pseudo-labels, an Entropy-based Uncertainty Modeling and Self-training (EUMS) framework is proposed to improve the novel fine-tuning stage by dynamically splitting and reassigning novel data into clean and unclean parts based on entropy ranking.

## 2.5 Tools for Novel Class Discovery

Some specific learning paradigms are often found in NCD works. Namely: (i) Self-Supervised Learning (SSL) is a popular approach for initializing an encoder, (ii) Pairwise pseudo-labels are used in almost all NCD methods to provide a weak form of supervision for classification neural networks, and (iii) contrastive learning has been employed by some to construct meaningful and discriminative representations. In this section, these 3 key paradigms to design NCD methods are presented and discussed.

### 2.5.1 Self-Supervised Learning

As illustrated in Table 2.2, many methods rely on similarity measures in the latent space to define pairwise relationships between unlabeled instances. To avoid measuring

the similarity after projection through an encoder that was randomly initialized, some methods train for a few epochs with cross-entropy on the labeled samples only. However, this could result in features that are highly biased towards the labeled data and that poorly represent the novel classes. Instead, recent methods have taken advantage of Self-Supervised Learning (SSL) to bootstrap their latent representation.

SSL is a technique that is widely used in computer vision and natural language processing. The general idea behind SSL methods is to define *pretext tasks* that do not require labels. A pretext task is a fake problem that can be defined depending on the type of data that is used. For example, predicting the angle of rotation of an image [51], re-coloring [67] and completing masked words in sentences [68] are common pretext tasks. Intuitively, SSL allows the model to exploit larger amounts of data by using both labeled and unlabeled data. The model pre-trained this way will be able to extract more interesting properties, subtle patterns and less common representations of the data, resulting in improved performance compared to solely relying on labeled data.

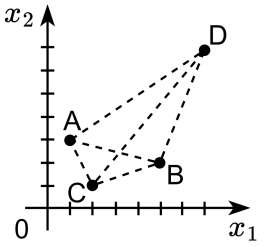
In the context of Novel Class Discovery, SSL allows the model to learn a robust representation that is not biased towards the known classes, as all of the data (labeled and unlabeled) is used. Among SSL methods, RotNet [51] has been a popular choice in NCD works [17, 39, 48]. It is a simple and efficient method where the network must predict the rotation angle, from 0, 90, 180 or 270 degrees, applied to an image. DINO (for *self-**di**stillation with **no** labels*) [69] has also been used in the context of GCD [36]. It employs a self-distillation scheme where a student network learns from a teacher given different crops of the same image. It is a powerful method for vision transformers that produces feature representations where similar objects are close to each other, which is ideal for NCD applications. Finally, VIME (for ***v**alue **i**mputation and **m**ask **e**stimation*) [55] has been used by TabularNCD [21] to pre-train dense layers in the context of tabular data by reconstructing corrupted samples. However, as SSL still struggles to be applied to domains such as tabular data, it has only marginally improved performance. This is partly due to the fact that SSL methods rely heavily on the spatial and semantic structure of image or language data to design pretext tasks. Thus, only a few works have been proposed to deal with heterogeneous data [55, 70, 71].

### 2.5.2 Pseudo-labels

Pseudo-labeling is a technique that provides "weak" labels for unlabeled data. It is particularly useful to exploit large amounts of unlabeled data with models that require

a target to be trained. Apart from NCD, pseudo-labels (sometimes called *soft labels*) are found in other domains, such as Semi-Supervised Learning where unlabeled samples that were predicted with high confidence are added to the training data [72]. In Deep Clustering, they are used to iteratively refine a latent representation by predicting these labels [33, 73].

As expressed in Section 2.3, most NCD methods define *pairwise* pseudo-labels to represent the relationships between pairs of instances in the unlabeled set $D^u$. In the case of learned-similarity–based NCD methods, they are a way of directly transferring knowledge from the known classes (see Section 2.3.1). For the rest, pairwise pseudo-labels are defined and used in a manner similar to Deep Clustering methods, where they provide supervision for a classification network tasked to partition the unlabeled data [2]. Instead of directly assigning class labels to instances, the model is only tasked to predict the same label for "positive" relations and a different label for "negative" relations. This conversion to a different task is called *problem reduction* [74]. It is considered as a less complex problem to solve and to have a lower cost to collect the target. All pseudo-labeling techniques that rely on a similarity measure make the assumption that instances close to each other (usually in the latent space) are likely to belong to the same class. Pairwise pseudo-labels are defined in $\{0, 1\}$ and can be compared for example to the inner product of the prediction through the binary cross-entropy (see Equation (2.3)).




(a) Representation of the data points of the batch.

| | A | B | C | D |
|---|---|---|---|---|
| A | 1 | 0.59 | 0.78 | 0.28 |
| B | 0.59 | 1 | 0.68 | 0.46 |
| C | 0.78 | 0.68 | 1 | 0.22 |
| D | 0.28 | 0.46 | 0.22 | 1 |

(b) Pairwise similarity matrix.

| | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 1 | 1 | 0 |
| B | 1 | 0 | 1 | 0 |
| C | 1 | 1 | 0 | 0 |
| D | 0 | 0 | 0 | 0 |

(c) Pairwise pseudo-labels matrix for $\lambda = 0.5$.

Figure 2.10 – The pairwise pseudo-labels definition process.

To aid the reader in his understanding, Figure 2.10 illustrates a simple pseudo-labeling process employed by OpenMix [45] and NCL [39]. Given a pair $(x_i^u, x_j^u)$ in a batch (Figure 2.10(a)), the latent representation $(z_i^u, z_j^u)$ is extracted and their cosine similarity

2. As this classification network is trained on unlabeled data using these pseudo-labels, it is referred to as a "clustering" network instead.

$\delta(z_i^u, z_j^u) = z_i^u \cdot z_j^u / \|z_i^u\| \|z_j^u\|$ is computed (Figure 2.10(b)). To use this pairwise similarity matrix as a target for the classification network, it needs to be binarized. And a solution is to set a threshold $\lambda$ for the minimum similarity score required to consider two instances as belonging to the same class (Figure 2.10(c)). So in this case, the pseudo-labels are defined as:

$$\tilde{y}_{ij} = \mathbb{1}[\delta(z_i^u, z_j^u) \geq \lambda] \tag{2.9}$$

Note that OpenMix sets $\lambda$ to 0.9 and NCL uses 0.95 arbitrarily, but this is a hyperparameter that can be optimized.

Figure 2.11 depicts this pseudo-labeling process on a real dataset. We take the first 5 classes of MNIST and project them using t-SNE, which allows us to easily distinguish the classes. On the left side, the points are colored according to their cosine similarity to a randomly selected point (the blue triangle). And on the right side, we define pseudo-labels using Equation 2.9 with $\lambda = 0.70$ (green = positive and red = negative). This is a case of a successful use of Equation 2.9, as almost all positive pairs are in the same class. However, this method may not consistently produce such good results, especially for the points at the edge of a class.

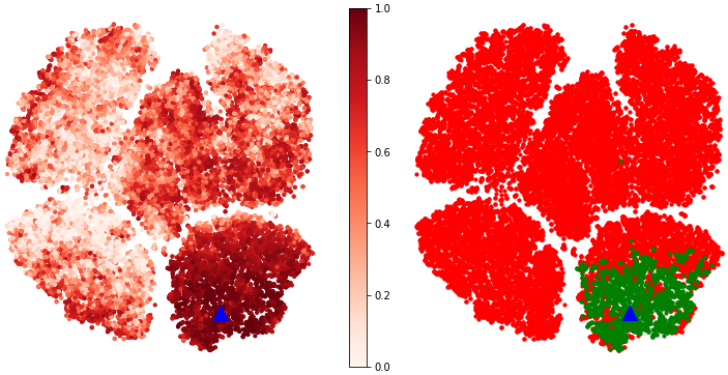



Figure 2.11 – Left: cosine similarity to a random point (the blue triangle). Right: pseudo-labels.

In the remainder of this section, some of the most commonly used pseudo-labeling techniques are introduced.

RankStats (for *ranking statistics*) is a pseudo-labeling approach introduced in AutoNovel [17]. Instead of computing a scalar product or a difference between vectors, a pair of instances is considered similar if their features that were "most activated" by the encoder are the same. The authors argue that the most discriminative features of an image should have the highest values after projection. Thus, RankStats tests whether the $k$

highest values of a pair of embeddings are in the same locations:

$$\tilde{y}_{ij} = \mathbb{1}[\text{top}_k(z_i^u) = \text{top}_k(z_j^u)] \tag{2.10}$$

$\text{top}_k$ is a function that returns the indices of the $k$ largest values in a vector. The order of the most activated features is not required to be the same. It must only contain the same *set* of indices, making RankStats more robust to discrepancies among the most discriminative features.

In [46], the Winner-Take-All (WTA) hash [47] is used to compare pairs of instances. WTA is an embedding method that maps vectors to integer codes. In more detail, the projection $z_i^u$ of an instance $x_i^u$ is randomly permuted, and the index of the largest elements in its $k$ first values is recorded in $c_i^h$. This process is repeated $H$ times for each sample $z_i^u$ to form the WTA hash code $c_i = (c_i^1, \ldots, c_i^h, \ldots, c_i^H)$. Samples are then compared by applying the same set of permutations and counting the number of indices equal to each other:

$$\tilde{y}_{ij} = \mathbb{1}[\mathbf{1}^T \cdot (c_i = c_j) \geq \mu] \tag{2.11}$$

with $\mu$ a threshold. For reference, in [46], $H$ is set to the size of the embedding (512), $\mu$ is selected empirically to be 240 and $k = 4$.

Intuitively, WTA considers many different orders of features, avoiding the comparison to be dominated by high frequency noise or small local regions that are highly activated. Replacing the RankStats pseudo-labeling method in AutoNovel [17] with WTA shows only marginal improvements. But for the NCD method proposed by the authors in [46], WTA consistently outperforms other alternatives, such as RankStats, cosine similarity or nearest neighbour.

Lastly, the quality of the pseudo-labels has been explored in some articles. It is often expressed that they can be noisy and unreliable, and as they have a strong influence on the clustering performance, some works have approached this problem. OpenMix [45] mixes labeled and unlabeled samples with MixUp [52] to generate higher confidence pseudo-labels. DualRS [48] focuses on multiple granularity of image crops to improve reliability. And [44] proposes utilizing local structure information in the feature space to construct pairwise pseudo-labels, as they are more robust against noise.

### 2.5.3 Contrastive Learning

Contrastive Learning [75, 76] is a self-supervised representation learning technique where the objective is to learn a robust representation. This is done by pulling together similar samples and pushing apart dissimilar samples. As labels are not available, a positive pair is usually formed of a sample and its augmented counterpart, while negative pairs are formed with the rest of the data.

Contrastive learning can be easily adapted to take into account labeled samples and to produce even higher quality discriminative representations [53]. For these reasons, it is an ideal technique for the task of Novel Class Discovery, and some NCD works have already used contrastive terms. For instance, NCL [39] adapts the contrastive loss to exploit both the labeled and the unlabeled sets into one holistic framework. Detailed in Section 2.3.2, their overall loss function is composed of (i) the loss of AutoNovel [17] to partition the unlabeled data and (ii) two contrastive terms. The first is the supervised contrastive loss [53] applied to the labeled data, and the second is the unsupervised contrastive loss for the unlabeled data. Their method outperforms all other baselines in the comparison, and they show that the contrastive terms help improve the discrimination of the model.

The Noise-Contrastive Estimation (NCE) [54], has been employed by the WTA-based NCD method of [46]. It is a parameter estimation method initially designed to be an alternative to the expensive softmax function. Instead of computing the prediction of the model for every class, only the true class and a few other (called *noisy*) classes have to be estimated. This principle inspired the supervised contrastive loss [53], and it is employed in the NCD method of [46]. Given a batch of size $n$ and the projection $z_i$ of an instance $x_i$, [46] defines the following loss:

$$\mathcal{L}_{NCE} = -\log \frac{\exp(z_i \cdot \hat{z}_i/\tau)}{\sum_n \mathbb{1}[n \neq i] \exp(z_i \cdot z_n/\tau)} \tag{2.12}$$

where $\hat{z}_i$ is the augmented counterpart of $z_i$, $\mathbb{1}[n \neq i]$ is an indicator function evaluating to 1 *iff* $n \neq i$ and $\tau$ is a temperature parameter. Note that since the projection $z$ is $\ell_2$-normalized, the cosine similarity can be simplified to the inner product. In the case of the NCD method of [46], this NCE loss is used to maintain a latent representation. Similarly to NCL [39], the unlabeled data has positive pairs formed by a sample and its augmented counterpart, while negative pairs are formed with all other samples in the batch. However, compared to [46], NCL reports higher accuracy on the CIFAR-100 and ImageNet datasets. This could be attributed to the fact that NCL defines additional positive pairs by selecting

the most similar pairs in a queue of samples.

OpenCon [77] is a method proposed for the Generalized Category Discovery problem, where the authors employ class prototypes to separate known and novel classes. All instances are assigned to their closest prototype, which allows the definition of a set of pseudo-positives $\mathcal{P}(x)$ and pseudo-negatives $\mathcal{N}(x)$ for each instance $x$. In conventional unsupervised contrastive learning frameworks, only the augmented counterpart of an instance is used to form a positive pair. In this case, $\mathcal{P}(x)$ can be used to define a larger number of positive pairs. Given an anchor point $x$, their contrastive loss is defined as:

$$\mathcal{L}_{OpenCon} = -\frac{1}{|\mathcal{P}(x)|} \sum_{z^+ \in \mathcal{P}(x)} \log \frac{\exp(z \cdot z^+/\tau)}{\sum_{z^- \in \mathcal{N}(x)} \exp(z \cdot z^-/\tau)} \tag{2.13}$$

where $\tau$ is a temperature parameter and $z$ is the $\ell_2$-normalized projection of $x$. Two additional terms are optimized during training: the supervised contrastive loss [53] on the labeled data $D^l$ and the self-supervised contrastive loss [76] on the unlabeled data $D^u$. During training, the class prototypes are defined as moving averages and cluster assignments are updated after each epoch.

## 2.6 Related works

### 2.6.1 Unsupervised Clustering

The NCD problem is closely related to unsupervised clustering. In both domains, the aim is to find a partition of a dataset where no prior knowledge on the novel classes is available. Just like in NCD, a common approach is to consider that the close neighborhood of an instance is likely to belong to the same class. In this case, groups where instances are more similar to each other than they are to other groups are created. The definition of this similarity can vary a lot depending on the purpose of the study or domain-specific assumptions. The most widely known methods of clustering are usually unsupervised, however we still distinguish them from the less common *semi-supervised* approach (see Section 2.6.2) that leverages a small amount of information to guide the definition of the clusters.

In the completely unsupervised case, many shallow and deep learning based methods have been proposed. We refer the reader to [43] for fundamental work and [78] for a more detailed survey. Some of the main categories of clustering algorithms are: Centroid-

based algorithms create clusters by determining the proximity of data points to a central vector. Connectivity-based algorithms group data points into clusters using a tree-like structure. Distribution-based algorithms model the data with a chosen distribution and form clusters based on the likelihood of data points belonging to the same distribution. Density-based algorithms define clusters as regions of high data density and consider points in sparsely populated areas as outliers. Finally, Deep Clustering methods aim at jointly conducting dimensionality reduction (or feature transformation) and clustering, which is done independently in other classical works [78].

As Deep Clustering methods learn rich informative representations while separating data into clusters without supervision, their architectures and loss functions are often similar to those used in NCD methods, and they are sometimes even used as baselines in NCD. They can be easily adapted to the NCD setting, for example by adding a supervised objective trained on the labeled data from $D^l$ to guide the clustering process.

**Discussion.** As expressed in the introduction, fully unsupervised clustering is not a complete solution to the NCD problem. Multiple and equally valid criteria to partition a dataset can be used, so the definition of what constitutes a good class becomes ambiguous. This is why the use of a labeled dataset becomes essential to narrow down what constitutes a proper class and guide the clustering process. Nonetheless, clustering methods are a frequent building block in NCD methods. An example of this is *Deep Transfer Clustering* [37], where the authors extend *Deep Embedded Clustering* [43] by guiding its training process with the known classes. A few works use $k$-means and its variations for label assignment in the feature space of a deep network [36, 38]. And [79] employs both $k$-means and spectral graph theory to explore the novel classes.

### 2.6.2 Semi-Supervised Learning

Semi-Supervised Learning [80] is an instance of *weak supervision*, as it uses a limited amount of information in order to carry out its task. It is often reviewed in Novel Class Discovery articles for the similarity of its setup. Four different scenarios can be distinguished in Semi-Supervised Learning: semi-supervised dimensionality reduction [81], semi-supervised regression [82], semi-supervised classification [83, 84] and semi-supervised clustering [85, 86, 87]. Only the last two are relevant for our problem, and they are briefly introduced below.

In **Semi-Supervised Classification**, only a small portion of the dataset is labeled,

while the rest is unlabeled. This is a setup that can arise when labeling every instance is too costly, but we still wish to leverage the unlabeled data. Similarly to supervised classification, the goal is to assign instances to one of the classes seen in training, however traditional supervised classification will not take advantage of the unlabeled data. In this situation, a more accurate model can often be built using semi-supervised learning. Examples of such models include *constrained k-means* and *seeded k-means* [83, 88]. They are extensions of $k$-means that use a labeled subset to initialize the centroids of the clusters. It is important to note that the methods in this domain focus on the classification task, where the classes in labeled and unlabeled sets are the same. This is the main difference with the NCD domain, and the reason why semi-supervised learning methods cannot be transferred to our problem.

In the case of **Semi-Supervised Clustering**, additional information in the form of "must-link" and "cannot-link" constraints is usually available. It indicates if pairs of instances must or must not be placed in the same cluster. Such relations can be derived from class labels. Examples of semi-supervised clustering algorithms include *COP-Kmeans* [85], *PCKmeans* [86] and *kernel spectral clustering* [87]. The Novel Class Discovery problem could be reformulated as a Semi-Supervised Clustering problem by defining must-link and cannot-link constraints. However, the complete set of constraints can only be defined for the labeled data thanks to the ground truth labels available. Only cannot-link constraints can be defined between the labeled and unlabeled data (using the hypothesis that $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$), and no constraints can be defined for pairs of unlabeled data. We do not expect this set of constraints to help the clustering process of the unlabeled data. Furthermore, most Semi-Supervised Learning methods are modified versions of the $k$-means algorithm, and will also suffer when the clusters are not spherical or when the dimension is too large and the euclidean distances becomes inadequate.

**Discussion.** Semi-supervised learning methods require either the classes to be known in advance (in the case of partially labeled data) or known constraints on the observations, which is not the case in NCD. Recent works [89, 90] have also shown that the presence of novel class samples in the unlabeled set negatively impacts the performance of such models. Some articles address this issue [91], but they do not attempt to discover the novel classes. As such, semi-supervised works are not directly applicable to the Novel Class Discovery problem.

### 2.6.3 Transfer Learning

Transfer Learning is an other domain often mentioned in NCD articles. It is a field of machine learning that aims at leveraging knowledge from a source domain or task to solve a different (but related) problem faster or with better generalization. In computer vision, Transfer Learning is commonly expressed by starting the training from a model that was pre-trained on the ImageNet [92] dataset. Two scenarios of transfer learning can be distinguished and they are introduced in Table 2.4.

Table 2.4 – Overview of the scenarios of transfer learning

| Scenario | Definition | Example |
|---|---|---|
| cross-domain transfer learning | Also known as *domain adaptation*, a model trained to execute a task on one domain is used to learn the same task on a different (but related) domain. | *The knowledge of a classifier trained to recognize positive or negative reviews on the domain of movies can be transferred to the domain of book reviews* [93]. |
| cross-task transfer learning | The knowledge gained by learning to distinguish some classes is then applied on other classes of the same domain. | *A model that was trained to recognize the 5 first digits of the MNIST dataset can be expected to more effectively learn to distinguish the 5 other digits of MNIST* [94]. |

With **cross-domain transfer learning**, a model can be pre-trained on a different but related source dataset. This is useful when the target dataset has too few instances to obtain good generalization. In this context, the "re-usability" of the source data depends on the degree of overlap between the features of the source and target domains. This idea is explored in [95], where the authors distinguish two categories of approaches. The instance-based approaches attempt to reuse the source domain data after re-sampling or re-weighting and are sensitive to such overlapping. And feature-representation-based approaches try to find a good representation for both the source and target domain.

In **cross-task transfer transfer learning**, the label spaces are different. In this case, methods learn a pair of feature mappings to transform the source and target domain to a common latent space [96, 97]. Another approach is to learn a feature mapping to transform data from one domain to another directly [98, 99].

**Discussion.** NCD can be viewed as an unsupervised cross-task transfer learning task, where the knowledge from a classification task on a source dataset is transferred to a clus-

tering task on a target dataset. The large majority of Transfer Learning articles require the labels of both the source and target domains to be known in advance, which makes the use of such methods impossible in our context of class discovery. The Constrained Clustering Network (CCN) [29] is an exception in this regard. It is a method proposed to solve two different transfer learning scenarios, one of which being a cross-task problem where the labels of the target data that must be inferred are not available. This is essentially the NCD problem, which eventually led to this paper being recognized as one of the earliest NCD works.

### 2.6.4 Open-World Learning

Rather than being a domain in and of itself, Open-World Learning (OWL) [25] is a broad term that encompasses all the domains that live under the *open-world* assumption. Traditional machine learning tasks focus on *closed-world* settings, where the test instances can only be from the distribution that was seen during training. This is in opposition to the *open-world* setting, where instances can come from outside of the training distribution. Some of these domains include Anomaly Detection (AD), Novelty Detection (ND), Open Set Recognition (OSR), Out-of-Distribution Detection (OOD Detection) and Outlier Detection (OD). They are concerned with either or both of semantic shift (when new classes appear) and covariate shift (when the definition of the known classes changes).

To help the reader distinguish these domains, Table 2.5 summarizes a few important criteria. And a general description of each of the 5 domains is provided below.

Table 2.5 – Overview of the domains in Open-World Learning.

| Need to ... | NCD[1] | GCD[2] | AD[3] | ND[4] | OSR[5] | OOD[6] Detection | OD[7] |
|---|---|---|---|---|---|---|---|
| recognize OOD instances | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| have OOD samples during training | ✓ | ✓ | ✓/✗ | ✗ | ✗ | ✓ | ✓ |
| accurately classify known samples | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ |
| discover the new classes | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |

[1]Novel Class Discovery, [2]Generalized Category Discovery, [3]Anomaly Detection, [4]Novelty Detection, [5]Open Set Recognition, [6]Out-of-Distribution, [7]Outlier Detection.

**Anomaly Detection:** Given a predefined "normality", the goal of AD is to identify abnormal observations. The abnormality can originate either from a semantic or covariate shift [100]. For example, given a set of pictures of dogs, a model capable of recognizing if

a picture is not a dog (i.e. a picture of a cat) falls under semantic shift AD. In this case, the normality corresponds to all pictures of dogs. And a model designed to recognize if a given picture of dog is from a breed seen in training falls under covariate shift AD. We can see that the key to successfully building an AD model is to precisely define the notion of normality.

Two categories of AD settings can be distinguished: either the training set represents the normality, or the training set is labeled "normal" and "abnormal". The first setting is usually preferred, as anomalous data is often found in limited quantities (or even completely unavailable), which makes unsupervised approaches more attractive than supervised ones.

**Novelty Detection:** From a clean training set with only instances of known classes, the goal of ND is to identify if new test observations come from a novel class or not. This problem is very close to Anomaly Detection, but it can be differentiated in two ways: First, this problem is concerned only with semantic shift (i.e. the apparition of new classes). And second, it does not consider novel samples as "anomalies" that must be discarded, but rather as new learning opportunities from events that were not seen during training [101]. ND stems from the idea that during training, a model cannot have seen all possible classes. Since this idea is very valid in production, traditional classification models can be difficult to apply, and ND models are more convenient.

However, the authors of [25] conclude that the goal of ND is only to distinguish novel samples from the training distribution, and not to actually discover the novel classes. Therefore, most methods assume that the discovery of the new classes in the rejected examples is either the duty of a human or a task that is outside of the scope of their research. This is a major difference with Novel Class Discovery (NCD), as ultimately, the goal of NCD is to explore the novel samples. To the best of our knowledge, [102] is an exception. In this work, an attempt is made by the system to solve this problem while still addressing the other concerns of open-world learning.

**Open Set Recognition:** The idea behind Open Set Recognition (OSR) [103] is that standard neural networks have a tendency to output high confidence predictions even when confronted with instances from classes that were never seen during training. OSR therefore tries to detect when samples come from unknown classes additionally to accurately classifying the known classes. An example of an OSR system would be an application

trained to recognize certain faces to allow entry into a building. Such a system must (i) identify known people and (ii) reject the faces from people it has never seen instead of predicting one of the known faces.

**Out-of-Distribution Detection:** Similarly to OSR, OOD Detection originates from the idea that machine learning models can predict labels with high confidence for instances of classes they have never seen during training. OOD Detection methods also aim to (i) accurately classify samples of known classes and (ii) reject samples from outside the known distribution. Because the definition of "distribution" depends on the application, OOD Detection methods cover a large range of methods. These methods are generally given both In-Distribution (ID) and Out-of-Distribution (OOD) samples during training (see Table 2.5) to narrow down the definition of ID. Note that OSR and OOD Detection are very close both in setting and goal. However, they can be differentiated primarily by the fact that OSR methods are tasked with identifying instances that suffer a semantic shift, but originate from the same source dataset, while OOD Detection methods seek to identify semantically different instances that come from a completely different dataset with non-overlapping classes.

**Outlier Detection:** OD is a task that deviates from the 4 other OWL tasks defined above, as there is no train/test split and all the data is processed together. The goal is to detect samples that present a significant semantic or covariate deviation from others according to some measure. Some of the applications of such methods include network intrusion detection [104], video surveillance [105] and dataset preprocessing [106]. Outlier Detection is a well-studied domain with a large number of proposed methods. Distance-based methods identify points that are far away from all of their neighbors [107], density-based methods select points in sparsely populated regions [108] and clustering-based methods capture samples that did not fall in any of the major clusters [109].

**Discussion.** The main objective of Open-World Learning (OWL) methods is generally to identify instances that come from a different distribution than the known classes in order to reject them and keep a high performance on known classes. These methods ignore the rejected instances and do not seek to cluster them into novel classes (see Table 2.5). Because in the *open-world* setting, the data at training or inference time will be a mix of In- and Out-of-Distribution samples, OWL methods are always at least tasked to recognize

Out-of-Distribution samples. This is not a concern in Novel Class Discovery (which does not belong to OWL), as we are given separate datasets during training and only samples from novel classes at inference. Instead, NCD could be seen as an extension from OWL works where, after novel samples were detected, we seek to discover the underlying classes. But as the main focus of these articles is not relevant to the NCD problem, it is difficult to transfer OWL works to NCD.

However, Generalized Category Discovery (GCD, see Section 2.4) can be seen as a domain that is halfway between OWL and NCD. Like in NCD, methods in GCD are given two separate sets during training: a labeled set of known classes and an unlabeled set of novel classes. And like in OWL, test samples in GCD can be either from known or novel classes. Generalized Category Discovery is very close to OSR and OOD Detection, as it shares their goal of accurately classifying known samples and identifying samples from novel classes. It can, however, be distinguished by the fact that semantically shifted samples originate from the same parent distribution (i.e. they are classes from the same dataset). Moreover, it does not stop at identifying which samples come from novel classes, but actually tries to discover the novel classes.

As many methods in AD/ND/OSR/OOD Detection/OD can be applied to detect instances that are semantically different from the known classes, they could potentially be used for the task of GCD to distinguish if instances come from known or novel classes. Such methods could be used in a two-stage approach, where test samples would first be designated as belonging to known or novel classes using OWL methods, and then the samples of novel classes would be clustered with NCD methods. However, holistic approaches are usually preferred by researchers and works in GCD seem to be following this path [36, 58, 60, 57].

## 2.7 Conclusion and perspectives

This chapter extensively examined the publications in the new field of Novel Class Discovery. We formally defined the setup and key components of NCD, and proposed a taxonomy that categorizes NCD frameworks based on the way knowledge is transferred between the labeled and unlabeled sets. We found that two-stage methods were initially popular, but their risk of overfitting on the known classes encouraged defining single-stage methods, which are now widely adopted. We believe this taxonomy will help guide future research by giving a clear overview of the families of approaches and techniques that

have already been explored. NCD is a newly emerging field that offers a more practical setting compared to fully supervised or unsupervised methods in certain situations. This has led to the creation of new domains, which we have also analyzed, as researchers have relaxed their assumptions and devised new challenges inspired by NCD. Additionally, we identified and presented techniques and tools that are commonly used in NCD. Finally, since this is a new domain that lies at the intersection of several others, it can become challenging to distinguish NCD from other areas of research. Thus, we also presented the domains most closely related to NCD and highlighted the main differences. We hope that this last section will help readers unfamiliar with NCD to understand what distinguishes it from other domains.

Despite the growing body of work in this area, several questions remain unanswered and some perspectives, in our view, are worthy of further study. As we have seen in this chapter, the majority of NCD works are applied only to image data due to specialized architectures and techniques such as data augmentation and self-supervised learning, which rely on the unique structure of images. They are partly responsible for the success of NCD methods, and since they are not directly applicable to other data types, most works are still limited to image data. However, it is worth exploring the potential of applying such methods to other data types such as text, tabular, and others. DTC [37] has shown that deep clustering methods can easily be transferred to the NCD problem, and we expect that more of them could be adapted and offer a new source of inspiration. Some procedures have been proposed to determine the number of novel classes automatically with varying degrees of success. Ideally, NCD methods should not make the assumption that this number is known in advance, but this is most likely not a limiting factor in real-world scenarios. We also believe that it is crucial to have a unified benchmark and evaluation protocol, since previous works have shown that the split of known/novel classes has an influence on the difficulty of the NCD problem [40]. Lastly, the accuracy of pseudo-labeling, which widely used in one-stage frameworks, is a decisive factor to the success of these methods. There is still room for improvement in this area, for instance, taking labeled data into account, or taking inspiration from graph theory and spectral clustering.

In the next chapter, we address one of these open questions by proposing a method specifically designed for the problem of Novel Class Discovery in tabular data. This is also the first step in solving our original FTTH fault diagnosis problem, as explained in Chapter 1. Due to the lack of competitors in this specific context, its effectiveness will only be compared to unsupervised clustering methods.

Chapter 3

# A Method for Discovering Novel Classes in Tabular Data

## Contents




This chapter has been published at an international conference: [21] A Method for Discovering Novel Classes in Tabular Data. Colin Troisemaine, Joachim Flocon-Cholet, Stéphane Gosselin, Sandrine Vaton, Alexandre Reiffers-Masson and Vincent Lemaire. In: *IEEE International Conference on Knowledge Graph (ICKG)*, 2022.

## 3.1 Introduction

The recent success of machine learning models has been enabled in part by the use of large quantities of labeled data. Many methods currently assume that a large part of the available data is labeled and that all classes are known. However, these assumptions do not always hold true in practice, prompting researchers to explore scenarios where unlabeled data is available [27, 28]. As we have seen in Chapter 2, Novel Class Discovery (NCD) [29] is a recently proposed domain where prior knowledge from known classes is used to identify new classes in unlabeled data. In this setup, the data is divided in two sets. The first is a labeled set containing known classes and the second is an unlabeled set containing novel classes that must be discovered. Some solutions have been proposed for the NCD problem in the context of computer vision [17, 37, 38, 39] and have displayed promising results.

However, research in this area is still new, and NCD has not yet been addressed for tabular data. While audio and image data have been of great interest in recent scientific publications, tabular data remains a very common information structure that is found in many real world problems where NCD could be beneficial. For instance, customer behavior analysis could leverage NCD techniques to identify previously unknown patterns of customers based on their behavior, preferences, or demographics [110]. Another example is cybersecurity, where NCD can be used to detect new types of cyber threats, malware, or anomalous network activities that are not covered by existing security systems.

Tabular data can come in large unlabeled quantities due to the labeling process being often costly and time-consuming, in part due to the difficulty of visualization, which is why a lot of unlabeled data is often left unexploited. In [18], the authors review the primary challenges that come with tabular data and identify three main obstacles to the success of deep neural networks on this type of data. The first is the quality of the data, that often includes missing values, extreme data (outliers), erroneous or inconsistent data and class imbalance. Then, the lack of spatial correlation between features makes it difficult to use techniques based on inductive biases, such as convolutions or data augmentation [111]. And finally, the heterogeneous and complex nature of the data (dense continuous and sparse categorical features) that can require considerable preprocessing, leading to information loss compared to the original data [112].

**Our proposal:** To address the NCD problem in the challenging tabular data environment, we propose TabularNCD (for Tabular Novel Class Discovery). In the first step

of our method, an unbiased latent representation is initialized by taking advantage of the advances in Self-Supervised Learning (SSL) for tabular data [55]. Then, we build on the idea that the local neighborhood of an instance in the latent space is likely to belong to the same class, and a clustering of the unlabeled data is learned through similarity measures. The process in this second step is jointly optimized with a classifier on the known classes to include the relevant features from the already discovered classes.

Our key contributions are summarized as follows:

— We propose TabularNCD, a new method for Novel Class Discovery. To the best of our knowledge, this is the first attempt at solving the NCD problem in the context of tabular data with heterogeneous features. Thus, we do not depend on the spatial inductive bias of features, which other NCD methods rely heavily on when using convolutions and specialized data augmentation techniques.

— We empirically evaluate TabularNCD on seven varied public classification datasets. These experiments demonstrate the superior performance of our method over common fully unsupervised methods and a baseline that exploits known classes in a naive way.

— We also introduce an original approach for the definition of pairs of pseudo-labels of unlabeled data. This approach exploits the local neighborhood of an instance in a pre-trained latent space by considering that its $k$ most similar instances belong to the same class. We study its robustness and the influence of its parameters on performance.

— Lastly, we conduct experiments to understand in depth the reasons for the advantage that the proposed method has over simpler approaches.

## 3.2 Novel Class Discovery and related works

The process of discovering new classes depends on the final application in the real world (offline vs online learning, nature of the data, etc). In the literature of "concept drift" [113], changes in the distribution of known classes can appear either gradually or suddenly. Some analogies can be made with the present chapter, where we consider the case of the "sudden drift", as the labeled dataset $D^l$ and unlabeled dataset $D^u$ do not share any classes (as described in Section 3.1). Furthermore, we assume that a form of knowledge can be extracted from $D^l$ to be then used on $D^u$. In this sense, "Novel Class

Discovery (NCD)" methods mainly lie at the intersection of several other lines of research that we briefly review here.

**Transfer Learning [114]:** To solve a problem faster or with better generalization, transfer learning leverages knowledge from a different (but related) problem. Transfer learning can either be *cross-domain*, when a model trained to execute a task on one domain is retrained to solve the same task but on another domain. Or it can be *cross-task*, when a model trained to recognize some classes is retrained to distinguish other classes of the same domain. NCD can be regarded as a cross-task transfer learning problem, in the sense that it aims to cluster unlabeled data by leveraging knowledge from different labeled data. However, most of the works in transfer learning require labeled data in the source and target domains to train a classifier for the new task. When there are no labels in the target domain, cross-task transfer learning methods usually employ unsupervised clustering approaches trained on features extracted from labeled data.

**Open-World Learning:** Unlike in the traditional *closed-world learning*, the problem of Open-World Learning expects at test time instances from classes that were not seen during training. The objective is therefore threefold: 1) to obtain good accuracy on the known classes, 2) to recognize instances from novel classes and optionally, 3) to incrementally learn the new classes. In practice, most methods [115, 116] assume that the discovery of the new classes in the rejected examples is either the duty of a human or a task that is outside the scope of their research. To the best of our knowledge, [117, 102] are exceptions in this regard. However, Open-World Learning still diverges from NCD, as classes in the unlabeled set can be novel or not. This is not the case for NCD, where it is assumed that it is already known if an instance belongs to a novel class or not. In other words, the focus of NCD is only on the third task of Open-World Learning.

**Semi-Supervised Learning:** When a training set is partially labeled, classical supervised methods can only exploit the labeled data. Semi Supervised Learning [80] is a special instance of *weak supervision* where additional information in the form of a labeled subset or some sort of known relationship between instances is available. Using this information, models can learn from the full training set even when not all of the instances are labeled. However, all of these methods build on the assumption that labeled and unlabeled sets contain instances of the same classes, which is not the case in NCD.

**Novelty Detection:** From a training set with only instances of known classes, the goal of *Novelty Detection* (ND) is to identify if new test observations come from a novel class or not. This problem is concerned with semantic shift (i.e. the apparition of new

classes). The authors of [25] conclude that the goal of ND is only to distinguish novel samples from the training distribution, and not to actually discover the novel classes. This is the first major difference with Novel Class Discovery, as the ultimate goal of NCD is to explore the novel samples. The second difference is that in NCD, the known and novel classes are already split, which is not the case in ND.

**Novel Class Discovery (NCD):** In a large part of the literature related to NCD, the goal is to cluster classes in an unlabeled set when a labeled set of known classes is available. Unlike in semi-supervised learning, the classes of these two sets are disjoint. The labeled data is used to reduce the ambiguity that comes with the many potentially valid clustering criteria of the unlabeled data. It is a challenging problem, as the patterns learned from the labeled data may not be relevant or sufficient to cluster the unlabeled data. In [19], the authors explore the assumptions behind NCD and give a formal definition of NCD. They find that this is a theoretically solvable problem under certain assumptions, the most important one being that the labeled and unlabeled data must share good high-level features and that it must be meaningful to separate observations from these two sets.

NCD methods have mainly been developed for the computer vision problem, and the pioneering works involve *Constrained Clustering Network* (CCN) [29], where the authors approach NCD as a transfer learning problem and rely on the KL-divergence to evaluate the distance between the cluster assignments distributions of two data points. In *Deep Transfer Clustering* (DTC) [37], the authors extend the *Deep Embedded Clustering* method [43] by taking the available labeled data into account, allowing them to learn more relevant high-level features before clustering the unlabeled data. Another important work is *AutoNovel* [17], where two neural networks update a shared latent space. Finally, *Neighborhood Contrastive Learning* [39], which is based on *AutoNovel*, adds two contrastive learning terms to the loss to improve the quality of the learned representation. The common denominators of these works are the learning of a latent space and the definition of pairwise pseudo-labels for the unlabeled data. Despite showing competitive performance, they are all solving NCD for computer vision problems. Thus, they are able to take advantage of the spatial correlation between the features with specialized architectures and they are not affected by the other challenges of tabular data. Because data augmentation is a crucial component to regularize the network, it is not possible to directly transfer their work on tabular data.

In the next section, the architecture of the proposed method is depicted, along with the two main steps of the training process.

## 3.3 The method

Given a labeled set $D^l = \{(x_i^l, y_i^l)\}_{i=1}^N$ where each instance $x_i^l$ of $X^l \in \mathbb{R}^{d \times N}$ has a one-hot label $y_i^l \in \mathcal{Y}^l = \{1, \ldots, C^l\}$ and an unlabeled set $D^u = \{x_i^u\}_{i=1}^M$ where only the instances $x_i^u$ of $X^u \in \mathbb{R}^{d \times M}$ are available. The goal is to identify the $C^u$ classes of $D^u$ by predicting their labels $y_i^u \in \mathcal{Y}^u = \{1, \ldots, C^u\}$. In this chapter, our setting assumes that this number $C^u$ of novel classes is known in advance and that the classes of $D^l$ and $D^u$ are disjoint (i.e. $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$). Additionally, we follow the formulation of the NCD problem from [19] and suppose that $P_{D^l}(Y|X)$ and $P_{D^u}(Y|X)$ are statistically different and separable, but still share high-level semantic features, so that we can extract general knowledge from $D^l$ of what constitutes a pertinent class. So to *identify* the classes in $D^u$, a transformation $\phi$ must be learned, such that $\phi(X^u)$ is separable.

The proposed method includes two main steps: (i) first, the representation is initialized by pre-training the encoder $\phi$ on $D^l \cup D^u$ without using any label. Then (ii), a supervised classification task and an unsupervised clustering task are jointly solved on the previously learned representation, further updating it. Each of these two steps has its own architecture and training procedure which are described below and in the next sections.

Similarly to [39], our method is based on AutoNovel [17], and the representation is obtained using a model that includes a feature extractor $\phi(x) = z$, which is a simple combination of multiple dense layers with non-linear activation functions (see Fig. 3.1). In the second step, the latent representation of the feature extractor is forwarded to two linear classification and clustering layers followed by Softmax layers (see Fig. 3.2).

**Why rely on a deep architecture?** Despite much research efforts in deep learning, the tasks of classification and regression in the context of tabular data are still dominated by "classical" approaches based on tree-ensembles. In [18], the authors benchmarked 12 state-of-the-art deep learning-based approaches against 9 classical methods on both regression and classification tasks. They found that XGBoost [118], LightGBM [119] and CatBoost [120] outperformed both deep and shallow neural networks on most datasets with significantly less training time. They conclude that for supervised tasks, the use of current deep learning techniques is generally inefficient compared to gradient boosting methods. However, for unsupervised problems such as clustering, deep learning methods offer certain advantages. Tabular data often suffers from sparse values, high dimensionality and heterogeneous features that make it difficult to accurately measure distances between points. Deep encoders are a good solution to these challenges because the data

is projected into a homogeneous latent space of reduced dimensionality. Furthermore, in a well-defined latent space, instances close to each other are likely to belong to the same class. These advantages, together with the supervised information available in NCD that can be exploited during training, lead us to expect better performance when using a deep architecture.

The following subsections describe in more details the two main training steps of the method and the consistency regularization term, which is required in the kind of architecture that is presented here.

### 3.3.1 Initialization of the representation

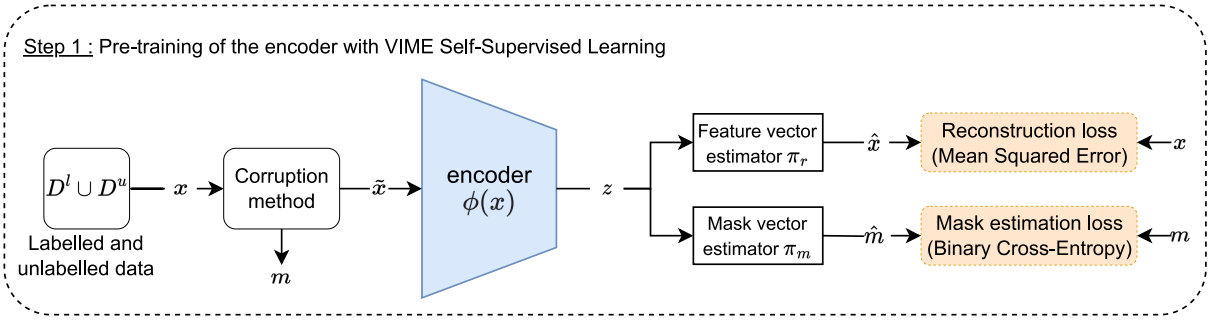



Figure 3.1 – Architecture of the pre-training step.

This first step aims at capturing a common and informative representation of both $D^l$ and $D^u$. This is important because the representation is used in the next step, among other things, to compute the similarity of pairs of instances and thus determine if examples should belong to the same cluster or not. Among the possibilities offered in the literature, the way we decided to elaborate this representation is to project examples in a latent space produced by a deep architecture trained on $D^l$ and $D^u$.

To pre-train the latent space using both labeled and unlabeled data, we take advantage of *Self-Supervised Learning* (SSL), and apply the *Value Imputation and Mask Estimation* (VIME) method [55]. As the name suggests, VIME defines two pretext tasks to train the encoder $\phi$. From an input vector $x \in \mathbb{R}^d$ that has been corrupted, the objective is to 1) recover the original values and 2) recover the mask used to corrupt the input. The corruption process begins with the generation of a binary mask $m = [m_1, ..., m_d]^\top \in \{0, 1\}^d$, with $m_j$ randomly sampled from a Bernoulli distribution with probability $p_m$. The corrupted vector $\tilde{x}$ is then created by replacing each dimension $j$ of $x$ where $m_j = 1$

with the dimension $j$ of a randomly sampled instance from the training set. So $\tilde{x} = (1 - m) \odot x + m \odot \bar{x}$, where $\odot$ is the element-wise product and each $\bar{x}_j$ of $\bar{x}$ has been randomly sampled from the empirical marginal distribution of the $j$-th feature of the training set.

The pre-training framework is illustrated in Fig. 3.1. The encoder and the mask and feature vector estimators are jointly trained with the following optimization problem:

$$\mathcal{L}_{VIME} = l_{recons.} + \alpha l_{mask} \tag{3.1}$$

where $\alpha$ is a trade-off parameter. Following [55], we use $\alpha = 2.0$. The *mean squared error* (MSE) loss is used to optimize the feature vector estimator:

$$l_{recons.} = \frac{1}{d} \sum_{j=1}^{d} (x_j - \hat{x}_j)^2 \tag{3.2}$$

where $\hat{x}$ is the input reconstructed from the corrupted vector $\tilde{x}$. The *binary cross-entropy* (BCE) loss is used to optimize the mask estimator:

$$l_{mask} = -\frac{1}{d} \sum_{j=1}^{d} [m_j \log(\hat{m}_j) + (1 - m_j) \log(1 - \hat{m}_j)] \tag{3.3}$$

where $\hat{m}$ is the estimated mask.

The authors argue that by training an encoder to solve these two tasks, it is possible to capture the correlation between the features and create a latent representation $z$ that contains the necessary information to recover the input $x$. While the general idea of the VIME method is similar to that of a *denoising auto-encoder* (DAE), it reports superior performance in [55] and there are two major differences. The first is the addition of the mask estimation task, and the second lies in the noise generation process. A DAE will create noisy inputs by adding Gaussian noise or replacing values with zeroes, while VIME randomly selects values from the empirical marginal distribution of each feature. This means that the noisy $\tilde{x}$ of VIME will be harder to distinguish from the input.

Note that a simple approach to initialize the encoder would be to train a classifier using the known classes. However, the resulting representation would rapidly overfit on these classes and the unique features of the novel classes would be lost. Instead, by generating pseudo-labels from pretext tasks on unlabeled data, a model can be trained to learn informative representations and high-level features that are unbiased towards labeled data.

This is the general idea behind Self-Supervised Learning and it is a technique commonly used in NCD [121, 17, 39]. An example of it is RotNet [51], where the model has to predict the rotation that was applied to an image from the possible 0, 90, 180 and 270 degrees values. Unfortunately, only a few works have attempted to adapt this technique for tabular data [70, 71, 55] and have not had the same success.

### 3.3.2 Joint training on the labeled and unlabeled data

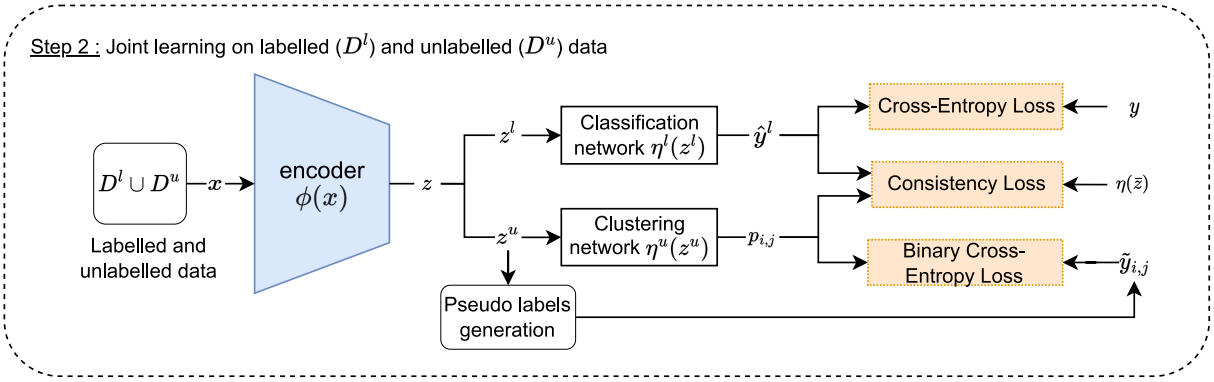



Figure 3.2 – Architecture of the joint learning step. After the input data $x$ has been projected, labeled data $z^l$ is used to train the classification network $\eta^l$ and unlabeled data is used to train the clustering network $\eta^u$ with the generated pseudo-labels as its target.

We formulate the novel class discovery process as multi-task optimization problem. In this step, two new networks are added on top of the previously initialized encoder, each solving different tasks on different data (see Fig. 3.2). The first is a classification network $\eta^l(z) \in \mathbb{R}^{C^l+1}$ trained to predict 1) the $C^l$ known classes from $D^l$ with the ground truth labels and 2) a single class formed of the aggregation of the unlabeled data. The second is another classification network trained to predict the $C^u$ novel classes from $D^u$. It will be referred as the *clustering* network $\eta^u(z) \in \mathbb{R}^{C^u}$. These two networks share the same embedding and both update it through back-propagation, sharing knowledge with one another.

The classification network is optimized with the *cross-entropy* loss using the ground truth one-hot labels $y$:

$$l_{class.} = -\sum_{c=1}^{C^l+1} y_c \log(\eta_c^l(z)) \tag{3.4}$$

where $z = \phi(x)$. Its role is to guide the representation to include the features of the known classes that are relevant for the supervised classification task.

To train the clustering network $\eta^u$ with unlabeled data in a supervised manner, pseudo-labels $\tilde{y}_{i,j} \in \{0, 1\}$ are generated for each pair $(x_i, x_j)$ of unlabeled data in a mini-batch. This is comparable to a pretext task in Self-Supervised Learning. Another possible approach would be to predict the similarity of instances directly, but using pseudo-labels has the advantage of directly associating classes to the instances and does not require an additional clustering of the embedding. In this case, pseudo-labels indicate if a pair of instances should belong to the same class or not, independently of the class number predicted by the network. They are defined as $\tilde{y}_{i,j} = 1$ if $z_i$ and $z_j$ are similar, and $\tilde{y}_{i,j} = 0$ if they are dissimilar. After the mini-batch $X$ has been projected in the encoder ($Z = \phi(X)$), there are $|Z| - 1$ pairwise pseudo-labels defined for each instance $z_i \in Z$. The clustering network is optimized with the *binary cross-entropy* loss, which is for a single instance $z_i$:

$$l_{clust.} = \frac{1}{|Z| - 1} \sum_{\substack{j=1 \\ j \neq i}}^{|Z|} [-\tilde{y}_{i,j} \log(p_{i,j}) - (1 - \tilde{y}_{i,j}) \log(1 - p_{i,j})] \tag{3.5}$$

where $p_{i,j} = \eta^u(z_i) \cdot \eta^u(z_j)$ is a score close to 1 if the clustering network predicted the same class for $z_i$ and $z_j$ and close to 0 otherwise. For this reason, $p_{i,j}$ can directly be compared to the pairwise pseudo-labels $\tilde{y}_{i,j}$. The intuition is that instances similar to each other in the latent space are likely to belong to the same class. Therefore, $\eta^u$ will create clusters of similar instances, guided by $\eta^l$ with the knowledge of the known classes.

**Note:** Predicting the unlabeled data as a single new class in the classification network is not done in AutoNovel [17] and is one of the main differences with our work. We found that having an overlap of the instances in the classification and clustering networks would improve the performance and result in the definition of cleaner latent space that does not mix labeled and unlabeled data.

**Definition of pseudo-labels.** As expressed in Section 2.5.2, many NCD works (e.g. [17, 42, 39]) define pseudo-labels as a form of weak supervision to iteratively refine the prediction of their clustering network. Pseudo-labels are defined for each unique pair of unlabeled instances in a mini-batch based on similarity. The most common approach is to define a threshold $\lambda$ for the minimum similarity of pairs of instances to be assigned to the same class. We refer the reader to Figure 2.10 for a more intuitive explanation of this process.

However, we found experimentally (see Appendix A.1) that defining for each point the *top* $k$ most similar instances as positives was a more reliable method. So, for each

pair $(x_i, x_j)$ in the projected batch of unlabeled data $Z$, the pseudo-labels are assigned as follows:

$$\tilde{y}_{i,j} = \mathbb{1}[j \in \underset{\substack{r \in \{1,\dots,|Z|\} \\ r \neq i}}{\text{argtop}_k}\ \delta(z_i, z_r)] \tag{3.6}$$

where $\delta(z_i, z_r) = \frac{z_i \cdot z_r}{\|z_i\| \|z_r\|}$ is the cosine similarity and $\text{argtop}_k$ is the subset of indices of the $k$ largest elements. This means that each observation has $k$ positive pairs and $|Z| - 1 - k$ negative pairs.

### 3.3.3 Consistency regularization

During the training of the two classification networks, the latent representation of the encoder changes. And since the pseudo-labels are defined according to the similarity of instances in the latent space, this causes a *moving target* phenomenon, where the pairwise pseudo-labels can differ from one epoch to another. To limit the impact of this problem, a regularization term is needed. Commonly employed in semi-supervised learning [90], we use "data augmentation". The idea is to encourage the model to predict the same class for an instance $x$ and its augmented counterpart $\bar{x}$. But while data augmentation has become a standard in computer vision, the same cannot be said for tabular data. The authors of [18] consider the lack of research in tabular data augmentation (along with the difficulty to capture the dependency structure of the data) to be one of the main reasons for the limited success of neural networks in tabular data. Nevertheless, we find that consistency regularization is an essential component for the performance of the method proposed here.

In this chapter we use SMOTE-NC (for Synthetic Minority Oversampling Technique [59]), which is one of the most widely used tabular data augmentation techniques. It synthesizes new samples from the minority classes to reduce imbalance, thus improving the predictive performance of models on these classes. Synthetic samples are generated by taking a point along the line between the considered sample and a random instance among the $k$ closest of the same class (in the original input space). Fig. 3.3 illustrates this technique. The larger the $k$ is, the wider and less specific the decision regions of the learned model will be, resulting in better generalization.

This method can be applied directly to instances of the labeled set, however we relax the constraint of selecting same-class instances for the unlabeled set, as no labels are available. It can also be noted that the nearest neighbors for the observations of the unlabeled set are not selected from the labeled set since one of the hypotheses of our

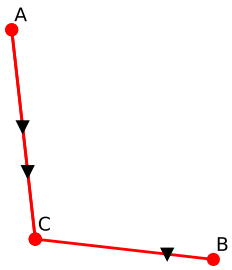



Figure 3.3 – Illustration of the SMOTE data augmentation technique. The red dots are the batch samples, and the black triangles are the synthetic new samples.

problem is that $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$ (the same goes for the labeled set).

The *mean squared error* (MSE) is used as the consistency regularization term for both the classification and the clustering network. For the clustering network, it is written as:

$$l_{reg.} = \frac{1}{C^u} \sum_{c=1}^{C^u} (\eta_c(z) - \eta_c(\bar{z}))^2 \tag{3.7}$$

where $\bar{z} = \phi(\bar{x})$ is the projection of $x$ augmented with the method inspired from SMOTE-NC. For the classification network, it is averaged over $C^l + 1$ instead.

### 3.3.4 Overall loss

The method proposed here falls under the domain of Multi-Task Learning (MTL) where instead of focusing on the only task that solves the problem at hand (discovering new classes in $D^u$), additional related tasks are jointly learned (classification of samples in $D^l$). By introducing new inductive biases, the model will prefer some hypotheses over others, guiding the model towards better generalization [122]. Our architecture falls under the *hard parameter sharing* domain of MTL, where the first layers are shared between the different tasks (i.e. the encoder), and the last few layers are task-specific.

We chose to adopt an alternating optimization strategy, which was introduced in [123]. In this case, we define one objective function and one individual optimizer per task. The loss of the classification network is:

$$\mathcal{L}_{classification} = w_1 l_{classif.} + (1 - w_1) l_{reg.} \tag{3.8}$$

And the loss of the clustering network can be written as:

$$\mathcal{L}_{clustering} = w_2 l_{clust.} + (1 - w_2) l_{reg.} \tag{3.9}$$

where $w_1$ and $w_2$ are trade-off hyperparameters for balancing the weight of the consistency regularization terms.

**Description of the training process:** The classification network and the clustering network are trained alternately. For each mini-batch during training, the classification network and the encoder are first updated through back-propagation with the loss from (3.8). Then, the unlabeled data of the same mini-batch is forwarded once again through the encoder to compute the loss from (3.9), which is back-propagated to update the clustering network and the encoder once more.

## 3.4 Experiments

### 3.4.1 Datasets and experimental details

**Datasets.** To evaluate the performances of the method proposed here, a variety of datasets in terms of application domains and characteristics have been chosen (see Table 3.1). Six public tabular classification datasets were selected: Forest Cover Type [124], Letter Recognition [125], Human Activity Recognition [126], Satimage [127], Pen-Based Handwritten Digits Recognition[127] and 1990 US Census Data [127], as well as MNIST [128], which was flattened to transform the $28 \times 28$ grayscale images into vectors of 784 attributes. We preprocess the numerical features of all the datasets to have zero-mean and unit-variance, while the categorical features are one-hot encoded. If the training and testing data were not already split, 70% were kept for training while the remaining 30% were used for testing.

Following the same procedure found in Novel Class Discovery articles [121, 17, 39], we hide the labels of some classes to create the *novel* classes. And the capacity of the compared methods to recover these classes is then evaluated. Around 50% of the classes are hidden, resulting in further splitting of the training and testing sets into labeled and unlabeled subsets. The partitions of the 7 datasets can be found in Table 3.1.

**Evaluation metrics.** To thoroughly assess the performance of the methods compared in the experiments described below, we report four different metrics. The first two are the clustering accuracy (ACC) and balanced accuracy (BACC), which can be computed after

Table 3.1 – Statistical information of the selected datasets.

| Dataset name | Attri-butes | # classes $C^l$ / $C^u$ | # train labeled | # train unlabeled | # test labeled | # test unlabeled |
|---|---|---|---|---|---|---|
| MNIST [128] | 784 | 5 / 5 | 30,596 | 29,404 | 5,139 | 4,861 |
| Forest Cover type [124] | 54 | 4 / 3 | 6,480 | 4,860 | 36,568 | 13,432 |
| Letter recognition [125] | 16 | 19 / 7 | 10,229 | 3,770 | 4,296 | 1,704 |
| Human activity [126] | 562 | 3 / 3 | 3,733 | 3,619 | 1,494 | 1,453 |
| Satimage [127] | 36 | 3 / 3 | 2,525 | 1,976 | 1,042 | 887 |
| Pendigits [127] | 16 | 5 / 5 | 3,777 | 3,717 | 1,764 | 1,734 |
| 1990 US Census [127] | 67 | 12 / 6 | 50,000 | 50,000 | 31,343 | 18,657 |

the optimal linear assignment of the class labels is solved with the Hungarian algorithm [2]. The use of the balanced accuracy is justified by the unbalanced class distribution of some datasets, which resulted in inflated accuracy scores that were not truly representative of the performance of the method. Another reported metric is the Normalized Mutual Information (NMI). It measures the correspondence between two sets of label assignments and is invariant to permutations. Finally, we compute the Adjusted Rand Index (ARI). It measures the similarity between two clustering assignments.

These four metrics range between 0 and 1, with values closer to 1 indicating a better agreement to the ground truth labels. The ARI is an exception and can yield values in $[-1, +1]$, with negative values when the index is less than the $Expected\_RI$. The metrics reported in the next section are all computed on the unlabeled instances from the test sets.

**Competing methods.** As expressed in Section 3.1, to the best of our knowledge, there is no other method that solves the particular Novel Class Discovery problem for tabular datasets. Nonetheless, we can compare ourselves to unsupervised clustering methods. This will also allow us to show that our method provides an effective way of incorporating knowledge from known classes. We choose the $k$-means algorithm for its simplicity and wide adoption, as well as the Spectral Clustering [129] method for its known good results to discover new patterns as in Active Learning [130]. We set $k$ to the ground truth (i.e. $k = C^u$) for both clustering methods, as it was already assumed that it is known in the proposed method.

We also define a simple baseline that clusters unlabeled data while still capitalizing on known classes: (i) first, a usual classification neural network is trained on the known classes from $D^l$; (ii) then the classifier's penultimate layer is used as a feature embedding

for $D^u$. In this projection, a $k$-means is applied to assign labels to the unlabeled test instances. Compared to the unsupervised approaches, this technique has the advantage of learning the patterns of the known classes in a homogeneous latent space. However, we still expect it to perform sub-optimally as the unique features and patterns of the unlabeled data might be lost in the last hidden layer after training.

**Implementation details.** For all datasets, we employ an encoder composed of 2 dense hidden layers that keeps the dimension of the input with activation functions and dropout values that are optimized hyperparameters. Following VIME [55] and AutoNovel [17], we use a single linear layer for the mask estimator, vector estimator, classification network and clustering network.

The hyperparameters (see Table A.3 in appendix) of the proposed method are optimized with a Bayesian search on a validation set which represents 20% of the training set. During the Self-Supervised Learning step (see Section 3.3.1), we observe little impact on the final performance of the model when optimizing the hyperparameters specific to the VIME method, and therefore use the values suggested in the original paper: a learning rate of 0.001, a corruption rate of 30% and a batch size of 128. In the joint training step (see Section 3.3.2), we use a larger batch size of 512 as the observations are randomly sampled from the merged labeled and unlabeled data. The hyperparameters of this step are the learning rates of both classification and clustering network optimizers, along with the trade-off parameters of their respective losses. The *top* $k$ number of positive pairs defined by (3.6) and the $k$ *neighbors* considered in the data augmentation method of Section 3.3.3 are also optimized and can all be found in Table A.3 in appendix. We use AdamW [131] as the optimizer for the pre-training step and for the two optimizers of the joint training step, and find that 30 epochs are enough to converge for both steps on any of the tabular datasets used.

We implemented our method under Python 3.7.9 and with the PyTorch 1.10.0 [132] library. The experiments were conducted on Nvidia 2080 Ti and Nvidia Quadro T1000 GPUs. The networks are initialized with random weights, and following [17], the results are averaged over 10 runs. The code is publicly available at `https://github.com/ColinTr/TabularNCD`.

### 3.4.2 Results

Table 3.2 – Performance of TabularNCD on the novel classes. Best results are in bold.

| Dataset | Method | BACC (%) | ACC (%) | NMI | ARI |
|---|---|---|---|---|---|
| MNIST | Baseline | 57.7±4.7 | 57.6±4.5 | 0.37±0.2 | 0.31±0.3 |
| | Spect. clust | - | - | - | - |
| | $k$-means | 60.1±0.0 | 61.1±0.0 | 0.48±0.0 | 0.38±0.0 |
| | TabularNCD | **91.5±4.1** | **91.4±4.2** | **0.82±0.06** | **0.81±0.04** |
| Forest | Baseline | 55.6±2.0 | 68.5±1.4 | 0.27±0.02 | 0.15±0.01 |
| | Spect. clust | 32.1±1.4 | 85.8±4.0 | 0.01±0.01 | 0.09±0.01 |
| | $k$-means | 32.9±0.0 | 62.0±0.0 | 0.04±0.00 | 0.05±0.00 |
| | TabularNCD | **66.8±0.6** | **92.2±0.2** | **0.37±0.09** | **0.56±0.09** |
| Letter | Baseline | 55.7±3.6 | 55.9±3.6 | 0.49±0.04 | 0.33±0.04 |
| | Spect. clust | 45.3±4.0 | 45.3±4.0 | 0.48±0.03 | 0.18±0.03 |
| | $k$-means | 50.2±0.6 | 49.9±0.6 | 0.40±0.01 | 0.28±0.01 |
| | TabularNCD | **71.8±4.5** | **71.8±4.5** | **0.60±0.04** | **0.54±0.04** |
| Human | Baseline | 80.0±0.5 | 78.0±0.6 | 0.64±0.01 | 0.62±0.01 |
| | Spect. clust | 70.2±0.0 | 69.4±0.0 | 0.72±0.00 | 0.60±0.00 |
| | $k$-means | 75.3±0.0 | 77.0±0.0 | 0.62±0.00 | 0.59±0.00 |
| | TabularNCD | **98.9±0.2** | **99.0±0.2** | **0.95±0.01** | **0.97±0.01** |
| Satimage | Baseline | 53.8±3.4 | 53.9±4.2 | 0.25±0.03 | 0.22±0.03 |
| | Spect. clust | 82.2±0.1 | 77.8±0.1 | 0.51±0.00 | 0.46±0.00 |
| | $k$-means | 73.7±0.3 | 69.2±0.2 | 0.30±0.00 | 0.28±0.00 |
| | TabularNCD | **90.8±4.0** | **91.4±5.0** | **0.71±0.11** | **0.79±0.07** |
| Pendigits | Baseline | 72.8±5.5 | 72.8±5.4 | 0.62±0.06 | 0.54±0.07 |
| | Spect. clust | 84.0±0.0 | 84.0±0.0 | **0.78±0.00** | 0.67±0.00 |
| | $k$-means | 82.5±0.0 | 82.5±0.0 | 0.72±0.00 | 0.63±0.00 |
| | TabularNCD | **85.5±0.7** | **85.6±0.8** | 0.76±0.02 | **0.71±0.02** |
| Census | Baseline | 53.0±3.5 | **55.0±6.5** | 0.49±0.02 | 0.30±0.03 |
| | Spect. clust | 23.6±3.3 | 51.3±5.5 | 0.24±0.11 | 0.18±0.09 |
| | $k$-means | 38.5±2.6 | 49.8±3.6 | 0.41±0.05 | 0.28±0.03 |
| | TabularNCD | **61.9±0.6** | 50.1±0.9 | 0.48±0.01 | 0.30±0.00 |

The standard deviation is computed over 10 executions. The 2 unsupervised clustering methods (Spect. clust and $k$-means) are only fitted to the test instances belonging to the novel classes. Values for the spectral clustering of MNIST are missing as the execution did not complete under 1 hour.

**Comparison to the competing methods.** In Table 3.2, we report the performance of the 4 competing methods and on the 7 datasets for the clustering task. The results show

that TabularNCD achieves higher performance than the baseline and the two unsupervised clustering methods, on all considered datasets and on all metrics. The improvements in accuracy over competing methods ranges between 1.6% and 21.0%. This proves that even for tabular data, useful knowledge can be extracted from already discovered classes to guide the novel class discovery process.

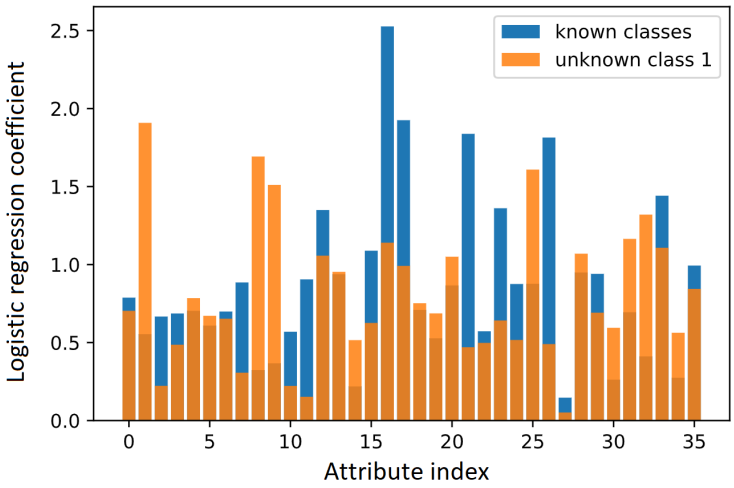



Figure 3.4 – Per-attribute coefficients of a logistic regression trained to predict the known and novel classes. The dataset is *Satimage*.

While the baseline competitor ($k$-means on a projection learned on the labeled data, see Section 3.4.1) improves the performance of $k$-means for some datasets (notably the Census and Forest datasets), it still lags behind our method in general. The baseline obtains a lower score than the simple $k$-means on only 3 datasets, which means that in some cases, the features extracted from the known classes are insufficient to discriminate novel unseen data. This is the case for the *Satimage* dataset, where the performance of the baseline is much lower compared to the simple $k$-means on the original data. To understand this phenomenon, we compare the importance of the features used to predict the known classes to the importance of the features of the novel classes. Because the features have been standardized, we can simply look at the coefficients of a logistic regression. Fig. 3.4 illustrates this experiment: the 5 most important coefficients in the prediction of one of the novel classes are more than twice as large as the addition of the coefficients of the attributes used to predict the known classes. Because these attributes that are critical in the discrimination of this novel class are relatively unimportant in the discrimination of the known classes, they are lost during the training of the classifier that serves as a feature extractor in the baseline, which results in a projection of the unlabeled instances

of poor quality. This issue is more pronounced as the known and novel classes are more dissimilar. The latent representation of TabularNCD is pre-trained using SSL on all the available data, which explains why it is unaffected by the phenomenon described above.

Table 3.3 – Performance of the proposed method against CD-KNet [38]. Best results are in bold.

| Dataset | MNIST | | |
|---|---|---|---|
| Method | ACC (%) | NMI | ARI |
| TabularNCD | 91.4 | 0.823 | 0.812 |
| CD-KNet | 86.9 | 0.683 | 0.707 |
| CD-KNet-Exp | **94.5** | **0.856** | **0.869** |

**Comparison to a NCD method on MNIST.** CD-KNet-Exp [38] (for *Class Discovery Kernel Networks with Expansion*) is one of the first methods that attempts to solve the NCD problem. They define their problem as an Open-World Class Discovery problem, which is actually another name for Novel Class Discovery. In their proposed method, they start by training a classifier on the known classes of $D^l$, and then fine-tune it on both $D^l$ and $D^u$ before applying a $k$-means on the learned latent representation to get the cluster assignments. Similarly to our experimental protocol, they select the first 5 digits of MNIST as known classes and use the last 5 as new unlabeled classes to discover. For this reason, we can directly compare their reported performance to ours in Table 3.3. From this table, we see that in terms of clustering accuracy, our method performs 4.5% better than their simpler implementation CD-KNet, and 3.1% worse than their complete approach CD-KNet-Exp, where the previously learned classifier is re-trained with the pseudo-labels assigned using $k$-means. Obtaining comparable results on MNIST is very encouraging because unlike our approach, CD-KNet-Exp takes advantage of convolutional neural networks as they only use image data. Their re-training technique could also be explored in future work to improve performance.

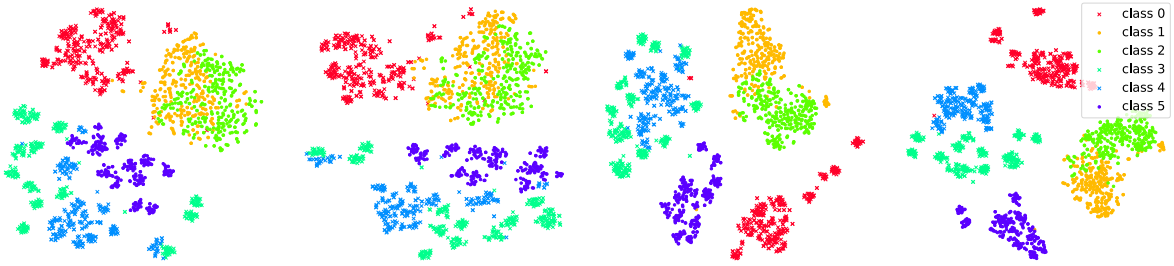
**Ablation study.** The usefulness of each of the components of the proposed method is estimated in Table 3.4 by ablating them and comparing the metrics after training to the metrics of full method. The first observation that can be made is that each component has a positive influence, and removing any of them results in a drop in performance. The most important one is the BCE (3.5), since without it the clustering network degenerates to a trivial solution where it predicts the same class for all observations. Next is the MSE from the consistency loss (3.7). The substantial drop in performance was expected as its role is crucial to (1) reduce the moving target phenomenon introduced in Section 3.3.3 and

Table 3.4 – Ablation study of the proposed method.

| Method | BACC (%) | ACC (%) | NMI | ARI |
|---|---|---|---|---|
| TabularNCD | 90.8±4.0 | 91.4±5.0 | 0.71±0.11 | 0.79±0.07 |
| - w/o SSL | 88.4±5.3 | 88.6±7.0 | 0.67±0.15 | 0.67±0.10 |
| - w/o CE | 72.0±6.1 | 69.5±6.0 | 0.44±0.12 | 0.49±0.08 |
| - w/o BCE | 33.3±0.0 | 51.7±0.0 | 0.00±0.00 | 0.00±0.00 |
| - w/o MSE | 66.7±5.7 | 63.9±4.4 | 0.44±0.02 | 0.37±0.02 |

**SSL:** Self-Supervised Learning, **CE:** Cross Entropy loss of the classification network, **BCE:** Binary Cross Entropy loss of the clustering network, **MSE:** Mean Squared Error consistency loss. The dataset is Satimage [127].

(2) regularize the network to improve its generalization. We also observe an important decrease in performance when removing the CE (3.5), meaning that in the case of the Satimage dataset, jointly learning a classification network with the clustering network helps guide the clustering process as intended. Finally, while it is beneficial to pre-train the encoder with SSL, the impact is small. This can be explained in part by the limited success that SSL works have had in the challenging area of tabular data.




(a) Original data (b) After SSL (c) Joint training epoch 15 (d) Joint training epoch 30

Figure 3.5 – Evolution of the t-SNE during the joint training of the model on the *Human Activity Recognition* dataset.

**Visualization.** In addition to quantitative results, we also report a qualitative analysis showing the feature space that is learned. In Fig. 3.5, we visualize the evolution of the representation of all the classes during the training process of the proposed method. Fig 3.5(a) corresponds to the original data, while the next figures are the data projected in the latent space. Classes overlap in the original representation, as well as in the latent space after Self-Supervised Learning (Fig 3.5(b)). However, the SSL step initialized the

encoder to a reasonable representation. This means that at the beginning of the joint training, the pseudo-labels defined on the unlabeled data using (3.6) will be accurate. During the joint training in the Figures 3.5(c) and 3.5(d), the classes are more and more separated, showing that the proposed method is able to successfully discover novel classes using knowledge from known classes. The plot clearly shows that our model produces feature representations where samples of the same class are tightly grouped.

Finally, we note that additional experiments can be found in Appendix A.5, where we attempt to automatically determine the optimal value of *top* $k$ in the pseudo-labeling process during training. The general idea is to apply the pseudo-labeling method to all the points in a mini-batch and find the *top* $k$ value that minimizes the error on the labeled data by comparing its results to the ground truth labels. Four distinct approaches are analyzed in two different settings, and the results are promising.

## 3.5 Conclusion and future work

In this chapter, we have proposed a first solution to the Novel Class Discovery problem in the challenging environment of tabular data. We demonstrated the effectiveness of our proposed approach, TabularNCD, through extensive experiments and careful analysis on 7 public datasets against unsupervised clustering methods. The greater performance of our method has shown that it is possible to extract knowledge from already discovered classes to guide the discovery process of novel classes, which demonstrates that NCD is not only applicable to images but also on tabular data. Lastly, the original method of defining pseudo-labels proposed here has proven to be reliable even in the presence of unbalanced classes.

However, while the results of this first method are convincing, it still has a number of limitations. Namely, the assumption that the number of novel classes is known in advance is impractical and has already been identified as an important issue in Chapter 2. Additionally, the many hyperparameters of our method were optimized using the labels of the novel classes, which are not available in practice. These issues will be the main focus of Chapter 4, where we aim to solve NCD in more realistic scenarios.

Chapter 4

# A Practical Approach to Novel Class Discovery in Tabular Data

## Contents




This chapter has been published in an international journal, which is part of the journal track of the *ECML/PKDD* conference: [23] A Practical Approach to Novel Class Discovery in Tabular Data. Colin Troisemaine, Alexandre Reiffers-Masson, Stéphane Gosselin, Vincent Lemaire and Sandrine Vaton. In: *Data Mining and Knowledge Discovery*, 2024.

## 4.1 Introduction

Recently, remarkable progress has been achieved in supervised tasks, in part with the help of large and fully labeled sets such as ImageNet [92]. These advancements have predominantly focused on closed-world scenarios, where, during training, it is presumed that all classes are known in advance and have some labeled examples. However, in practical applications, obtaining labeled instances for all classes of interest can be a difficult task due to factors such as budget constraints or lack of comprehensive information. Furthermore, for models to be able to transfer learned concepts to new classes, they need to be designed with this in mind from the start, which is rarely the case. Yet this is an important skill that humans can use effortlessly. For example, having learnt to distinguish a few animals, a person will easily be able to recognise and "cluster" new species they have never seen before. The transposition of this human capacity to the field of machine learning could be a model capable of categorizing new products in novel categories.

This observation has led researchers to formulate a new problem called Novel Class Discovery (NCD) [20, 29]. Here, we are given a labeled set of known classes and an unlabeled set of different but related classes that must be discovered. Lately, this task has received a lot of attention from the community, with many new methods such as AutoNovel [17], OpenMix [45] or NCL [39] and theoretical studies [110, 40]. However, the majority of these works tackle the NCD problem under the unrealistic assumption that the number of novel classes is known in advance, or that the target labels of the novel classes are available for hyperparameter optimization [21]. This is, for example, the case with the method proposed in Chapter 3. These assumptions render these methods impractical for real-world NCD scenarios. To address these limitations, we propose a general framework for optimizing the hyperparameters of NCD methods where the ground truth labels of novel classes are never used, as they are not available in real-world NCD scenarios. Furthermore, we show that the latent spaces obtained by such methods can be used to accurately estimate the number of novel classes.

We also introduce three new NCD methods. Two of them are unsupervised clustering algorithms modified to leverage the additional information available in the NCD setting. The first one improves the centroid initialization step of $k$-means, resulting in a fast and easy to use algorithm that can still give good results in many scenarios. The second method focuses on optimizing the parameters of the Spectral Clustering (SC) algorithm. This approach has a potentially higher learning capacity as the representation itself (i.e.

the spectral embedding) is tuned to easily cluster the novel data. Finally, the last approach is a deep NCD method composed of only the essential components necessary for the NCD problem. Compared to SC, this method is more flexible in the definition of its latent space and effectively integrates the knowledge of the known classes.

While these contributions can be applied to any type of data, our work focuses on tabular data. The NCD community has focused almost exclusively on computer vision problems and, to the best of our knowledge, only one paper [21] has tackled the problem of NCD in the tabular context. This is despite the many applications of NCD in tabular data. Some examples include *cybersecurity*, where network traffic could be used to infer new types of cyber-attacks that deviate from known attack patterns; *sensor data analysis*, where new categories of equipment failures could be diagnosed and lead to faster repairs; or *fraud detection*, where new types of fraudulent transactions that deviate from known patterns could be discovered. However, this previous work [21] required the meticulous tuning of a large number of hyperparameters to achieve optimal results. Methods designed for tabular data cannot take advantage of powerful techniques commonly employed in computer vision. Examples include convolutions, data augmentation or Self-Supervised Learning methods such as DINO [69], which have been used with great success in NCD works [36, 57, 133], thanks to their strong ability to obtain representative latent spaces without any supervision. On the other hand, tabular data methods have to rely on finely tuned hyperparameters to achieve optimal results. For this reason, we believe that the field of tabular data will benefit the most from our contributions.

By making the following contributions, we demonstrate the feasibility of solving the NCD problem with tabular data and under realistic conditions:

— We develop a hyperparameter optimization procedure tailored to transfer the results from the known classes to the novel classes with good generalization.

— We show that it is possible to accurately estimate the number of novel classes in the context of NCD, by applying simple clustering quality metrics in the latent space of NCD methods.

— We modify two classical unsupervised clustering algorithms to effectively utilize the data available in the NCD setting.

— We propose a simple and robust method, called PBN (for *Projection-Based NCD*), that learns a latent representation that incorporates the important features of the known classes, without overfitting on them.

The code is available at `https://github.com/Orange-OpenSource/PracticalNCD`.

## 4.2 Related work

The setup of NCD [20], which involves both labeled and unlabeled data, can make it difficult to distinguish from the many other domains that revolve around similar concepts. In this section, we review some of the most closely related domains and try to highlight their key differences in order to provide the reader with a clear and comprehensive understanding of the NCD domain.

**Semi-supervised Learning** is another domain that is at the frontier between supervised and unsupervised learning. Specifically, a labeled set is given alongside an unlabeled set containing instances that are assumed to be from the same classes. Semi-supervised Learning can be particularly useful when labeled data is scarce or annotation is expensive. As unlabeled data is generally available in large quantities, the goal is to exploit it to obtain the best possible generalization performance given limited labeled data.

The main difference with NCD is that the all the classes are known in advance. Some works have shown that the presence of novel classes in the unlabeled set negatively impacts the performance of Semi-Supervised Learning models [89, 91]. So as these works do not attempt to discover the novel classes, they are not applicable to NCD.

**Transfer Learning** aims at solving a problem faster or with better performance by leveraging knowledge from a different problem. It is commonly expressed in computer vision by pre-training models on ImageNet [92]. Transfer Learning can be either *cross-domain*, when a model trained on a given dataset is fine-tuned to perform the same task on a different (but related) dataset. Or it can be *cross-task*, where a model that can distinguish some classes is re-trained for other classes of the same domain.

NCD can be viewed as a cross-task Transfer Learning problem where the knowledge from a classification task on a source dataset is transferred to a clustering task on a target dataset. But unlike NCD, Transfer Learning typically requires the target spaces of both sets to be known in advance. Initially, NCD was characterized as a Transfer Learning problem (e.g. in DTC [37] and MCL [42]) and the training was done in two stages: first on the labeled set and then on the unlabeled set. This methodology seemed natural, as with Transfer Learning, both sets are not available at the same time.

The field of **Open Set Recognition** (OSR) [103] stems from the observation that neural networks have a tendency to make predictions with high confidence, even for instances of classes they have never seen before. It is based on the Open-World assumption in which instances of new classes may appear during inference, so standard models can-

not be used. Therefore, in OSR, samples from novel classes must be correctly identified as unknown. An example would be a fingerprint recognition model that needs to reject images of fingerprints (or anything else!) from people who are not registered. However, as in many domains under the Open-World assumption, the goal of OSR is only to identify samples from novel classes. In NCD, we assume that this identification has already been made, and we try to actually discover the novel classes.

**Generalized Category Discovery** (GCD) was first introduced by [36] and has also attracted attention from the community [58, 60, 57]. It can be seen as a less constrained alternative to NCD, since it does not rely on the assumption that samples belong exclusively to the novel classes during inference. However, this is a more difficult problem, as the models must not only cluster the novel classes, but also accurately differentiate between known and novel classes while correctly classifying samples from the known classes.

Some notable works in this area include ORCA [121] and OpenCon [77]. Namely, ORCA trains a discriminative representation by balancing a supervised loss on the known classes and unsupervised pairwise loss on the unlabeled data. And OpenCon proposes a contrastive learning framework which employs Out-Of-Distribution strategies to separate known vs. novel classes. Its clustering strategy is based on moving prototypes that enable the definition of positive and negative pairs of instances.

**Novel Class Discovery** has a rich body of papers in the domain of computer vision. Early works approached this problem in a *two-stage* manner. Some define a latent space using only the known classes, and project the unlabeled data into it (DTC[37] and MM [19]). Others train a pairwise labeling model on the known classes and use it to label and then cluster the novel classes (CCN [29] and MCL [42]). But both of these approaches suffered from overfitting on the known data when the high-level features were not fully shared by the known and novel classes.

Today, to alleviate this overfitting, the majority of approaches are *one-stage* and try to transfer knowledge from labeled to unlabeled data by learning a shared representation. In this category, AutoNovel [17] is one of the most highly influential works. After pre-training their latent representation with SSL [51], two classification networks are jointly trained. The first simply learns to distinguish the known classes with the ground truth labels. And the other learns to separate unlabeled data from pseudo-labels defined for each epoch based on pairwise similarity. NCL [39] adopts the same architecture as AutoNovel, and extends the loss by adding a contrastive learning term to encourage the separation of novel classes. OpenMix [45] utilizes the MixUp strategy to generate more robust pseudo-labels.

As expressed before, although these methods have achieved some success, they are not applicable to tabular data. To date, and to the best of our knowledge, only TabularNCD [21] tackles this problem. Also inspired by AutoNovel, it pre-trains a dense-layer autoencoder with SSL and adopts the same loss terms and dual classifier architecture. Pseudo-labels are defined between pairs of unlabeled instances by checking if they are among the most similar pairs.

For a more complete overview of the state-of-the-art of NCD, we refer the reader to the survey [20].

## 4.3 Approaches

In this section, after introducing the notations, we define two simple but potentially effective models derived from classical clustering algorithms (Sections 4.3.2 and 4.3.3). The idea is to use the labeled data to improve the unsupervised clustering process, and make the comparison to NCD methods more challenging. Then, we present a new method, PBN (for Projection-Based NCD, Section 4.3.4), characterized by its low number of hyperparameters needed to be tuned.

### 4.3.1 Problem setting

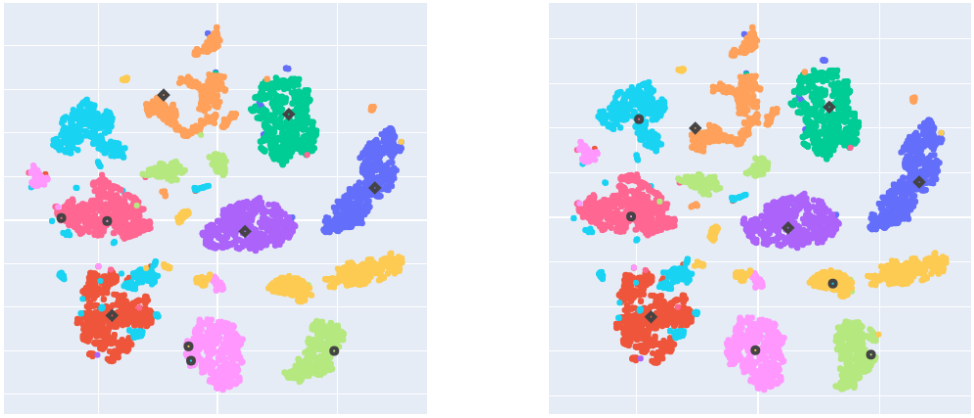
We begin by describing the Novel Class Discovery setup and the necessary notations. Here, data is provided in two distinct sets: a labeled set of known classes $D^l = \{(x_i^l, y_i^l)\}_{i=1}^N$ where each instance $x_i^l$ belongs to $X^l \in \mathbb{R}^{d \times N}$ and has a one-hot label $y_i^l \in \mathcal{Y}^l = \{1, \ldots, C^l\}$. And an unlabeled set $D^u = \{x_i^u\}_{i=1}^M$ where only the data samples $x_i^u$ of $X^u \in \mathbb{R}^{d \times M}$ are available. The objective is to exploit the knowledge from $D^l$ to accurately cluster $D^u$ by predicting the labels $y_i^u \in \mathcal{Y}^u = \{1, \ldots, C^u\}$ of the novel classes. In NCD, the novel classes are different but related to the known classes, with no overlap between these two sets of classes (i.e., $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$).

Following previous research, we first assume that the number of novel classes $C^u$ is known in advance, and later propose an approach to estimate this number in a later section.

### 4.3.2 NCD k-means

This is a straightforward method that takes inspiration from $k$-means++ [134], which is an algorithm for choosing the initial positions of the centroids (or cluster centers). In $k$-means++, the first centroid is chosen at random from the data points. Then, each new centroid is chosen iteratively from one of the data points with a probability proportional to the squared distance from the point's closest centroid. The resulting initial positions of the centroids are generally spread more evenly, which yields appreciable improvement in the final error of k-means and convergence time.

As shown in Figure 4.1(a), we naively adapt $k$-means++ to the NCD setting by defining $C^l$ initial centroids. They are set as the mean class points of the known classes using the ground truth labels. Then, we follow $k$-means++ and randomly select $C^u$ new centroids in the unlabeled set, with similarly decreasing probability when closer to existing centroids. We found experimentally (see Appendix B.6) that, after the initialization is complete, the best accuracy is achieved when only the centroids of the novel classes are updated, and using the unlabeled data only. In other words, the data of the known classes is only used during the initialization of the new centroids, but not during the convergence phase. Intuitively, if the centroids of the known and novel data are updated together, they have a higher risk of drifting and capturing data of the other set. The pseudocode of the proposed method is summarized in Algorithm 2 of Appendix B.1.



(a) Before convergence (b) After convergence

Figure 4.1 – t-SNE plots of the *Pendigits* dataset depicting the centroids before and after convergence. Note how the centroids of the known classes (the squares) do not move, as they stay the mean class point.

Similarly to $k$-means++, we repeat the initialization process a few times and keep the

centroids that achieved the smallest inertia. Note that, to stay consistent with the $k$-means algorithm, we use also the $L_2$ norm (i.e. the Euclidean distance) for NCD $k$-means.

### 4.3.3 NCD Spectral Clustering

Spectral Clustering (SC) is an alternative to distance-based clustering methods (such as $k$-means). It makes no assumptions about the structure of the data and considers the clustering problem as a graph partitioning problem and seeks to decompose the graph into connected components [129]. The input of the Spectral Clustering algorithm is an adjacency matrix (sometimes called similarity graph) which must accurately represent the neighborhood relationships between data points. There are multiple ways to construct such a graph, however there is no theoretical result on the relation between the similarity graph construction method and the Spectral Clustering results. Here, we employ a popular approach to construct the adjacency matrix, which is through a Gaussian kernel: $A_{i,j} = \exp\left(-\|x_i - x_j\|_2^2/(2\sigma^2)\right), \forall x_i, x_j \in X$ where the parameter $\sigma$ controls the width of the neighborhood.

Following the Ng-Jordan-Weiss algorithm [135], we use the symmetric normalized Laplacian $L_{sym} = D^{-1/2}LD^{-1/2}$. Here, the degree matrix $D$ is a diagonal matrix with degrees $d_1, \cdots, d_n$ defined as $d_i = \sum_{j=1}^{n} A_{i,j}$. $n$ is the number of data samples and $L = D - A$ is the unnormalized Laplacian matrix. The next step consists in finding the first $u$ eigenvectors of $L_{sym}$ to form the spectral embedding $U \in \mathbb{R}^{n \times u}$, where row $i$ corresponds to point $x_i$. Finally, the points in $U$ are partitioned with $k$-means into clusters.

The optimal value of $\sigma$ in the Gaussian kernel can vary widely depending on the distribution of inter-point distances. For this reason, we take inspiration from the rules of thumb given in [129] and employ a minimum spanning tree (MST) to choose $\sigma$. In the past few years, several graph-based clustering methods that use the MST have been proposed [136], as it reliably represents the layout of the data and is inexpensive to compute. In the approach proposed here, we denote $d_{max}$ the length of the longest edge in the MST of inter-point distances. The longest edge of the MST is a much studied object [137] and is representative of the scale of the dataset. The scaling factor $\sigma$ is then calculated such that, after applying the Gaussian kernel, $d_{max}$ is transformed into a chosen similarity $s_{min}$. This ensures that the resulting graph is safely connected. By optimizing this similarity $s_{min}$ we can accurately represent the neighborhood relationships.

Therefore, for a given value of $s_{min}$, we derive $\sigma$ from the length of the longest edge

$d_{max}$ in the MST:

$$\exp\left(-d_{max}^2/(2\sigma^2)\right) = s_{min}$$
$$\Leftrightarrow \sigma = d_{max}/\sqrt{-2\ln\left(s_{min}\right)} \tag{4.1}$$

Optimizing $s_{min}$ instead of $\sigma$ should be more robust to variations in the distribution of inter-point distances and give better results across different datasets or parts of a dataset.

To incorporate the knowledge from the known classes in the Spectral Clustering process, our initial approach was to utilize NCD $k$-means within the spectral embedding. In short, we would initially compute the full spectral embedding for all the data and determine the mean points of the known classes with the help of the ground truth labels. These mean points would then serve as the initial centroids. However, the observed performance improvement over the fully unsupervised SC was quite marginal.

Instead, the idea that we will use throughout this chapter (and that we refer to as "NCD Spectral Clustering") stems from the observation that SC can obtain very different results according to the parameters that are used. Among these parameters, the temperature parameter $\sigma$ of the kernel holds particular importance, as it directly impacts the adjacency matrix's accuracy in representing the neighborhood relationships of the data points. The rule of thumb of Equation 4.1 still requires to choose a value, but significantly reduces the space of possible values. Additionally, while the literature often sets the number of components $u$ of the spectral embedding equal to the number of clusters, we have observed that optimizing it can also improve performance.

Therefore, rather than a specific method, we propose the parameter optimization scheme illustrated in Figure 4.2. For a given combination of parameters $\{s_{min}, u\}$, the

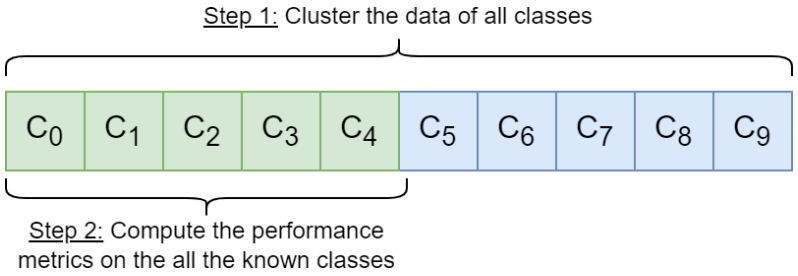



Figure 4.2 – NCD Spectral Clustering parameter optimization process.

corresponding spectral embedding of all the data is computed and then partitioned with $k$-means. The quality of the parameters is evaluated from the clustering performance on

the known classes, as the ground truth labels are only available for the known classes. Indeed an important hypothesis behind the NCD setup is that the known and novel classes are related and share similarities, so they should have similar feature scales and distributions. Consequently, if the Spectral Clustering performs well on the known classes, the parameters are likely suitable to represent the novel classes. The pseudocode of this approach can be found in Algorithm 3 of Appendix B.1.

**Discussion.** This idea can be applied to optimize the parameters of any unsupervised clustering algorithm in the NCD context. For example, the $Eps$ and $MinPts$ parameters of DBSCAN [109] can be selected in the same manner. It is also possible to use a different adjacency matrix in the SC algorithm. One option could be to substitute the Gaussian kernel with the $k$-nearest neighbor graph, and therefore optimise $k$ instead of $\sigma$. However, for the sake of simplicity, we will only investigate SC using the Gaussian kernel.

### 4.3.4 Projection-Based NCD

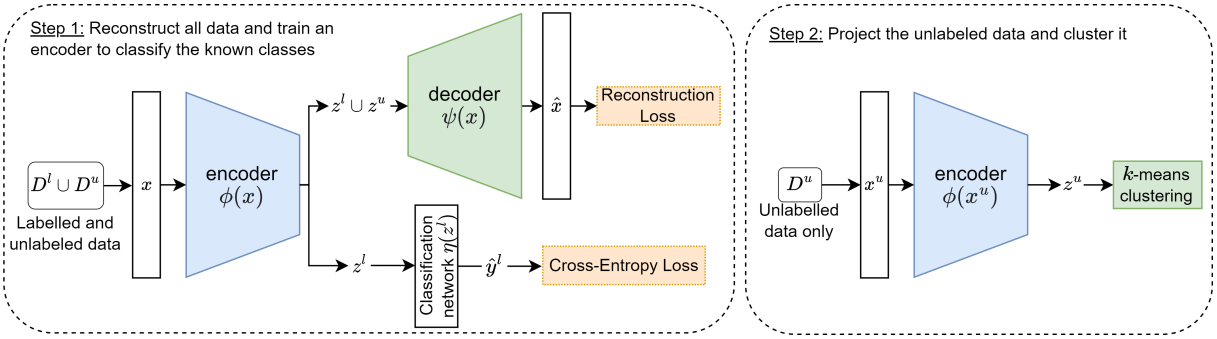



Figure 4.3 – Architecture of the PBN model.

Projection-based NCD (PBN) can be seen as an extension of the baseline method used in TabularNCD [21]. PBN is illustrated in Figure 4.3 and consists of 3 key components: (1) an encoder that learns a shared representation between the known and novel classes; (2) a classification network trained to distinguish the known classes of $D^l$ in order to incorporate their relevant features into the representation; and (3) a decoder that reconstructs the data for both known and novel classes of $D^l \cup D^u$, ensuring that the latent space contains the information necessary to represent all classes. The decoder serves a dual purpose: it provides regularization and mitigates overfitting on the known classes, thus improving generalization, as shown in [138].

The training loss is defined as:

$$\mathcal{L}_{PBN} = w \times \mathcal{L}_{CE} + (1 - w) \times \mathcal{L}_{MSE} \tag{4.2}$$

where $w \in (0, 1)$ is a trade-off parameter that allows to balance the strength of the cross-entropy loss and the reconstruction loss.

The cross-entropy loss on the known classes is defined as:

$$\mathcal{L}_{CE} = -\sum_{c=1}^{C^l} y_c \log\left(\eta_c(z)\right) \tag{4.3}$$

where $\eta(z) = (\eta_c(z))_{c=1}^{C^l}$ is the output of a classification network composed of a single dense layer of neurons, $z = \phi(x)$ is the projection of instance $x$ through the encoder $\phi$ and $(y_c)_{c=1}^{C^l}$ is the one-hot encoded ground truth label of instance $x$.

The reconstruction loss of the instances from all classes is written as:

$$\mathcal{L}_{MSE} = \frac{1}{d}\sum_{j=1}^{d}\left(x_j - \hat{x}_j\right)^2 \tag{4.4}$$

where $\hat{x} = \psi(z)$ is the reconstruction of instance $x \in \mathbb{R}^d$.

Once the encoder, decoder and classification network have been trained (step 1), unlabeled data $D^u$ is projected by the trained encoder into the latent space and then clustered with $k$-means to discover novel classes (step 2).

Projection-based NCD requires tuning of four hyperparameters. The trade-off parameter $w$ is inherent to the method and the other three come from the choice of architecture: the learning rate, the dropout rate and the size of the latent space.

Note that this method does not employ complex schemes to define pseudo-labels unlike many NCD works. They have been proven to be accurate with image data (notably thanks to data augmentation techniques) [17, 48], but we found in preliminary results not detailed here, that for tabular data, they introduce variability in the results and new hyperparameters that need to be tuned.

**Discussion.** Similarly to PBN, the baseline method of TabularNCD [21] relies on the assumption that known and novel classes share similar high-level features, and defines a latent space that highlights these features. This baseline first trains a deep classifier to distinguish only the known classes of $D^l$. After training, the output and softmax layers are discarded, and the last hidden layer is now considered as the output of an encoder. It

then projects the novel data of $D^u$ into this latent space and partitions it using $k$-means. This is the basic workflow of two-stage latent space-based NCD methods identified in [20]. It is also similar to DTC [37], which uses the more refined DEC [43] clustering model instead of $k$-means in the baseline. The problem with such two-stage methods is that the resulting representations are at risk of being heavily biased towards the known classes. Thus, if some concepts or high-level features are not shared between the known and novel classes, the novel classes will not be well represented and these approaches will fail.

### 4.3.5 Summary of proposed approaches

In this section, we have proposed 3 distinct methods for solving the NCD problem, all of which leverage knowledge from the known classes in different ways. Firstly, NCD $k$-means uses the labeled data to improve the initialization of its centroids. Secondly, instead of using the labels of the known classes during the clustering process itself, NCD Spectral Clustering uses them to find parameters that are likely to be suitable for the whole domain. More precisely, by clustering $D^l \cup D^u$ together, the adequacy of the parameters $s_{min}$ and $u$ can be evaluated on the known classes. Finally, PBN is a straightforward method that includes only the essential components to define a latent representation suitable for clustering the novel classes. In this case, an encoder is trained with a classification loss on the known classes and a reconstruction loss on all the data to ensure that the novel classes are not misrepresented. The novel data is then projected into this representation and clustered with $k$-means.

In the next section, we present an approach to finding hyperparameters without using the labels of the novel classes, which are not available in realistic scenarios. Indeed in the experiments (see Section 4.7), it should become clear why the simplicity of the proposed approach is a desirable feature for hyperparameter optimization in the NCD context.

## 4.4 Hyperparameter optimization

The success of machine learning algorithms (including NCD) can be attributed in part to the high flexibility induced by their hyperparameters. In most cases, a target is available and approaches such as the $k$-fold Cross-Validation (CV) can be employed to tune the hyperparameters and achieve optimal results. However, in a realistic scenario of Novel Class Discovery, the labels of the novel classes are never available. We must therefore

find a way to optimize hyperparameters without ever relying on the labels of the novel classes. In this section, we present a method that leverages the known classes to find hyperparameters applicable to the novel classes. This tuning method is designed specifically for NCD algorithms that require both labeled data (known classes) and unlabeled data (novel classes) during training [1]. This is the case for Projection-based NCD, as described in Section 4.3.4.

The process that we devised is represented in Figure 4.4. For each of the splits, the instances of around half of the known classes are selected to form the set $D^{hid}$ and their labels are hidden. The labeled set now becomes the instances of $D^l \setminus D^{hid}$ and the unlabeled set becomes the instances of $D^u \cup D^{hid}$. After training the model with this new data split, it is evaluated for its performance for partitioning the instances of $D^{hid}$ only since their labels are available.

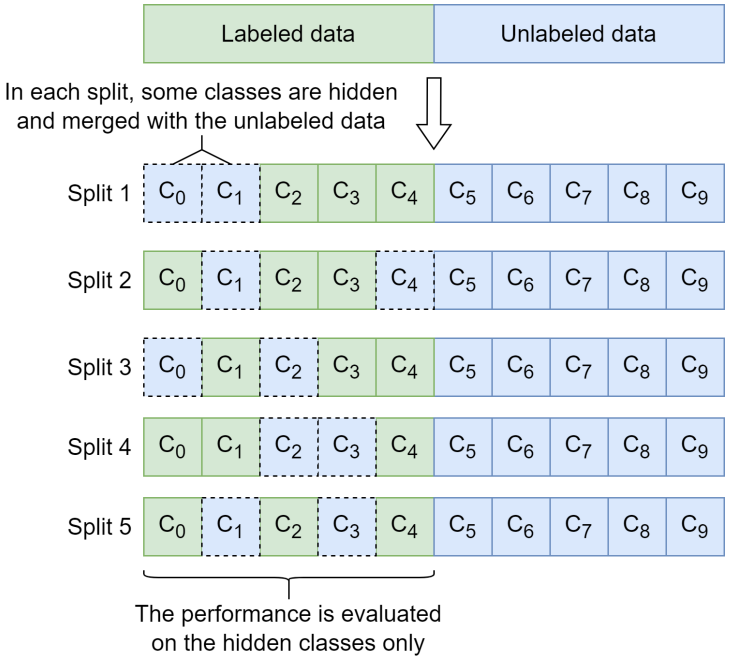



Figure 4.4 – The $k$-fold cross-validation approach for hyperparameter optimisation of NCD methods.

To illustrate, in the split 1 of Figure 4.4, the model will be trained with the subsets of classes $\{C_2, C_3, C_4\}$ as known classes and $\{C_0, C_1, C_5, \ldots, C_9\}$ as novel classes. It will be evaluated for its performance on the hidden classes $\{C_0, C_1\}$ only.

1. To optimize purely unsupervised clustering methods for NCD, we refer the reader to the optimization process of Section 4.3.3.

To evaluate a given combination of hyperparameters, this approach is applied to all the splits, and the performance on the hidden classes is averaged. After repeating this process for many combinations, the combination that achieved the best performance is selected. For the final evaluation on the novel classes, in a realistic scenario of NCD their labels are never available. However, in the datasets employed in this chapter, the novel classes are comprised of pre-defined classes. Therefore, even though these labels are not employed during training, they can still be used to assess the final performance on the novel classes of different models and compare them against each other.

This tuning method stems from the same idea behind the NCD Spectral Clustering parameterization process. Namely, if the clustering in the learned representation successfully partitions the hidden classes in $D^{hid}$, it is also likely suitable for the novel classes in $D^u$. Furthermore, keeping the unlabeled data during training even though the model is not evaluated on the novel classes is important, as it increases the chances of the representation being adapted for all the classes. For the same reason, the $k$-means of PBN (see Section 4.3.4) is fitted on $D^u \cup D^{hid}$ together (instead of just $D^{hid}$) and the performance is then computed on $D^{hid}$ only. So cases where the classes in $D^u$ and $D^{hid}$ are tangled will be penalized.

In Table 4.1, we report for all datasets used in our experiments the number of known classes that are hidden in each split, as well as the number of splits. Note that when the number of known classes is small (e.g. 3 for *Human*), this approach may be difficult to apply.

Table 4.1 – Classes splits of the $k$-fold cross-validation.

| Dataset | Known classes | Novel classes | Hidden classes | Splits |
|---|---|---|---|---|
| Human | 3 | 3 | 2 | 3 |
| Letter | 19 | 7 | 7 | 5 |
| Pendigits | 5 | 5 | 2 | 5 |
| Census | 12 | 6 | 6 | 5 |
| m feat | 5 | 5 | 2 | 5 |
| Optdigits | 5 | 5 | 2 | 5 |
| CNAE-9 | 4 | 5 | 2 | 5 |

**Discussion.** Similarly to NCD, there are no labels available in unsupervised clustering problems, which makes the task of hyperparameter selection very difficult. To address this issue, clustering algorithms are sometimes tuned using *internal* metrics that do not

rely on labeled data for computation. These metrics offer a means of comparing the results obtained from different clustering approaches. Examples of such metrics include the Silhouette coefficient, Davies-Bouldin index, or Calinski-Harabasz index [139]. However, it is important to note that these metrics make assumptions about the structure of the data and can be biased towards algorithms which make a similar assumption. But unlike unsupervised clustering, the NCD setting provides known classes that are related to the novel classes we are trying to cluster.

## 4.5 Estimating the number of novel classes

Cluster Validity Indices (CVIs) are commonly used in unsupervised data analysis to estimate the number of clusters and are also applicable to the NCD problem. CVIs are scores that compare the compactness and separation of clusters without the help of external information such as ground truth labels. However, the knowledge from the known classes is not used if the CVIs are directly applied to estimate the number of novel classes. Therefore, we propose to apply the CVIs in the latent representation learned by PBN. Projection-based NCD methods such as PBN are designed to create a latent space that emphasizes the relevant features of the known classes. Since these features are shared to some extent with the novel classes, this representation should be better at revealing the clusters we are trying to discover than the original feature space. Consequently, it makes sense that applying the different estimation techniques in the learned latent space should yield better results.

Note that this is only applicable to NCD methods such as PBN that do not require the number of novel classes $C^u$ to train their latent space (unlike TabularNCD). For the others, the estimation can be done once in the original feature space, but should have higher error.

Some NCD works have also previously attempted to estimate the number of novel classes. For instance, [29] defines a large number of output neurons in their clustering network (e.g. 100). In this case, the clustering network is expected to use only the necessary number of clusters while leaving the remaining output neurons unused. Clusters were counted if they contained more instances than a certain threshold. However, since, with the exception of TabularNCD, the models studied in this chapter do not use a clustering network, we will not evaluate this method.

Another technique, proposed by [36], consists in training a $k$-means on the combined dataset $D^l \cup D^u$ and selecting the $k$ that yielded the highest accuracy on $D^l$. While this approach worked well for balanced datasets [57], it has been shown to underperform in the case of unbalanced class distributions [140]. For the sake of simplicity, we will call this method KM-ACC (for $k$-means ACC) in the remainder of this chapter.

To select the CVI that we will use for our application, we rely on the results of [139]. Here, the authors conducted an extensive performance evaluation of 30 CVIs. They concluded that the Silhouette, Davies–Bouldin, Calinski–Harabasz and Dunn indices behaved better than other indices in almost all cases. In the experiments, the performance of these 4 indices will be compared, with the addition of the elbow method and the NCD-specific method KM-ACC.

## 4.6 Full training procedure

In the previous sections, we presented the models, the hyperparameter optimization and the estimation procedure of the number of novel classes independently. In this section, these components are brought together to form a complete training procedure. To ensure that no prior knowledge about the novel classes is ever used in this process, the number of novel classes is naturally estimated during the $k$-fold CV introduced in Section 4.4. As the whole process is quite complex, we try to summarize it in clear terms in this section and in Algorithm 1.

To gauge a given set of hyperparameters, we evaluate the performance of the model over $n_{folds}$, where in each fold, a random combination of known classes is "hidden" and merged with the unlabeled data of $D^u$. In a fold, the encoder of the NCD model is first trained on this new data split. The number of novel classes is then estimated with a CVI in the projection of the unlabeled data. At this point, the novel and hidden classes are partitioned by the model in the latent space using the previous estimate of the number of clusters. And the accuracy for this fold is calculated on the hidden classes. This process is repeated for all folds and for many combinations of hyperparameters, and the combination that achieved the best performance on average is selected for the final evaluation of the model.

**Algorithm 1** Agnostic NCD model evaluation

**Require:** Training data $\{D^l, D^u\}$, hyperparameters $\theta$, number of classes to hide $n_{hid}$, number of folds $n_{folds}$

**Initialize:** $folds \leftarrow$ set of $n_{folds}$ random combinations of $n_{hid}$ known classes
**for** each $fold$ **in** $folds$ **do**
  $D^{hid} \leftarrow$ the data from $D^l$ of the classes in $fold$
  $D^{l'} \leftarrow D^l \setminus D^{hid}$
  $D^{u'} \leftarrow D^u \cup D^{hid}$
  Train model on $\{D^{l'}, D^{u'}\}$ with hyperparameters $\theta$
  $Z^u \leftarrow \phi_\theta(X^u)$ the projection of the novel data
  $k' \leftarrow$ the estimation of $C^u$ in $Z^u$ with a CVI
  Get the clustering prediction of the model for $D^{u'}$ using $k = n_{hid} + k'$
  $ACC_{hid} \leftarrow$ clustering performance on $D^{hid}$
**end for**
**Return:** Average of all the $ACC_{hid}$

## 4.7 Experiments

### 4.7.1 Experimental setup

**Datasets.** To evaluate the performance of the methods compared in this chapter, 7 tabular classification datasets were selected: Human Activity Recognition [126], Letter Recognition [125], Pen-Based Handwritten Digits [127], 1990 US Census Data [127], Multiple Features [127], Handwritten Digits [127] and CNAE-9 [127].

Following the previous NCD works [39, 37, 17], the instances of about 50% of the classes are hidden a priori to form the unlabeled set of novel classes $D^u$, while the rest form the labeled set $D^l$. We use a 70/30% train/test split if it was not already provided. Statistical information on the datasets is shown in Table 4.2, and the number of known/novel classes (along with the number of classes hidden during the $k$-fold CV) can be found in Table 4.1. The numerical features of all the datasets are preprocessed to have zero mean and unit variance, while the categorical features are one-hot encoded.

**Metrics.** We report the *clustering accuracy* (ACC) on the unlabeled data. It is defined as:

$$ACC = \max_{perm \in P} \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}\left\{y_i = perm(\hat{y}_i\right\} \tag{4.5}$$

where $y_i$ and $\hat{y}_i$ are the ground truth labels and predicted labels for instance $x_i$ respec-

Table 4.2 – Details of the datasets.

| Dataset | Human | Letter | Pendigits | Census | m feat | Optdigits | CNAE-9 |
|---|---|---|---|---|---|---|---|
| Features | 562 | 16 | 16 | 67 | 515 | 62 | 856 |
| Known classes | 3 | 19 | 5 | 12 | 5 | 5 | 4 |
| Known data | 3733 | 10229 | 3777 | 12000 | 802 | 1918 | 377 |
| Novel classes | 3 | 7 | 5 | 6 | 5 | 5 | 5 |
| Novel data | 3619 | 3770 | 3717 | 6000 | 798 | 1905 | 487 |
| Test data | 1453 | 1704 | 1734 | 6000 | 202 | 905 | 113 |

tively. Here, $M = |D^u|$. $P$ is the set of all possible permutations between ground truth and predicted labels. It can be easily computed using the Hungarian algorithm [2]. The Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) are also reported in Appendix B.2.

**Competitors.** We report the performance of $k$-means and Spectral Clustering, along with their NCD-adapted versions introduced in Sections 4.3.2 and 4.3.3. We also include PBN and the pioneering NCD work for tabular data, TabularNCD [21]. Finally, we implement the same baseline used in [21]. It is a simple deep classifier that is first trained to distinguish the known classes. Then, the penultimate layer is used to project the data of the novel classes before clustering it with $k$-means. See the discussion of Section 4.3.4 for more details. We will call this approach the "baseline" for the remainder of this chapter.

**Implementation details.** The neural-network based methods (i.e. PBN, TabularNCD and the baseline) are trained with the same architecture: an encoder of 2 hidden layers of decreasing size, and a final layer for the latent space whose dimension is optimized as an hyperparameter. The dropout probability and learning rate are also hyperparameters to be optimized. The classification networks are all a single linear layer followed by a softmax layer. All methods are trained for 200 epochs and with a fixed batch size of 512. For a fair comparison, the hyperparameters of these neural-network based methods are optimized following the process described in Section 4.4 (whereas the parameters of NCD SC are simply optimized following Section 4.3.3). Thus, the labels of the novel classes are never used except for the computation of the evaluation metrics reported in the result tables. The hyperparameters of the deep-based methods are tuned by a random search of the hyperparameter space and selecting the combination which obtained the best ARI on the hidden classes. The search space and selected values can be found in Appendix B.3.

**Objectives of the experiments.** In the following section, an extensive evaluation of the clustering accuracy of the 7 competitors is performed. This evaluation is carried

out in two steps. First, when the number of novel classes is known in advance, we seek to determine whether (1) the NCD-adapted versions of $k$-means and Spectral Clustering outperform their purely unsupervised versions when labeled data is available, and (2) which of the deep-based NCD methods performs best when the labels of the novel classes are not available for hyperparameter tuning. And second, in the most realistic scenario where the number of classes is not given in advance, we compare the ability of different CVIs to accurately estimate this number and the impact on the NCD methods of using this estimation.

### 4.7.2 Results analysis

#### Results when the number of novel classes is known in advance

**Clustering.** In Table 4.3, we first examine the performance of the unsupervised clustering methods when the number of novel classes $C^u$ is known in advance. The aim is to determine which of the clustering algorithms performs best and should be compared with the NCD methods. We observe that NCD $k$-means is never worse than $k$-means, and NCD SC is only once worse than SC. This result confirms the efficacy of both NCD approaches and demonstrates that even simple clustering techniques can benefit from the known classes, although the improvements are sometimes only marginal.

The comparison between NCD $k$-means and NCD SC confirms the idea that no single clustering algorithm is universally better than the others in all scenarios, as noted by [141]. However, NCD SC outperforms its competitors on 4 occasions and has a the highest average accuracy. Therefore, this algorithm is selected for the next step of comparisons.

Table 4.3 – Test ACC of the clustering algorithms averaged over 10 runs. Best results are in bold.

| Dataset | $k$-means | NCD $k$-means | SC | NCD SC |
|---|---|---|---|---|
| Human | 75.7±0.2 | 75.9±0.0 | 76.3±0.3 | **93.1±9.7** |
| Letter | 50.7±0.2 | 51.9±2.3 | 55.9±0.0 | **57.4±5.8** |
| Pendigits | 81.7±0.0 | 81.7±0.0 | **83.0±0.0** | 81.7±2.7 |
| Census | 49.9±4.0 | **50.4±1.1** | 48.5±0.3 | 48.0±1.8 |
| m feat | 89.1±0.3 | **89.7±0.4** | 89.6±0.3 | 89.2±2.3 |
| Optdigits | 79.1±4.5 | 94.2±0.0 | 89.7±0.0 | **95.4±5.3** |
| CNAE-9 | 60.6±5.9 | 61.2±4.5 | 53.8±4.8 | **69.0±6.7** |
| Average | 69.5 | 72.1 | 71.0 | **76.3** |

**NCD.** As shown in Table 4.4, PBN outperforms both the baseline and TabularNCD by an average of 21.6% and 12.9%, respectively. It is only outperformed by the baseline on the *Letter Recognition* [125] dataset. This dataset consists of primitive numeric attributes describing the 26 capital letters in the English alphabet, which suggests a high feature correlation between the features used to distinguish the known and novel classes. Since the baseline learns a latent space that is strongly discriminative for the known classes, this gives the baseline model a distinct advantage in this specific context. On the other hand, we observe that it is at a disadvantage when the datasets do not share as many high-level features between the known and novel classes.

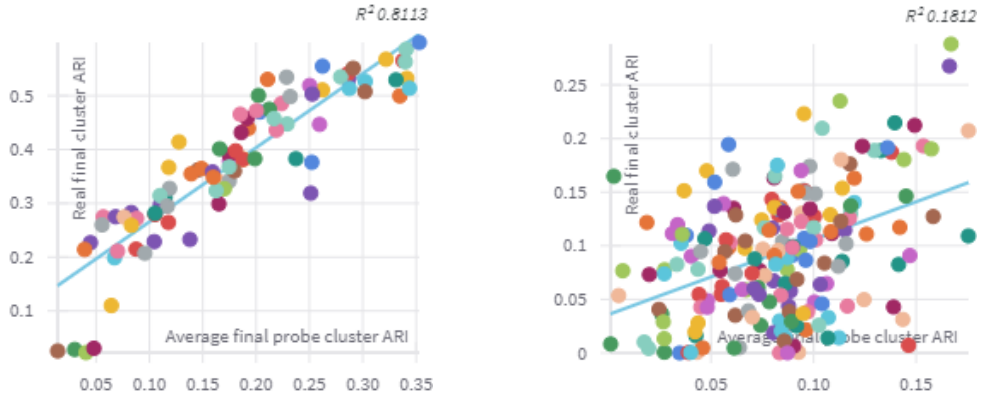
Table 4.4 also demonstrates the remarkable competitiveness of the NCD Spectral Clustering method, despite its low complexity. On average, it trails behind PBN by only 1.0% in ACC and manages to outperform PBN twice over 7 datasets.

Table 4.4 – Test ACC of the NCD methods averaged over 10 runs. Best results are in bold.

| Dataset | Baseline | NCD SC | TabularNCD | PBN |
|---|---|---|---|---|
| Human | 71.6±1.7 | **93.1±9.7** | 72.2±2.6 | 76.7±1.8 |
| Letter | **64.9±2.6** | 57.4±5.8 | 62.1±3.0 | 62.4±2.0 |
| Pendigits | 53.4±6.6 | 81.7±2.3 | 57.0±6.0 | **82.8±0.6** |
| Census | 59.1±0.8 | 48.0±1.8 | 45.2±4.8 | **62.4±0.9** |
| m feat | 66.7±4.1 | 89.2±2.3 | 90.2±2.7 | **91.7±0.8** |
| Optdigits | 40.7±5.1 | **95.4±5.3** | 73.0±8.4 | 92.6±2.3 |
| CNAE-9 | 40.2±3.2 | 69.0±6.7 | 51.3±5.2 | **72.6±4.6** |
| Average | 55.7 | 76.3 | 64.4 | **77.3** |

To investigate the reasons behind the subpar performance of TabularNCD, we look at the correlation between the ARI on the hidden classes and the final ARI of the model on the novel classes. A strong correlation would imply that if a combination of hyperparameters performed well on the hidden classes, it would also perform well on the novel classes. To examine this, we plot the average ARI on the hidden classes against the ARI on the novel classes. Figure 4.5 is an example of such a plot. It shows that, in the case of the *Letter Recognition* dataset, PBN has a much stronger correlation than TabularNCD. We attribute this difference to the large number of hyperparameters of TabularNCD (7, against 4 for PBN), which causes the method to overfit on the hidden classes, resulting in a lack of effective transfer of hyperparameters to the novel classes.

In conclusion, this section has shown that when the number $C^u$ of novel classes is known, NCD SC performs almost as well as PBN. Therefore, in this specific scenario,




(a) PBN (b) TabularNCD

Figure 4.5 – Comparison between the ARI on the hidden and novel classes. Each point is a different hyperparameter combination.

NCD SC is a viable candidate for addressing the NCD problem due to its lower complexity and shorter training time. Conversely, despite its strong learning capacity, TabularNCD is penalized by its high number of hyperparameters.

### Results when the number of novel classes is estimated

As expressed in Section 4.5, the number of clusters is estimated in the representation learned by PBN during the $k$-fold CV. The result is a method that never relies on any kind of knowledge from the novel classes. This approach is also applicable to the baseline and the spectral embedding of SC, but not to TabularNCD as it requires the number of clusters during the training of its representation. For a fair comparison, TabularNCD is trained here with a number of clusters estimated beforehand with a CVI.

To determine which CVI will perform the best in this application, we estimate the number of classes in the latent spaces learned by PBN when $C^u$ was known in advance. Figure 4.6 displays the average ranks of the CVIs in the latent spaces, and the details of the results can be found in Appendix B.5. The Nemenyi post-hoc test was used to compare the methods against all others. However, given the relatively small number of datasets, the Critical Difference (CD) is large and the CVIs are not statistically different from each another according to the Nemenyi test.

Nevertheless, we find that the Silhouette coefficient performed the best in the latent space of PBN, closely followed by the Calinski-Harabasz index. The elbow method is

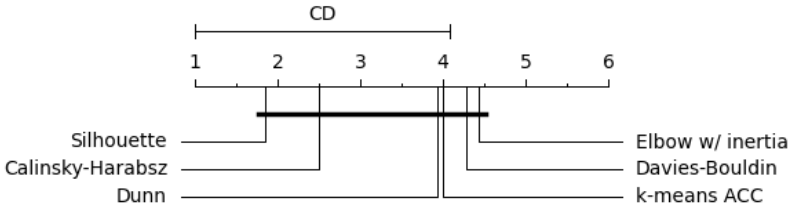



Figure 4.6 – Comparison of the CVIs in the latent space of PBN using the Nemenyi test with a 95% confidence interval.

ranked last, which could be explained by the difficulty in defining an elbow [2].

To summarize, in the following results, the Silhouette coefficient will be used for all estimations of $C^u$. The full training procedure described in Section 4.6 will be used to train the baseline and PBN, with $C^u$ estimated in their latent spaces as detailed in Algorithm 1. For TabularNCD, $C^u$ is estimated once in the original feature space (see Appendix B.5 Table B.5 for the values used). And NCD SC is trained as it was described in Section 4.3.3, but with $C^u$ estimated in its spectral embedding.

Table 4.5 – Test ACC averaged over 10 runs. With $C^u$ estimated with the Silhouette coefficient. Best results are in bold.

| Dataset | Baseline | NCD SC | TabularNCD | PBN |
|---|---|---|---|---|
| Human | 70.8±2.9 | 30.2±4.2 | **71.1±0.0** | **71.1±0.0** |
| Letter | **64.0±6.1** | 34.8±2.3 | 41.8±4.9 | 61.3±4.7 |
| Pendigits | 46.7±3.6 | 74.1±2.1 | 57.0±6.0 | **83.0±0.3** |
| Census | **56.6±3.6** | 29.0±3.5 | 35.7±1.6 | 49.8±0.1 |
| m feat | 59.5±7.7 | 73.2±3.9 | 41.1±0.2 | **90.6±2.2** |
| Optdigits | 42.1±4.6 | 79.5±4.0 | **96.9±1.3** | 90.5±4.8 |
| CNAE-9 | 33.8±3.8 | 44.6±3.9 | 39.3±0.2 | **50.8±1.5** |
| Average | 53.4 | 52.2 | 54.7 | **71.0** |

As emphasised earlier, Table 4.5 reports the results of the different NCD algorithms in the most realistic scenario possible, where both the labels and the number of novel classes are not known in advance. This is the setting where PBN exhibits the greatest improvement in performance compared to the other competitors, achieving an ACC that is 17.6%, 18.8% and 16.3% higher than the baseline, NCD SC, and TabularNCD, respectively.

Remarkably, TabularNCD outperforms PBN on the Optdigits datasets where the number of clusters was overestimated by the Silhouette coefficient in the original feature space.

2. There are no widely accepted approaches, as the concept of an "elbow" is subjective. In this study, we employed the *kneedle* algorithm [142] through the *kneed* Python Library [143].

This suggests that TabularNCD probably only utilized the output neurons necessary for clustering, leaving the others unused, which was the method proposed in [29] for estimating $C^u$. This is not true for the Letter dataset where $C^u$ was significantly overestimated, indicating that accurate estimations will likely result in improved performance.

Compared to the case where $C^u$ is known in advance, the ACC of the baseline falls from 55.7% to 53.4% and NCD SC falls from 74.3% to 52.2%. This shows that they are both unable to find a latent space suitable for the estimation of $C^u$.

However, the ACC of PBN remains an impressive 71.0%, demonstrating that this simple method, comprising of only two loss terms, is the most appropriate for tackling the NCD problem in a realistic scenario. In contrast to the baseline and NCD SC, its reconstruction term enables it find a latent space where the unlabeled data is correctly represented. And unlike TabularNCD, it has a low number of hyperparameters which decreases the probability of overfitting on the hidden classes during the $k$-fold CV procedure.

**Impact of the ratio of novel classes**

In the literature on Novel Class Discovery, the ratio of the number of novel classes to the number of known classes is usually set arbitrarily (e.g. 0.5 if there are few classes, and 0.2 if there are many). In reality, this ratio is not known in advance, so NCD methods should ideally be robust over a wide range of ratios. To compare the robustness of the 5 methods discussed in this paper, we perform experiments on the *Letter Recognition* [125] dataset when the number of novel classes increases. For each number of novel classes that was evaluated, we defined 5 random combinations of known/novel classes. For example, in Figure 4.7, the second point from the left corresponds to the average performance of the methods over 5 random combinations of 3 novel and 23 known classes.

From the results displayed in Figure 4.7, we find that the performance of the methods is almost always uniformly decreasing as the ratio of novel classes increases. We also observe that the accuracy of the deep-based methods (PBN, TabularNCD and the baseline) starts to decrease faster than NCD $k$-means and NCD SC when the number of novel classes becomes too large. As the number of known classes $C^l$ decreases, the quality of their latent representation is reduced and their clustering is negatively affected. Of course, this is especially true for the baseline, which trains its latent space solely on the known classes.

The performance of TabularNCD peaks around 7 novel classes, the number for which it was specifically optimized. On the other hand, PBN is more stable and outperforms its competitors over a wide range of values. Its small number of hyperparameters, coupled

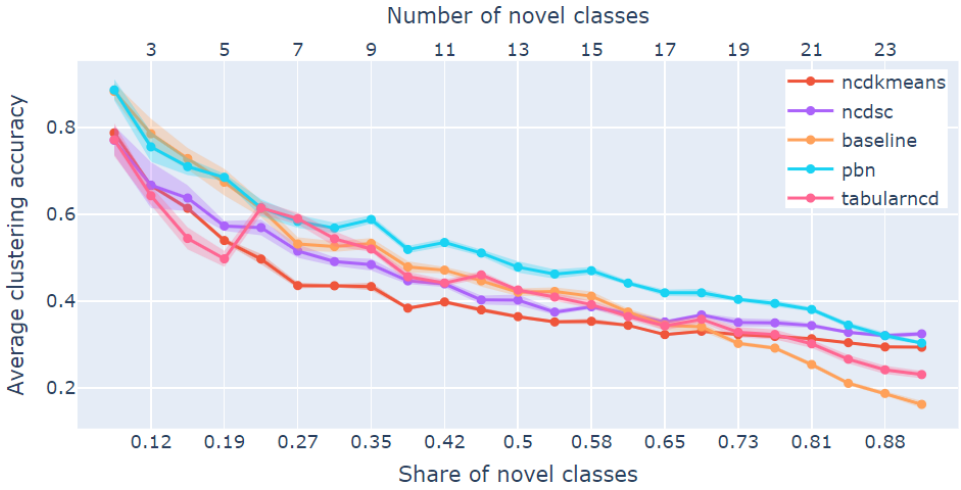



Figure 4.7 – Performance of the methods when the number of novel classes increases.

with its reconstruction loss, makes it less prone to overfitting, and it still extracts relevant information in its latent space even when there are few known classes.

## 4.8 Conclusion

In this chapter, we have shown that in the NCD setting, unsupervised clustering algorithms can benefit from knowledge of the known classes and reliably improve their performance by implementing simple modifications. We have also introduced a novel NCD algorithm called PBN, which is characterized by its simplicity and low number of hyperparameters, which proved to be a decisive advantage under realistic conditions. In addition, we have proposed an adaptation of the $k$-fold cross-validation process to tune the hyperparameters of NCD methods without depending on the labels of the novel classes. Finally, we have demonstrated that the number of novel classes can be accurately estimated within the latent space of PBN. These two previous contributions have shown that the NCD problem can be solved in realistic situations where no prior knowledge of the novel classes is available during training.

With the multiple methods proposed in this chapter, along with TabularNCD in Chapter 3, there should now be enough options to get reasonable results for NCD problems in most tabular datasets. In the next chapter, we will present a web interface where these models can be used to partition a datasets in just a few clicks. It is primarily intended to allow domain experts who do not necessarily have coding or data science experience to take advantage of our algorithms.

Chapter 5

# An Interactive Interface for Novel Class Discovery in Tabular Data

## Contents



This work was presented in a demo paper at an international conference: [24] An Interactive Interface for Novel Class Discovery in Tabular Data. Colin Troisemaine, Joachim Flocon-Cholet, Stéphane Gosselin, Alexandre Reiffers-Masson, Sandrine Vaton and Vincent Lemaire. In: *Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Applied Data Science and Demo Track*, 2023.

## 5.1 Introduction

Novel Class Discovery (NCD) [29, 20] is a new and growing field, where we are given during training a labeled set of known classes and an unlabeled set of different classes that must be discovered. In recent years, many methods have been proposed in the context of computer vision [19, 17, 37].

Tabular data refers to data arranged in a table, where each row is an observation and each column is an attribute. It is one of the most common types of data in practical applications such as medical diagnosis, customer churn prediction, cybersecurity, and credit risk assessment. [144]. An intuitive example of application of NCD in tabular data would be customer churn prediction: by using a dataset that includes the reasons why customers stopped using a product, we can more accurately identify other causes of churn in an unlabeled set where the reasons have not yet been identified.

While in practice, tabular data is one of the most prevalent data types in the real world, to the best of our knowledge, only two papers have attempted to solve NCD specifically for tabular data [21, 23]. This is partly due to the heterogeneous nature of tabular data and its lack of spatial and semantic structure, which makes it difficult to apply some computer vision techniques such as data augmentation or Self-Supervised Learning [18]. Furthermore, tabular data contains attributes that are specific to each domain. This means that analyzing and understanding the results of NCD or clustering algorithms can be challenging for a data scientist who is not necessarily familiar with the attributes of the dataset. On the other hand, the domain expert does not necessarily have the knowledge required to write code and run NCD or clustering algorithms.

In an ideal scenario, the domain expert would be included in the training loop to interpret the results produced by the data scientist. But for practical reasons, it can be difficult to dedicate two people to this task, as having a data scientist run an algorithm, present the results to the expert, and update the parameters based on the expert's feedback can be a slow and tedious process.

Hence, the goal of the interface proposed here is to allow a domain expert to visualize his data and run NCD or clustering algorithms without having to write code, as in visual data mining [145]. Given a preprocessed dataset, a user can employ this interface to (i) get a first idea of the separability of the data with t-SNE, (ii) select which features and classes to use, and which classes are considered novel (iii) parameterize and execute NCD and clustering algorithms and (iv) train decision trees to generate rules and interpret the

classes or clusters. Based on theses results, an expert can remove features or classes that have too much influence on the results, re-train a clustering model and re-generate rules. This process can be very tedious through code, but it can be done in only a few clicks with this interface (which even a data scientist could benefit from).

Currently, TabularNCD [21] is the only NCD algorithm implemented in this interface. The unsupervised methods of spectral clustering and $k$-means are also available, as well as a simple baseline method for solving NCD. This baseline trains a classification neural network on the labeled data, and then projects the unlabeled data in its last layer before clustering it with $k$-means.

As expressed before, this interface cannot replace the domain expert. It only allows him to explore his dataset using machine learning tools without writing code. This interface is also upgradeable, as new NCD or clustering algorithms can be quickly implemented. The application is open source and can be installed locally using the code at `https://github.com/ColinTr/InteractiveClustering`. The video of the demonstration is available at `www.youtube.com/watch?v=W7ru8NHPj-8`.

## 5.2 Interface description

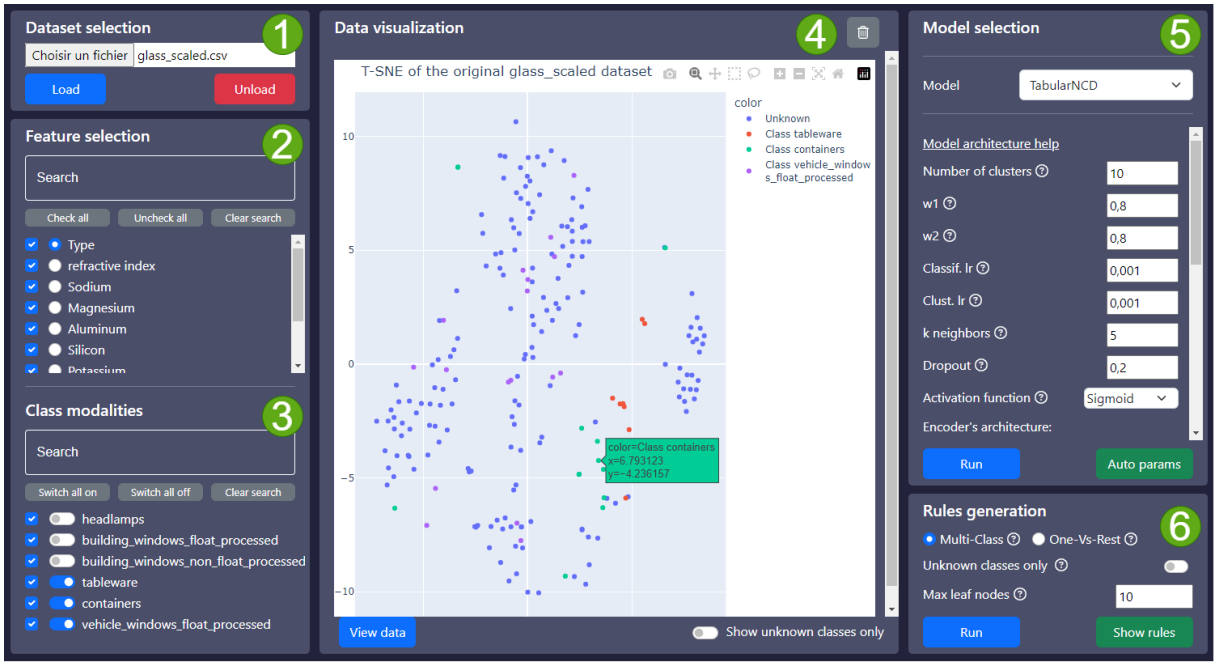



Figure 5.1 – The interface for interactive clustering and Novel Class Discovery.

As shown in Figure 5.1, the interface is composed of 6 different panels that we will describe in this section. For reference, the interface was made in JavaScript with React 18.2.0, and the Python code is executed by a Flask 2.2.2 backend server.

After selecting and loading a dataset with panel (1), the user can select in panel (2) which features to use in the dataset, and indicate which is the class feature. Panel (3) lists the modalities of the class feature picked earlier. Here, the user can choose to remove some classes from the dataset by unchecking them and indicate which classes are considered as known or novel. Given a real dataset including both labeled and unlabeled data, a group of observations could be labeled as "novel", which can be selected in this panel.

With panel (4), the data can be visualized in 2 dimensions by running a t-SNE. The user also has the option to view only the novel classes for easier readability. Clicking on a point displays all its attributes. Note that in an effort of optimization and better responsiveness, if a data plot is requested and has the same coordinates as a previous request, the t-SNE is re-used and only the coloring of the points is updated.

The NCD and clustering models can be selected and configured in panel (5). Currently, 4 models are available: TabularNCD [21] is a NCD method that pre-trains a simple encoder of dense layers with the VIME [55] self-supervised learning method. It adopts an architecture with two "heads": one to classify the known classes and introduce relevant high-level features in the latent space of the encoder, and another classifier for the unlabeled data trained with pseudo-labels defined without supervision in the latent space. Next is $k$-means, which was implemented for its simplicity and wide adoption in the community. It has the advantage of having a single parameter (the number of clusters). Spectral clustering is also available. It is known for its good results and its ability to discover new patterns across a wide variety of datasets [135]. And finally the baseline method described in Section 5.1 can be selected. Both TabularNCD and the baseline rely on an architecture composed of a combination of dense layers, dropout and activation functions which can all be modified through the interface (even the sizes and number of hidden layers).

Starting the training of TabularNCD or the baseline will produce a pop-up that displays the current progress of the training and the estimated time to completion. It is also possible to visualize a t-SNE of the latent space of these models, instead of visualizing the original features of the data.

Finally, in panel (6), the user can get an interpretable description of the results by training a decision tree to classify the known classes and the discovered clusters. Figure 5.2

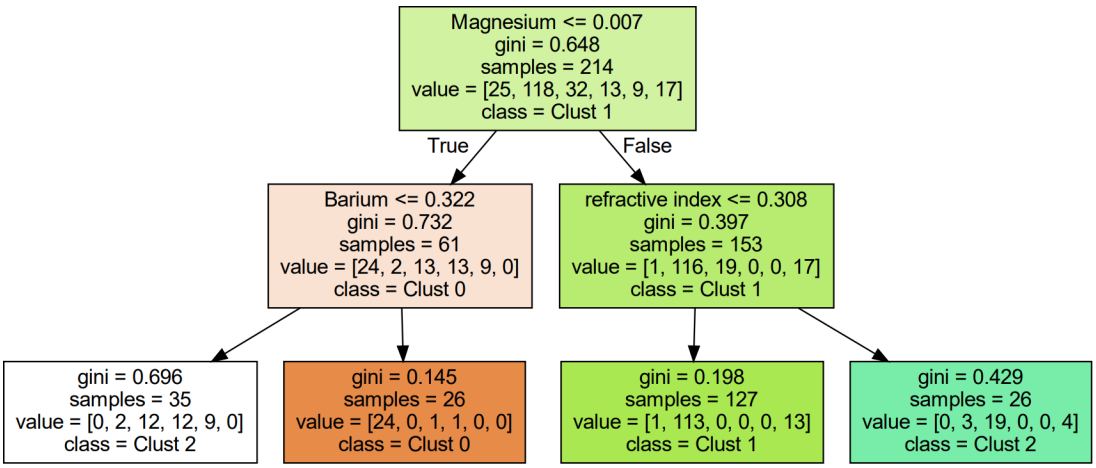



Figure 5.2 – Example of rules that describe the classes of the *glass identification* dataset.

is an example of rules in a decision tree obtained for the *glass* dataset. Each box represents a node/leaf of the tree and displays the rule and the majority class. The tree can be multi-class and will give an overview of the relations between all the classes and clusters, but it can be hard to comprehend because of its complexity. For this reason, we can instead use a *one-versus-rest* approach, where for each class or cluster, a decision tree has to predict the class or cluster against all the others. As each individual tree solves a problem of lower complexity, they are shorter compared to the multi-class case and are more easily interpretable.

## 5.3 Conclusion

This chapter introduces an interactive data exploration interface for the problem of Novel Class Discovery in tabular data. This interface is mainly targeted to domain experts and data scientists. The user can quickly visualize the data and generate clusters of novel classes along with interpretable decision trees to describe them. Furthermore, the user can easily identify both features and classes to remove from the training process and start a new clustering with different parameters.

In the future, this interface could be improved by adding a function to estimate the number of clusters (i.e. the number of novel classes). New NCD and clustering methods such as the ones studied in Chapter 4 could also be implemented. Giving the user the ability to merge or split some clusters and update the decision tree's rules accordingly could also be an interesting addition.

At this point, we have gathered enough experience, tools and methods in Chapters 3, 4 and 5 to expect to get interesting results for our initial diagnosis discovery problem. Therefore, we dedicate the next chapter to the creation of a dataset describing faults in operational customer FTTH data, and use it to benchmark our algorithms.

Chapter 6

# Exploration of operational FTTH data

## Contents

## 6.1 Introduction

This thesis is carried out at the *Orange* company, which is an Internet Service Provider (ISP) and is therefore tasked with delivering reliable internet connectivity to homes and businesses. Over the past decade, *Orange*'s telecommunication infrastructure has undergone significant changes. The introduction of Asymmetric Digital Subscriber line (ADSL) in 1999 has given access to high-speed internet access by utilizing existing phone copper cables in France. However, it has become obsolete with the advancements in optical fiber technology. Optical fiber not only provides significantly enhanced bandwidth capabilities over longer distances [146], but is also more energy efficient [147, 148]. Consequently, *Orange* has started to progressively shut down the ADSL network since 2022, and new ADSL subscriptions are no longer available where optical fiber is accessible. In anticipation of the complete stop of ADSL services planned for 2030, *Orange* is currently rapidly expanding its Fiber-To-The-Home (FTTH) network.

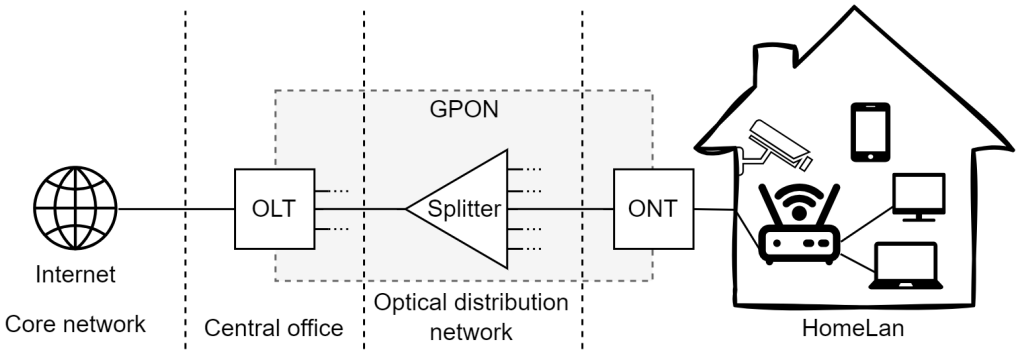
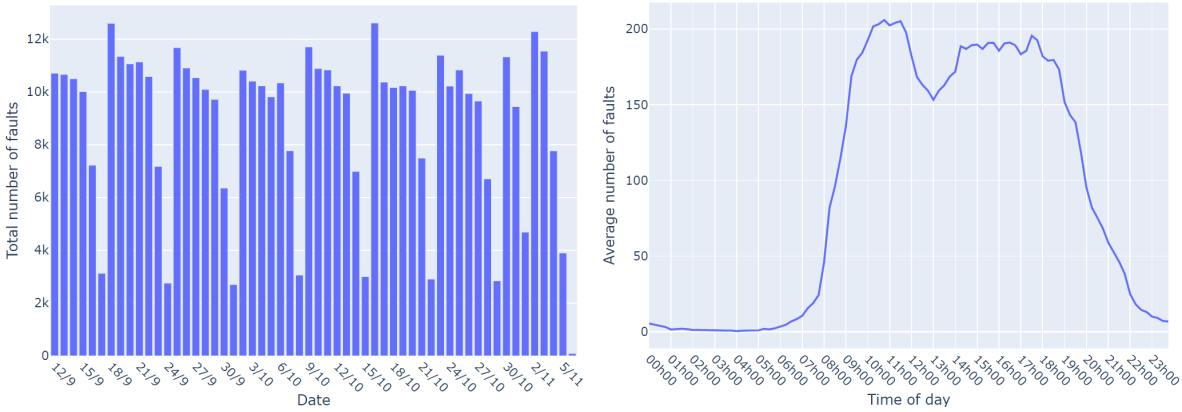



Figure 6.1 – Simplified architecture of GPON FTTH networks.

To deploy FTTH networks, ISPs have widely adopted the Gigabit-capable Passive Optical Network (GPON) solution [149]. Speeds are typically 2.4 Gbits/s down and 1.2 Gbits/s up, and a single fiber can generally connect up to 64 customers. Incidentally, *Orange* has also started to deploy XGS-PON [150], where a single fiber can provide speeds of 10 Gbits/s up and down to 128 customers. As shown in Figure 6.1, the network can be divided into 4 main segments [151]. The first is the **core network**, which is the backbone of an ISP's infrastructure and is responsible for routing and transmitting data across long distances. It consists of high-capacity routers, optical fiber cables, and other networking equipment that enable the efficient transport of data packets between different geographical locations. Following the core network is the **central office**. It houses the Optical Line Termination (OLT) where the primary optical access transmitter and receiver

can be found. From this fiber starts the passive infrastructure known as the **optical distribution network** (ODN). It is mainly composed of splitters that broadcast the single incoming signal into many signals for up to 64 consumers. This signal is finally received by the Optical Network Termination (ONT) at the customer premises. The ONT converts the optical signals carried by the fiber into digital electrical signals that can then be processed by a residential gateway. As such, it marks the end of the optical distribution network and the beginning of the customer's **Home LAN** (Local Area Network). In newer models, the ONT is often integrated directly into the residential gateway offered by ISPs, which provides essential services such as WiFi, firewall or IP routing.




(a) Total number of faults per day during the months of 2023.

(b) Average number of faults over the course of a day (every 15 minutes).

Figure 6.2 – FTTH faults detected between September and November 2023 [1].

The number of homes eligible for FTTH offers in France has grown from 10.9 million to 38.9 million between 2018 and 2023, as noted in Chapter 1. Even with reliable hardware and software, faults will inevitably occur in such a large network. They can be caused by a variety of factors such as weather damage, configuration errors, overloading, and so on. If an ISP takes too long to resolve the faults in its network, customers will become frustrated and businesses may find their operations disrupted. This dissatisfaction can cause customers to seek alternative service providers, resulting in customer churn and revenue loss for the ISP. For these reasons, the ability to quickly identify and correct faults is critical. In this context, *Orange* has developed an automatic diagnosis system to assist its

1. The storm "Ciarán" hit France between the 1st and 2nd November, where there is a drop in detected faults in Figure 6.2(a). It may be because electricity was unavailable for many homes, so no diagnosis could be made.

technicians and network experts determine the root cause of faults. This system is called DELC (for *Diagnostic Expert de la Ligne Client*, or *expert diagnosis of the customer's line*) and is a rule-based software developed by experts of the network. When a customer experiences problems with their internet services and requests a diagnosis, a description of the state of the network is first gathered through key performance indicators such as power voltages, optical reception powers, device temperatures, or software versions. Then, a set of rules comparable to decision trees is applied to this data to produce a final diagnosis. These rules are written by network experts, and are a translation of their knowledge (more details will be given in Section 6.2).

In 2023, approximately 200,000 FTTH tests were performed per day in France by this system, with an average of 7000 unique faults detected daily (see Figure 6.2). The DELC system is currently capable of diagnosing around 85% of the FTTH faults at *Orange*. Although this is an impressive rate, the remaining 15% of faults that the DELC system cannot diagnose must often be investigated by technicians. This is a significant problem, given that millions of interventions are carried out each year. Since its introduction in 2012, the DELC system's diagnosis rate has improved from 48% to the current 85% in 2023. However, this rate has recently stopped progressing. Network experts are struggling to find new diagnoses and speculate that the most obvious ones have already been discovered, leaving only complex ones to be found. The large number of faults and their many descriptive variables make manual analysis of the data difficult. In this thesis, we have developed machine learning tools in Chapters 3, 4 and 5 that network engineers may not be familiar with. By applying previously unexplored approaches to DELC's data, we hope to discover new and interesting diagnoses.

The objective is not to replace the current rule-based system, as it has the advantage of being easy to interpret. In this chapter, we will only suggest new diagnoses to the network experts from groups of similar undiagnosed faults. By diagnosing faults more accurately, the number and duration of interventions are reduced. This means technicians make fewer trips to customer premises, reducing fuel consumption and ultimately lowering the operational costs and carbon footprint of *Orange*'s FTTH network.

We begin in Section 6.2 with an overview of the DELC system and practical details on the data it handles. Section 6.3 describes the extensive data collection and preprocessing that was conducted to transform DELC's large volumes of data into exploitable tabular data for our algorithms. Finally, in Section 6.4, the multiple clustering and NCD methods proposed in Chapters 3 and 4 are benchmarked for their ability to rediscover known

diagnoses that were hidden during the training process. The best performing algorithms will be selected and applied to the real undiagnosed data before presenting the results to the network experts.

## 6.2 Orange's diagnosis system

Every time a customer experiences an interruption of their internet services, *Orange*'s diagnosis software, DELC, is executed. It can be launched by the support hotline, by the customer through the dedicated app, or even by technicians during an intervention to verify the completion of their work. In order to make a diagnosis, around 3000 variables describing the status of all the equipment related to the client are collected from the core network all the way to the home LAN (see Figure 6.1). To illustrate the type of data used during diagnosis, we list a few examples of equipment along with their descriptive variables:

— OLT/ONT: model, vendor, received and transmitted power level in dBm, list of recent alarms, fiber length.

— Residential gateway: firmware version, first and last connection dates, IP configuration, activated WiFi channels.

— DSLAM: current working state, manufacturer, port number, power voltage.

Some information about the customer's local equipment (e.g. PLC adapters [2], TVs, phones or computers) is also obtained. This can help evaluate performance indicators such as WiFi quality, or if the same IP address has been assigned to two different devices. External information like weather data is also taken into account, as thunder strikes are one of the most common causes of disruption. Finally, the neighbors of the customer that is being tested can also be inspected. If several customers were affected simultaneously, the fault is considered *collective*, and the root cause is probably not in the customer's home, but further upstream in the network.

```
IF there is a set-top in the home network
    AND IF its average 5GHz WiFi RSSI over 7 days <= -87dB
        THEN
            The set-top box is too far from the home router
```

Figure 6.3 – Example of a diagnosis rule in DELC.

2. PLC (Power-Line Communication) adapters are devices that can carry data over electric lines.

In order to minimize the time spent by field technicians and network experts on troubleshooting, one of the main requirements of this software is to generate pertinent diagnoses that clearly identify the root cause of the fault. In addition, it must reach a conclusion in a reasonable amount of time ($< 60$ sec), since it can be run while on-call with a customer. To do so, all devices in the customer's network are queried simultaneously, and all previously defined rules, such as the one shown in Figure 6.3, are applied. Since multiple rules can be valid for the same fault, they are assigned weights by network experts according to their criticality and confidence, and the final diagnosis is chosen by the largest weight.

Table 6.1 – Faults validating rules.

| | Rule 1<br>*weight = 20* | Rule 2<br>*weight = 50* | Rule 3<br>*weight = 100* |
|---|---|---|---|
| Fault $A$ | ✗ | ✓ | ✗ |
| Fault $B$ | ✓ | ✗ | ✓ |
| Fault $C$ | ✗ | ✗ | ✗ |

This approach used by DELC to handle multiple rules is illustrated through Table 6.1, where 3 faults are confronted with 3 diagnosis rules with different weights. For fault $A$, only rule 2 was valid, so there is no confusion about the final diagnosis. In the case of fault $B$, both rules 1 and 3 are valid, so the diagnosis of rule 3 is considered final, since its weight is higher. Finally, fault $C$ does not validate any of the rules. As noted in Section 6.1, this happens on average for 15% of the faults. These undiagnosed faults are referred to as **DNIs** (for *Défauts Non Identifiés*, or *unidentified faults*) and will be the main topic of our experiments. In the rest of this chapter, we consider the faults of the known diagnoses as our labeled set of known classes, and the faults marked as DNIs as our unlabeled set.

Reports from field technicians about discrepancies between a diagnosis and the actual fault observed during an intervention are usually the starting point for network experts. By manually reviewing such faults and investigating the causes of the anomalies, experts can sometimes create rules for new diagnoses or update existing ones. The new or updated rules are validated against previously recorded faults and, after some time in production, are fully integrated into DELC. Currently, there are 203 known diagnoses for the scopes of FTTH and HomeLan faults, and the distribution of their occurrences is displayed in Figure 6.4. The unidentified faults are colored in red, and are split in a few sub-categories by the DELC system according to the general state of the internet services of the customer (e.g. is the session in an active, inactive or unknown state).

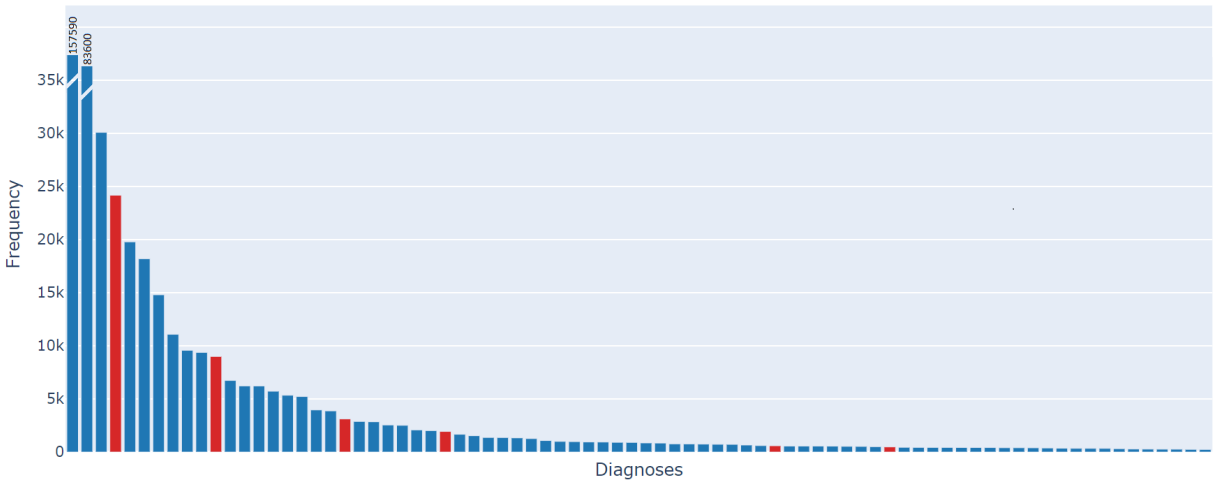



Figure 6.4 – Distribution of the 80 most represented classes of DELC between the 12/9/2023 and the 6/11/2023 (55 days). Known classes are in blue and the families of DNIs are in red. The most two represented classes were capped for the sake of clarity, but had 157590 and 83600 instances respectively.

Compared to tabular datasets in fields that deal with similar data, such as sensor data analysis or cybersecurity, the DELC dataset is a true outlier due to its exceptional number of features (2914) and classes (203). In addition, the highly subjective definition of our objective (discovering "pertinent" diagnoses) makes it uniquely challenging. In the following section, we describe the difficult data collection and preprocessing of the DELC dataset. In the absence of a target, it is difficult to predict which features might be useful to create new diagnoses. Despite the heterogeneous nature of the numerous features, the preprocessing is critical and must be performed carefully to obtain any relevant results.

## 6.3 Data collection and preprocessing

The raw features found in the dataset exploited by DELC are very diverse in nature. For example, there are several formats of timestamps, some XML and JSON, categorical variables with many modalities, unique identifiers, IP addresses, ... Since the DELC system is a custom software developed by network experts, each of its rules can utilize information in any format, so it's not a problem. However, machine learning algorithms such as neural networks can usually only exploit clean numerical data where no values are missing. Therefore, some preprocessing is required to prepare the raw data from DELC's databases for the methods we have developed. In this section, the data wrangling process that was

followed is presented. Data wrangling is a general term that designates the process of transforming the raw data into a format that is suited for a specific application. It is a time consuming but essential process that is generally divided into 6 main steps which are detailed below.

**1. Data acquisition** marks the first phase of data wrangling, where all the relevant information is sourced from databases, APIs, or files. This step requires extensive exploration of the multiple sources of data exploited by DELC. Through discussions with the leading developers of DELC, we identified 4 databases that contained data relevant for the problem of fault diagnosis. Concretely, this step mainly involved writing SQL queries to retrieve the data from the databases. Since the faults of the entirety of *Orange*'s network are centralized in shared databases (e.g. ADSL, xDSL, GPON and XGS-PON), we limited our requests to FTTH customers between the months of September and November 2023. In addition, the diagnoses were restricted to a specific version of the DELC software to avoid inconsistencies in the target, as DELC's rules are regularly updated.

During this step, the objective is to gather as much relevant data as possible. All the descriptive attributes in the databases are collected, and the superfluous data will be filtered in the next steps. Nevertheless, with respect to the General Data Protection Regulation (GDPR), we actively avoid including any personal data of the customers, which is anyway unrelated to the network diagnosis task.

**2. Structuring** (or *transforming*) consists in reshaping the raw data into a format usable by the analysis tools of the next steps. The multiple tables that were just collected are pivoted so that each row corresponds to a fault and each column to the descriptive attributes. Information contained within concatenated features (such as devices configurations in JSON format) is also extracted into individual features. The result is a CSV table that can be loaded into the data analysis library of choice, which in our case is the Python package *Pandas*.

**3. Cleaning** is mainly dedicated to rectifying errors, inconsistencies and missing values. This is an important step to guarantee the accuracy of the dataset and should not be overlooked. Here, we start with a manual inspection of each of the 2914 attributes to detect unusable content, which mainly included IP addresses, logs, URLs and attributes that were unrelated to our context. Constants and unique identifiers are also removed, and duplicates are detected by computing the correlation matrix. At this point, there are 501 raw features remaining, which are all rich in information. Attributes are then cast into their intended data types (floats, integers, booleans, strings, and so on), and the

formats of the timestamps are standardized. Finally, the missing values are either filled with zeroes (e.g. number or duration of alarms) or with the average value (e.g. device temperature or voltage).

**4. Enriching** a dataset with information from new sources provides additional context and insight. In this case, since enough information has already been collected during the *data acquisition*, this step only consists of extending the data of each fault with the information from the 4 tables, along with the final diagnosis of DELC (including the DNIs).

*Feature engineering* can also be considered part of the enriching process. It refers to the process of extracting the information in the raw features into relevant features that can be directly exploited by models. While various automated feature engineering software packages are available [152, 153], they cannot handle the specificity of DELC's data and may create redundant or unimportant features. Therefore, most of the preprocessing time was spent in this phase, since almost each of the 501 raw features is unique and requires a different treatment. If the purpose of a feature was not clear, the source code of the DELC software was consulted, and the way the features were used in the rules was translated in the preprocessing. For example, the DELC software often checks whether features are present or not, but the actual value is not important. The result was binary variables with a 1 if the feature was present and 0 otherwise. Finally, features containing categorical data were one-hot encoded and numerical data was standardized to have a mean of 0 and a standard deviation of 1.

After this process, the 501 raw attributes became 781 numerical features that could already be used to train models.

**5. Validating** the data involves assessing its quality and identifying any remaining errors, anomalies or inconsistencies that may have been overlooked in earlier stages. In this dataset, redundant and ambiguous information is stored in different formats such as XML, JSON or timestamps. For this reason, perfectly correlated pairs of features and constant features were inadvertently introduced during the enrichment process and were removed in this step. Studying the confusion matrix of a random forest classifier trained to predict the known diagnoses was also very helpful. It revealed that some classes could not be reliably predicted, so we were able to identify which important features were missing and add them to the dataset. This leaves the dataset with 714 features.

At this point, all the features have been carefully processed and the dataset could already be used to explore the DNIs. However, we make the observation that 714 features is

a lot and the NCD and clustering algorithms that will be applied in the following section might be subject to the *curse of dimensionality*. The term "curse of dimensionality" was coined by Bellman in 1957 [154] and mainly refers to two undesirable phenomena happening in high dimensional data. First is the issue of sparsity, where an exponential amount of data is needed to fill high dimensional spaces. Without enough data, machine learning algorithms tend to overfit or fail to find the relevant patterns. Secondly, and perhaps most importantly in our case, the Euclidean distance measure used by clustering algorithms becomes gradually meaningless as the number of features increases [155]. Intuitively, with many dimensions, the probability that two points differ greatly in at least one dimension is large. Since the Euclidean distance is a squared sum of differences, it accentuates this problem and all the points in a set appear far away from each other.

Another reason for reducing the dimensionality of our current dataset is that there are a number of unnecessary features that could negatively affect the results of our NCD and clustering algorithms. For instance, many of the features are very sparse, as 33% of the features have a mode (i.e. the value that appears most often) that represents at least 99% of the values of the feature. Overall, reducing the dimensionality has many benefits, it reduces the risk of overfitting, mitigates the effects of the curse of dimensionality, prunes irrelevant features and shortens training time [156].

There is a rich literature of feature selection methods in supervised settings, as the objective is usually clear and the results are easily evaluated [157]. However, fewer approaches have been proposed for unsupervised scenarios as they have to rely on arbitrary characteristics of the data to evaluate the importance of the features. The Novel Class Discovery problem tackled in this thesis has the specificity of presenting both a labeled and unlabeled set of data during training. Thus, a naive approach to reduce the dimensionality of our dataset could be to apply supervised feature selection on the labeled set. For example, the importance of a variable in a random forest can be measured by the decrease in prediction accuracy when its values are randomly permuted (see *Mean Decrease Accuracy* or *Gini Importance* in [158]). Another approach, applicable to any model, is to add a small amount of noise to each variable and measure the change in the output [159]. The input variable whose change had the largest effect on the output is considered the most important. However, importance measures are always determined relative to a target variable (so here, the known diagnoses). In our diagnosis discovery problem, these known classes are only here to support the exploration of an unlabeled set. In this unlabeled set, no target variable is available to reliably evaluate the output of the models,

and variable importance measures cannot be applied. Although NCD problems assume that known and novel classes share some high-level discriminative features, they rarely overlap completely. In other words, a feature selection based on the variable importance computed only on the labeled set will be heavily biased towards the known classes, and the resulting features might not be sufficient to partition the novel classes. This aspect was also taken into account in the Projection-Based NCD (PBN) method proposed in Chapter 4. During the training of the latent representation, the classification loss on the known classes is accompanied by a reconstruction loss on all the data to avoid overfitting.

For these reasons, we turn to *unsupervised feature selection* (UFS). In contrast to supervised feature selection, UFS methods are unbiased and perform well when prior knowledge is not available. They are divided in three main categories by [156]: *filter* methods measure intrinsic properties of the data [160], *wrapper* methods measure the impact of the features on the result of a clustering algorithm [161] and *hybrid* methods try to find a compromise between filter and wrapper approaches [162]. Filter based feature selection methods are the most popular for their speed and scalability. They evaluate features without involving a clustering algorithm, so they are not biased towards specific learning models. On the other hand, wrapper and hybrid methods have a high computational cost, which discourages their use on our large dataset of almost 500,000 points and 714 dimensions.

In [156], the authors reviewed the state-of-the-art UFS methods and concluded that there is no method that is universally superior in all cases. The quality of the features selected by different approaches depends mainly on the nature of the dataset, the evaluation metric and the clustering algorithm used. Thus, we decided to compare three filter UFS methods: the first is SVD-Entropy [160], sorts the features by their impact on the entropy of the singular values of the dataset. Then the Laplacian score [163] ranks the features according to their importance in preserving the overall structure of the data. And the Variance Threshold is a simple approach that sorts the features according to their variance, where features with a larger variance (i.e. many unique values) are considered to be more useful than features with many repeated values [3]. Finally, our baseline UFS method is a random ordering of the features.

Although it was stated earlier that we should avoid directly using the labels of the

3. Since the continuous features have been standardized, they have a variance of 1, which causes them to be ranked first by the Variance Threshold. This is one of the limitations of this simple UFS method, but it is not significant in this case since we only have 51 continuous features and they all appear to contain relevant information.

known classes to select the features, using these labels to determine which UFS method is best suited to our problem should not introduce a strong bias toward the known classes. In the following experiments, subsets of features will be selected by the three UFS methods. The performance of a $k$-means trained on the known class data with these features will then be assessed. The UFS method for which the results were closest to the ground truth labels will be used to make the final feature selection.

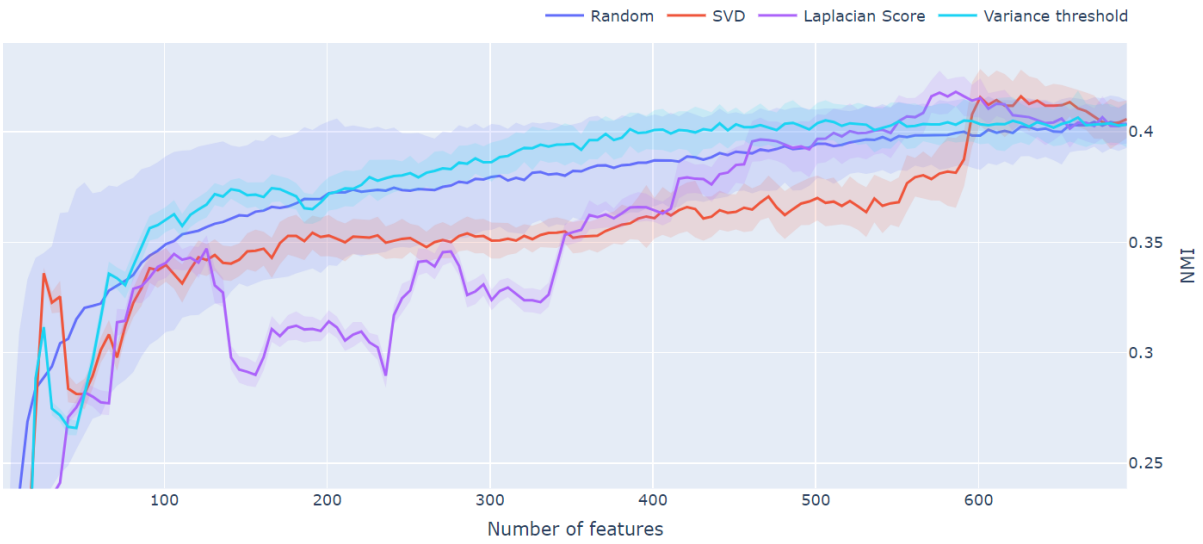



Figure 6.5 – Comparison of the performance of $k$-means on the 68 most represented known classes of DELC with different feature selection methods.

In Figure 6.5, the performance of three filter UFS methods is evaluated on the 68 most represented known classes of our dataset. Specifically, the average NMI of $k$-means trained with the $n$ most important features is plotted for each approach. So, at the 100 feature mark on the x-axis, the average performance of 50 runs of $k$-means trained on the first 100 features selected by each method is recorded. The ranking of the UFS methods is deterministic and therefore computed only once, while the performance of the random selection is averaged over 50 random rankings. While the random selection shows a surprisingly good average performance, its high variability is dissuasive. Both SVD-Entropy and Laplacian score are almost always outperformed by Variance Threshold, which shows stable performance over a wide range of numbers of features. The DELC dataset has many features that are almost constant (with the mode representing the majority of the values), so the Variance Threshold approach is well suited to filtering out these variables with little information [4].

4. Note that some of these low-information variables are likely to be relevant for rare fault types.

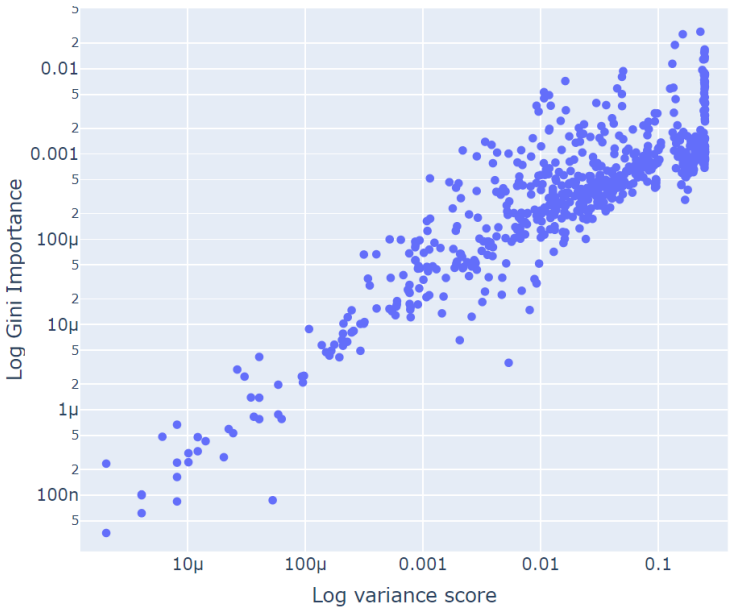



Figure 6.6 – In log scale, variance of each feature against its Gini Importance calculated from a Random Forest classifier trained to predict the known classes.

To conclude this feature selection step, the variance of each feature is calculated over the entire dataset, and we arbitrarily choose to keep the 500 features with the highest scores. Finally, to ensure that none of the features relevant to predicting the known classes have been removed by this unsupervised selection, a random forest classifier is trained to predict the known classes. And the corresponding Gini Importance [158] of the features is computed. We plot the Gini Importance and variance in Figure 6.6 and find a correlation of 0.804 using Spearman's rank correlation, showing that features with greater variance are more likely to be useful. Of the 500 features with the largest Gini Importance, 37 were not selected by the variance approach, so they are added to our dataset, bringing the total to 537.

**6. Publishing** is the last step of the data wrangling process. It aims at making the dataset available to anyone in the company who might have a use for it. The data must be accompanied by comprehensive documentation to ensure that the preprocessing can be reproduced and possibly improved. To this end, documentation is written that focuses on the required authorizations, data sources, and procedures to be followed. Finally, the availability of this dataset will be widely publicized through presentations illustrated with

However, we do not expect our NCD and clustering algorithms to be able to capture them, and we are primarily focused on discovering the most common diagnoses among the DNIs, not the rare event.

some possible use-cases.

Before proceeding with the experiments, let us describe the dataset that was created here. It contains 493,336 faults described by 537 variables, of which 51 are continuous and 486 are boolean. Among these variables, 51 are dedicated to alarms during diagnosis, 114 to the description of the FTTH network, 88 to the configuration of the modem and 97 to the client devices, while the rest are diverse (OLT, ONT, neighbors, WiFi, ...). As shown in Figure 6.4, the distribution of the 203 diagnoses is highly unbalanced, as the 10 most represented diagnoses make up 76.7% of all diagnoses. To improve the balance of the classes in the dataset, we decide to discard classes with less than 150 instances. This eliminates 105 classes that accounted for only 0.9% of the total number of faults. Similarly, some classes are overrepresented, so the maximum number of instances in a single class is limited to 2000. This pruning is not applied to the DNIs in order to preserve as much of their information as possible. In the end, the dataset that will be used in the experiments has 83,067 rows of 91 diagnoses (i.e. known classes) and 39,311 rows of 6 categories of DNIs (i.e. novel classes), which is a ratio of 68/32%.

Finally, Table 6.2 summarizes the evolution of the dataset through the 5 data wrangling steps (excluding the *publishing* step which does not update the dataset) in terms of number of classes, instances and features.

Table 6.2 – Numerical description of the DELC dataset across the data wrangling steps.

| Step | Classes | Instances | Features |
|---|---|---|---|
| 1. Acquisition | 203 | 327,228,003 | 4 |
| 2. Structuring | 203 | 493,336 | 2914 |
| 3. Cleaning | 203 | 493,336 | 501 |
| 4. Enriching | 203 | 493,336 | 781 |
| 5. Validating | 203 | 493,336 | 537 |
| Experiments | 91 | 83,067 | 537 |

## 6.4 The experiments

### 6.4.1 Objectives

As stated throughout this manuscript, the main objective of this thesis is to discover groups of similar faults among the unlabeled set of points (the DNIs). The groups will

be explained by rules, in the form of decision trees, and must be relevant enough to help technicians more quickly resolve the root cause of the faults for each group. However, the definition of "relevant" in this context is unclear, which makes this problem almost impossible to automatise. And as network experts have spent more time manually examining the DNIs, the most obvious groups have all been found, leaving only the more obscure ones to be discovered. For this reason, we believe that even in the best case, the algorithms we have developed so far will only be able to discover a few relevant groups of faults at a time.

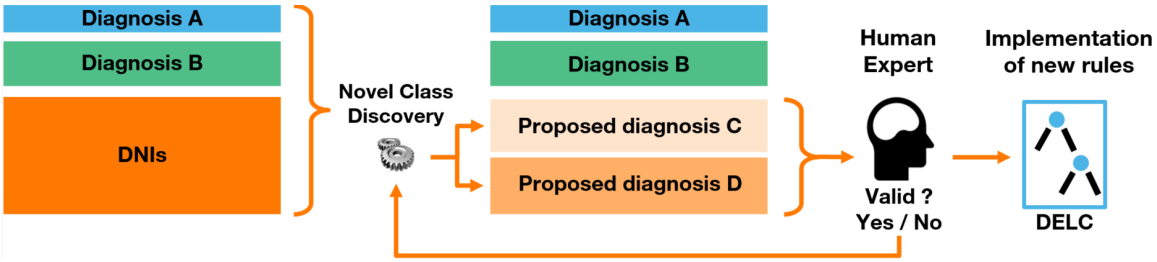



Figure 6.7 – Continuous Novel Class Discovery process with a human in the loop.

Since the opinion of a network expert is required to validate or invalidate the groups of faults we create (i.e. *clusters*), the process pictured in Figure 6.7 naturally comes to mind. First, a Novel Class Discovery algorithm is applied on the dataset to propose some new diagnoses. These diagnoses are then presented in the form of rules to a human expert who can select the valid ones and incorporate them into the software of DELC. This process can then be repeated after the new validated diagnoses have been added to the known classes, incrementally reducing the number of DNIs.

We believe that this is an approach likely to produce good results in a practical scenario. However, given the time constraints of this thesis, we did not have the opportunity to interact with a network expert and validate a large number of clusters to evaluate the results of our algorithms on the DNIs. Instead, we limit our work to the comparison of the different algorithms introduced in earlier chapters by evaluating their ability to rediscover known classes that were hidden during training (just like the result tables in Chapters 3 and 4). And the algorithm that performed best on the hidden known classes can later be used to generate clusters of DNIs.

In summary, the following sections will be used to benchmark the algorithms developed so far on this dataset and to create examples of diagnoses in the form of decision trees. The best performing NCD method could then be applied to the full dataset to actually

propose new diagnoses for the DNIs to the network experts.

### 6.4.2 Experimental setup and methodology

**Dataset.** The dataset used in the experiments below is summarized in Table 6.3 and consists of 83,067 rows and a total of 91 diagnoses. Around 30% of the known classes (i.e. 28 diagnoses) are hidden during training to serve as novel classes, while the remaining 70% (i.e. 63 diagnoses) are the known classes. In contrast to the methodology adopted in Chapters 3 and 4, no train-test split is used here. Instead, the models are evaluated on their ability to cluster the entire set of hidden classes. This is a common practice in unsupervised clustering papers, as clustering algorithms aim only to discover patterns in the data without relying on a labeled target. NCD algorithms are the same in this respect, so there is no notion of overfitting as in supervised learning. Furthermore, the goal of NCD is not to create a model destined to be deployed in production to predict new samples. The only objective is to discover the inherent classes within the unlabeled data, as we've shown in Figure 6.7 of Section 6.4.1. In this context, a train/test split is not justified and would deprive our exploratory analysis of valuable data points [5].

Table 6.3 – Details of the dataset.

| | Classes | Instances | Features |
|---|---|---|---|
| Labeled set | 63 | 57,015 | 537 |
| Unlabeled set | 28 | 26,052 | |
| Total | 91 | 83,067 | |

**Competitors.** The same methods used in the experiments of Chapter 4 will be compared here. They include:

— The unsupervised clustering algorithms $k$-means and Spectral Clustering, as well as their NCD-adapted versions (see Sections 4.3.2 and 4.3.3).

— The deep-based methods TabularNCD (see Chapter 3) and Projection-Based NCD (PBN, see Section 4.3.4).

— And the baseline method, where the data of the novel classes is projected through a multilayer perceptron that was trained to distinguish the known classes, and is then clustered with $k$-means (see the discussion of Section 4.3.4).

5. In chapters 3 and 4, we used a train/test split to have an experimental methodology comparable to most NCD papers, but as explained, it is not justified in this case.

The neural-network based methods (i.e. TabularNCD, PBN and the baseline) are trained with the same base architecture: an encoder of 2 hidden layers of decreasing size, and a final layer producing the latent space whose dimension is optimized. All methods are trained for 200 epochs with a fixed batch size of 512. Importantly, the hyperparameters are optimized following the $k$-fold cross-validation approach described in Section 4.4.

**Evaluation.** The clustering accuracy (ACC) on the unlabeled set will be the main metric of comparison during the experiments. It is defined in Equation 4.5 of Chapter 4.7.1, and uses the Hungarian algorithm (also known as the *Munkres assignment algorithm*) to optimally match the predicted clusters to the ground truth labels. Additionally, the Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) are also reported.

Note that in these experiments we chose to use the ground truth number of novel classes (i.e. 28). While it is unrealistic to assume that this number is available in a practical scenario, finding the exact number is not critical to the success of the clustering and NCD algorithms. In fact, as stated in Section 6.4.1, we expect that at best the algorithms will discover only a few relevant groups at a time. Therefore, if the number of novel classes is estimated within a reasonable margin of error, the few correct clusters generated should not be affected.

### 6.4.3 Fast spectral clustering with $k$-means

Spectral clustering is notoriously difficult to scale to large problems [164]. The reason is simple: given a dataset of $n$ points, it needs to construct an adjacency matrix of size $n \times n$, before searching for the eigenvectors of that matrix, an operation with a computational complexity of typically $\mathcal{O}(n^3)$. Since the distance function we use is the Gaussian kernel (see Section 4.3.3), the adjacency matrix is dense and its memory footprint of $n^2$ cannot be reduced with a sparse representation. In the previous chapters, the largest dataset was *US Census 1990* [127] with 18,000 instances. Given that a *float64* has a size of 8 bytes, the adjacency matrix required $18000 \times 18000 \times 8$ bytes, or 2.4 gigabytes of memory, and the spectral embedding could be computed relatively quickly. However, the dataset in these experiments has a total of $83,067$ instances, so its matrix requires 51.4 gigabytes of memory. Even using the *float32* format, which has half the precision and memory footprint, this is still 25.7 gigabytes, making eigenvector computation impractical.

To mitigate these scalability issues, various approximation techniques, parallelization strategies, and optimizations have been proposed. For example, a popular method proposed in [165] approximates the eigenvectors by sampling a subset of data points. The

eigenvectors are computed for this subset and the Nyström method is used to approximate the full solution. While this reduces computation, it is not efficient for datasets with a large number of features and is sensitive to the initial random subsampling.

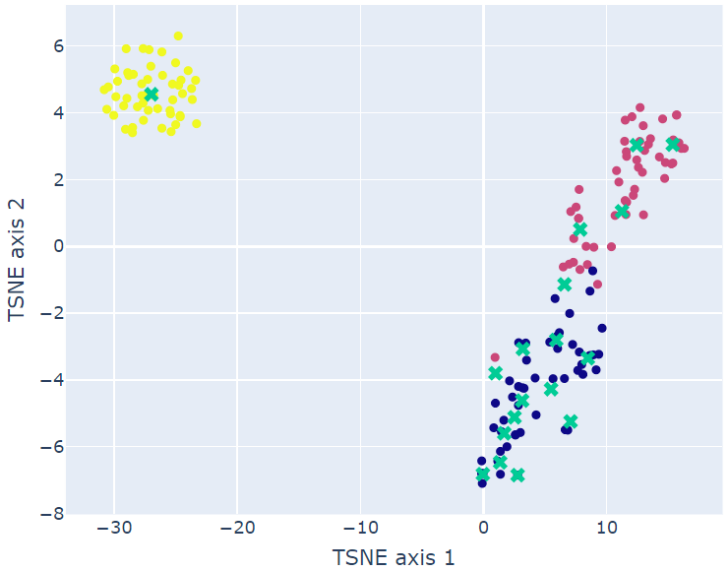



Figure 6.8 – t-SNE representation of the *iris* dataset approximated by 18 centroids (green crosses). The points are colored according to the prediction of the KASP algorithm.

Instead, we turn to the "$k$-means-based approximate spectral clustering" (KASP) method [164]. The main idea behind this method is to approximate spectral clustering by first running a $k$-means algorithm with a large number of centroids. These centroids are then used as a reduced representation of the dataset and are partitioned using spectral clustering. Finally, the labels assigned to the centroids by spectral clustering are propagated to the nearest points, effectively clustering the entire dataset. Because it uses $k$-means++ which defines multiple initial centroid positions and selects the best one in terms of inertia, the approximated representation is relatively stable even in high-dimensional problems where the data is sparse. This concept is illustrated in Figure 6.8, where the *iris* dataset is approximated by 18 centroids, and the points are colored by performing spectral clustering on these centroids and propagating the labels to the nearest points.

**Note:** In our experiments, the purely unsupervised spectral clustering is applied only to the unlabeled data of the novel classes (which has 26,052 points). The adjacency matrix weighs only 2.5 gigabytes when using a *float32* format, and the eigenvectors can be computed in a reasonable amount of time, so we do not use KASP in this case. On the other hand, NCD Spectral Clustering computes the spectral embedding of the entire dataset

dataset (both known and novel classes) and needs to be approximated for the reasons mentioned above.

### 6.4.4 Results

Table 6.4 – Clustering performance of the algorithms averaged over 50 runs. Best results are in bold.

| Method | ACC | NMI | ARI |
|---|---|---|---|
| $k$-means | 28.3±2.0 | 42.3±1.6 | 21.2±2.5 |
| NCD $k$-means | 31.0±1.6 | 44.0±1.0 | 24.3±2.4 |
| Spectral Clustering | 20.2±1.2 | 37.9±0.6 | 13.7±1.2 |
| KASP NCD Spectral Clustering | 29.4±1.3 | 39.1±1.4 | 22.3±1.9 |
| Baseline | **45.3±1.1** | 53.8±0.7 | 35.6±1.0 |
| TabularNCD | 40.7±2.2 | 50.5±1.0 | 33.4±2.4 |
| PBN | **45.5±1.1** | **56.2±0.7** | **40.0±0.9** |

**Numerical results.** Table 6.4 measures the average performance over 50 executions of all competitors in terms of ACC, NMI and ARI. The simple $k$-means performs moderately well, while NCD $k$-means shows slight improvements on all metrics. The purely unsupervised spectral clustering (SC) method has the lowest performance among the methods in this study, with an ACC of only 20.2%. While KASP NCD SC achieves results comparable to NCD $k$-means, it still outperforms SC by 9.2% in terms of ACC. This confirms both the effectiveness of KASP in creating a suitable representation with a reduced number of points, and the importance of tuning the hyperparameters using the known classes, as described in Section 4.3.3.

The three neural network based methods significantly outperform the four traditional clustering methods. Although slightly worse than the baseline and PBN, the performance of TabularNCD is commendable and suggests that it is able to capture meaningful clusters within DELC's data. By design, the baseline focuses on capturing only the features of the known classes during training, before clustering the novel classes (see the discussion of Section 4.3.4). Therefore, its impressive performance reveals that there is a strong similarity between the high-level features used to distinguish the known and hidden classes. Finally, PBN is the overall top performer in this benchmark. Its ACC is comparable to the baseline, but its NMI and ARI are 2.2% and 4.4% higher respectively.

**Confusion matrix and decision trees.** Since we found that PBN performed the best among the competitors, we use the clusters it generated for the hidden classes to

illustrate the kind of results that can be expected for the novel classes (i.e. the DNIs). Figure 6.9 represents the confusion matrix of PBN's prediction against the ground truth labels. The Hungarian algorithm was used to find the best match between the cluster numbers and the class labels. To improve readability, it only includes the first 15 hidden classes out of the 28. This figure demonstrates that an ACC of 45.5% does not mean that each cluster is only 45% pure individually. Instead, we observe that some clusters fail and overlap multiple classes, and others are almost pure (e.g. clusters number 3, 7, 8 or 14).

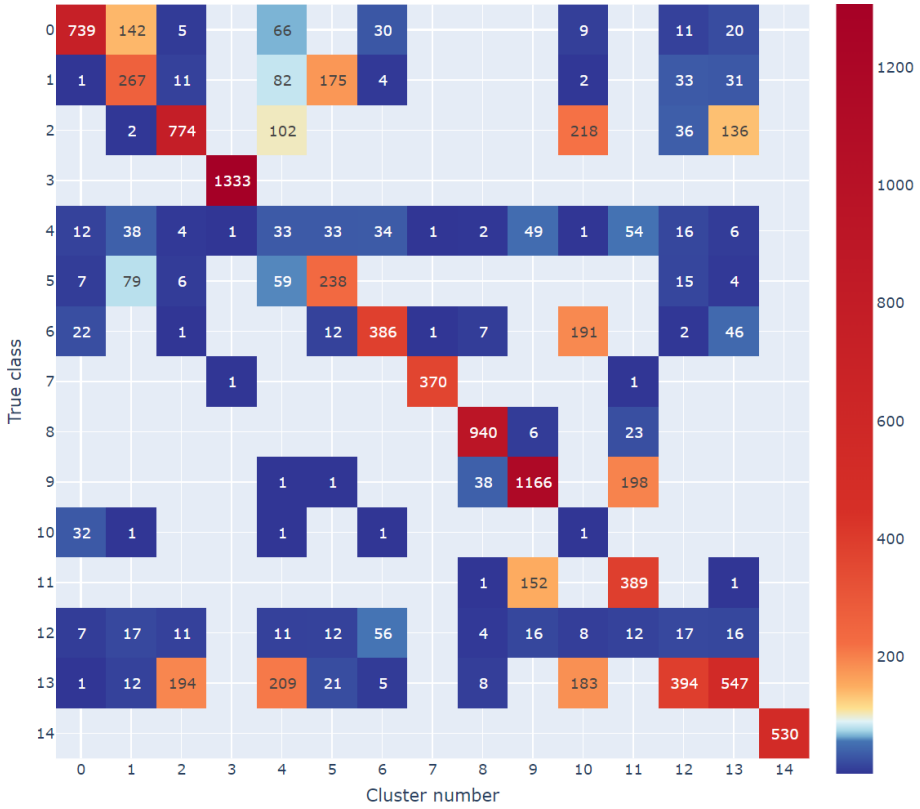



Figure 6.9 – Confusion matrix of the prediction of PBN for the first 15 hidden classes of DELC.

The last step of this series of experiments is essential to validate the results obtained so far. The objective here is to create a representation of the clusters that can be easily interpreted by the network experts. Similarly to the interface of Chapter 5, we display the decision rules of trees trained to predict the clusters. Specifically, the trees are trained using a *one-vs-rest* approach: for each cluster, a tree is trained to distinguish that cluster from all others. We chose this approach over using a single multi-class decision tree because it produces shallower and simpler sets of rules, which are closer to what can be found in

DELC's software.

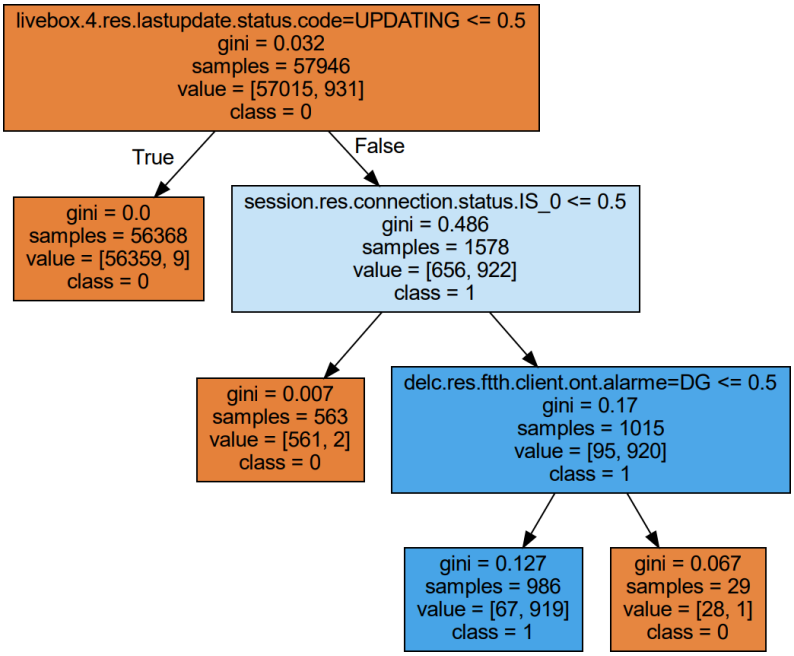



Figure 6.10 – One-vs-rest decision tree for cluster 8, visualized with the *Graphviz* library.

To illustrate with a successful example, we select cluster number 8 in Figure 6.9 for its high correspondence with the ground truth, and train a decision tree to distinguish it against all known classes. The resulting tree is shown in Figure 6.10, and achieves 99.9% classification accuracy with just 4 leaf nodes. To navigate the tree to "class = 1" in blue, which corresponds to cluster 8, the following path must be taken [6]:

— The root node "livebox.4.res.lastupdate.status.code=UPDATING <= 0.5" must evaluate to *false*, indicating that its value is 1.

— Next, the value of "session.res.connection.status.IS_0" must also be 1.

— And finally "delc.res.ftth.client.ont.alarme=DG <= 0.5" must evaluate to *true*, meaning its value is 0.

A network expert will thus understand from this figure that the faults in cluster 8 are characterized by a modem last status code of "UPDATING", a disconnected internet session and an ONT alarm different from DG (for *dying gasp*), although the last node captures very few faults and could be omitted. This interpretation corresponds very closely to DELC's diagnosis for a failed modem software update. The only difference is that it

6. Note that the features used by this decision tree are all boolean so their values are either 0 or 1.

uses the *current* internet session status instead of the attribute for the *last* connection status, and both have a very close meaning.

Finally, it is important to note that we selected a cluster which we knew was accurate thanks to the ground truth labels, but they will obviously not be available in the case of a practical exploration of the DNIs. However, we believe that network experts will be able to easily identify nonsensical rules and focus on potentially relevant ones. For example, a diagnosis is unlikely to be characterized by an attribute related to the GPON network and the battery level in the TV remote control, so such clusters can be ignored.

## 6.5 Conclusion

After briefly introducing the architecture of GPON FTTH networks, we have motivated the importance of a reliable automatic diagnosis system. While *Orange*'s DELC software is capable of diagnosing most faults, it has become difficult to maintain using traditional means. Manual investigation of undiagnosed faults by network experts is becoming increasingly difficult as the FTTH network continues to expand. Thus, in an attempt to solve this problem, we have proposed a semi-automatic process for discovering new diagnoses using the tools developed in the previous chapters of this thesis.

However, DELC is a rule-based software developed internally, capable of collecting data from multiple sources and using features in very different formats (strings, lists, identifiers, dates, ...). The first step was therefore to transform these features into a tabular dataset that could be fed to our algorithms. We collected a large number of faults and descriptive attributes over a period of 2 months to ensure that there was enough information to produce relevant results. We then followed a rigorous *data wrangling* process, where the data went through several steps such as structuring, cleaning, feature extraction or evaluation. This previously unavailable representation of the data and the scripts required to produce it were released internally and are already being used for various use-cases where machine learning can now be used to uncover difficult patterns.

In the experiments, we evaluated the performance of 7 clustering and NCD algorithms on this dataset. The undiagnosed faults are, by definition, without labels and evaluating the clusters produced by each method is impossible unless a network expert reviews many of them, which is unrealistic. Therefore, similarly to the open source datasets in the previous chapters, we hid a portion of the known classes to act as novel classes. We followed the optimization procedure introduced in Chapter 4 and found that deep-based

NCD methods significantly outperformed the other solutions, with PBN having the best performance. Finally, we validated our rule generation process by training a decision tree to predict a cluster and found that it closely resembled the true rules of DELC's software.

Chapter 7

# Conclusion and future directions

## Contents

In this dissertation, we have addressed the problem of Novel Class Discovery (NCD), which uses labeled data to more accurately cluster unlabeled data containing novel but related classes. Most of the existing work in NCD is specialized to computer vision and relies on techniques that are not applicable to tabular data, which is the focus of this thesis. Consequently, we developed new approaches specifically tailored to NCD in tabular data and demonstrated the value of our contributions through the discovery of new fault diagnoses in operational FTTH data.

In this conclusion, we first review the contributions made in each chapter and highlight their limitations in Section 7.1. Then, in Section 7.2, we discuss possible extensions of our work and interesting future research directions.

## 7.1 Summary of the contributions

We started in **Chapter 2** with a survey of the state-of-art in Novel Class Discovery. We established a categorization of existing NCD methods into two main approaches: two-stage methods that first learn a representation on the labeled data, before exploring the unlabeled data; and one-stage methods that process both sets through a joint objective function. This chapter highlights the fundamental differences between these approaches, their general workflows, strengths and weaknesses. In addition, we presented some new related tasks inspired by NCD, with Generalized Category Discovery (GCD) being of particular interest. It considers that samples in the unlabeled set can belong to either known or novel classes, making it applicable to many real-world scenarios. Finally, we provided an overview of important components commonly found in NCD algorithms, such as pseudo-labeling, contrastive learning, and methods for estimating the number of novel classes.

During this survey, we noted a lack of NCD methods for tabular data, as the majority of works focus on image data which benefits from spatial correlation between pixels. This specificity enables the use of techniques like data augmentation or contrastive learning, both of which have poor results in tabular data [18].

**Chapter 3** is dedicated to the definition of a first NCD method adapted to tabular data. Inspired by the most prominent work at that time, AutoNovel [17], we defined TabularNCD, a one-stage method where an encoder is trained by a joint objective on multiple tasks. The first is a simple classification of the known classes to include their discriminative features in the latent space. The second is a clustering of the unlabeled data

by defining pseudo-labels based on the similarity of the points in the encoder's output. And the last is a consistency objective to regularize the outputs of the classification and clustering networks. Similar to AutoNovel, the latent representation of the encoder is initialized by Self-Supervised Learning on both labeled and unlabeled data. However, ablation studies revealed that this step was not essential for the success of the method and increased its complexity for limited gains.

While TabularNCD displayed strong performance during the evaluation on public tabular datasets, the optimization process was not realistic. Indeed, both the number and true labels of the novel classes in the training set were used to optimize the model's hyperparameters. In addition, we found that the performance of TabularNCD is very sensitive to its many hyperparameters (7 in total, 4 of which are just for the loss function), making tuning a delicate process.

These limitations are the focus of **Chapter 4**, where we proposed three new models. The first two are derived from unsupervised clustering methods: *NCD k-means* enhances the initialization of centroids using the labeled data, while *NCD Spectral Clustering* (NCD SC) optimizes SC's hyperparameters based on clustering performance on the known classes. The last is a neural network based approach dubbed *Projection Based NCD* (PBN). It trains an encoder with a classification loss on the known classes and a reconstruction loss on all the data to avoid overfitting, before clustering the latent representation of the unlabeled data with $k$-means.

Importantly, this chapter addresses the realistic scenario where the labels and number of novel classes are not available during training. Thus, we defined a hyperparameter optimization technique where a portion of the known classes are hidden and used to validate the performance of different combinations of hyperparameters. And the number of clusters is estimated using Cluster Validity Indices in both the original and learned latent spaces. The intentionally low complexity and number of hyperparameters of the proposed methods proved advantageous, as the hyperparameters of PBN and NCD SC did not overfit during optimization, unlike TabularNCD.

Next, we developed and presented an interactive data exploration interface in **Chapter 5**. The objective here was to allow domain experts to easily apply the algorithms developed in the previous chapters on their datasets without requiring extensive coding and data science knowledge. In this interface, users can select which features and classes to use, visualize their data in 2 dimensions using t-SNE, configure and run NCD and clustering models, and generate interpretable decision tree rules to describe the discovered

clusters.

The idea of involving domain experts in the NCD process came from the observation that the domain- and application-specific attributes of tabular data can be difficult to grasp for data scientists unfamiliar with the domain. By allowing the domain expert to directly manipulate the data and retrain models with different parameters, the iterative process of NCD is streamlined and much less tedious than when done through coding and interaction with a data scientist.

While this interface is a valuable tool in practical applications of NCD for tabular data, it still has some limitations. For example, the decision tree rules generated to interpret the discovered clusters can become cluttered and difficult to read, especially in the multi-class case. The one-vs-rest approach helps mitigate this, but can still struggle with highly complex datasets. Lastly, there is no built-in functionality for estimating the optimal number of clusters or novel classes, and the user may have to rely on external methods or trial and error to approximate this value.

Finally, **Chapter 6** represents the synthesis and application of the concepts and methods developed throughout this manuscript, in the context of FTTH network fault diagnosis. After introducing the architecture of GPON FTTH networks, we described the limitations of *Orange*'s rule-based diagnosis system and why NCD could help network experts in their work. To this end, we collected a large number of operational faults in *Orange*'s FTTH network. Since this data was disorganized and contained information in very varied formats, extensive preprocessing was necessary to transform the data into a usable format. We then proposed a semi-automated diagnosis discovery process in which a network expert selects and refines the clusters produced by our algorithms. In the experiments, we evaluated the performance of 7 clustering and NCD algorithms on the DELC dataset and found that our deep-based approaches largely outperformed the simpler unsupervised clustering alternatives, with PBN being the top performer. We concluded by validating the rules of the decision tree we generated to interpret the clusters and observed that they closely resembled the true rules in the DELC software.

Evaluating the models when the number of novel classes is not known in advance is an important aspect that was not explored in the experiments and should be done in the future. We also did not have the opportunity to test the process of Figure 6.7 on undiagnosed faults, as some additional time and engineering is required to fully develop the system. Nevertheless, the encouraging results of the experiments indicate that this is the right approach in this specific context.

## 7.2 Future directions

### 7.2.1 Selecting the most relevant clusters

When trying to discover a large number of new classes in a dataset, interpreting the generated clusters with decision trees can become difficult, as multi-class decision trees will quickly contain an overwhelming number of nodes. Using a one-vs-rest approach spreads the complexity over many trees, but can still be problematic when the number of features is high and the clusters are convoluted. Add to this the fact that NCD or clustering algorithms must be fine tuned until the domain expert is satisfied with the results, and the number of trees to be examined can quickly become enormous. The dataset we created in Chapter 6 has 537 features, so it is clearly affected by this problem. Thus, we need to find a way to optimize the limited time that domain experts can dedicate to this task.

One way to address this challenge of efficiently interpreting a large number of clusters might be to rank the clusters from most to least likely to be correct. By automatically determining the most relevant clusters, domain experts could focus their efforts on the most promising clusters first. This can be achieved, for example, by sorting the clusters according to some Cluster Validity Indices (CVIs) [139]. In Chapter 4, we used CVIs to estimate the number of novel classes by summarizing the entire clustering prediction of a model into a single quality score. And since some of these CVIs are defined as an average of the individual scores they assign to each cluster, the individual scores could be used to rank the clusters from most to least relevant. A good example is the Silhouette Index [166]:

$$S_{sil} = \sum_{n=1}^{k} \frac{1}{|I_n|} \sum_{i \in I_n} \frac{b(i) - a(i)}{max(a(i), b(i))} \tag{7.1}$$

with $k$ the number of clusters, $I_n$ the set of indices of points belonging to cluster $n$, $a(i)$ the mean distance of point $i$ to all other points in the same cluster and $b(i)$ the smallest mean distance of point $i$ to the points in any other cluster. The clusters could then be ranked according to the average of the Silhouette coefficients (the fraction in Equation 7.1) of all points in a cluster. Other examples of scores that can be diverted for this application include the inertia, the distortion and the Davies-Bouldin index [167]. Note that depending on the NCD or clustering algorithm used, these scores may give more accurate results when computed in the latent or original feature space. Similar to the experiments of Chapter 6, the best approach can be selected by finding the CVI that best correlates with the ground truth accuracy of the clusters on some hidden classes.

Stable clusters that appear across different executions or with slight variations in the clustering algorithm are also more likely to be relevant. Intuitively, if a cluster persists across multiple runs of the same model (or even across different algorithms), it increases the confidence that this cluster reflects a genuine pattern within the data. While we could not find research specifically focused on identifying consistent clusters across algorithms, the field of Consensus Clustering seems to share similar concepts [168].

Finally, in addition to validation metrics, the number of points in a cluster may be a useful metric for domain experts. Clusters with very few points may not be reliable, whereas clusters with many points may indicate dominant patterns in the data.

### 7.2.2 NCD and human-in-the-loop

Throughout this thesis, we have explored ways to improve the clustering of the novel classes based only on the knowledge contained in the known classes. Yet, it should be possible to involve a domain expert in the learning process through human-in-the-loop (HITL) [169] strategies to further refine the results. Domain experts can provide valuable insights by evaluating the clusters produced by our algorithms and suggest improvements. In this section, we describe some ways to integrate this feedback into our models and iteratively refine the results.

This problem should first be approached by asking what questions can realistically be answered by a human expert. An impractical example would be to ask someone to tell for all pairs of points in a cluster whether they should be grouped together. Indeed, given the large number of features in our dataset, even comparing two points will be slow. Instead, questions should be formulated precisely to make the best use of the limited time that humans can spend on such a task. Some examples are given below:

*"Should these two points be in the same cluster?"* is very similar to the previous one, but is of a much smaller scale. This could be easily implemented by must- and cannot-link constraints, and could be taken into account by using COP-Kmeans [85] or PCKMeans [86]. These constrained clustering algorithms can simply replace the $k$-means in the latent space of PBN or in the spectral embedding of NCD Spectral Clustering. In the case of TabularNCD, which does not directly rely on a clustering algorithm like $k$-means, this pairwise information could be integrated into the training with a contrastive loss. This

could, for example, be achieved with the supervised contrastive learning loss [53]:

$$\ell_i = -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp\left(z_i \cdot z_p / \tau\right)}{\sum_{a \in A(i)} \exp\left(z_i \cdot z_a / \tau\right)} \tag{7.2}$$

with $z_i = \phi(x_i)$ the projection of $x_i$ in the latent space, $\tau$ a temperature parameter, $A(i) = I \backslash \{i\}$ [1] and $P(i)$ the set of indices of the points that were labeled as positives relative to $i$ by the expert. By adding this objective function to the model during training, the points in $P(i)$ are pulled together, while the rest are pushed apart. However, we warn the reader that to be effective, the contrastive loss may require many labeled pairs, or heavy data augmentation which has been observed to perform less favourably with heterogeneous tabular data than with image data [18]. A possible solution would be to locally approximate the expert's feedback by generalizing the labels assigned to the pairs of points to the immediate neighborhood.

*"Should these two features be important in the same cluster?"* is difficult to formulate due to its subjectivity. To illustrate this question, we can argue, for example, that it would be unexpected for a feature related to the quality of the Wi-Fi and a feature expressing the configuration of the GPON network to be present in the same diagnosis. Thus, instead of simply dismissing such a cluster as nonsensical, an expert could justify the reasons behind his or her decision. The importance of the features in a cluster could be measured by the Gini importance [158] of a random forest classifier trained to predict the cluster labels.

Finally, *"Should we group some of these clusters?"* can be easily implemented by comparing the closest clusters first, and re-generating their interpretation if the expert decides to merge them. And the opposite can also be done by asking if some clusters should be split.

### 7.2.3 Other directions

We finish this conclusion chapter by briefly presenting several ideas that we were not able to develop in this manuscript, but that we believe are still important topics for the resolution of the NCD problem in practical scenarios.

In this work, we relied heavily on the assumption that none of the samples in the unlabeled sets belonged to known classes. However, it is likely that the coverage of DELC's rules is incomplete, and that some faults labeled as DNIs actually belong to known cat-

1. $I = \{1...N\}$ the indices of all points in the dataset

egories. While it is impossible to determine the exact number of faults affected by this problem, violating NCD's assumption that $\mathcal{Y}^l \cap \mathcal{Y}^u = \emptyset$ could still have an important impact on performance. If a significant portion of the unlabeled data is confirmed to belong to the known classes, it may be worthwhile to explore the literature of Generalized Category Discovery (GCD), where $\mathcal{Y}^l \subset \mathcal{Y}^u$. The more flexible setting of the GCD field probably contributes to its growing popularity. In 2022, we reviewed 20 papers and found 14 focused on NCD and 6 on GCD. In 2023, out of 29 papers, 9 addressed NCD and 20 focused on GCD.

Instead of representing data in the usual Euclidean space, hyperbolic neural networks typically project data into a Poincaré ball where the curvature is negative. In this unusual data representation, the distance from the origin increases exponentially as you move away from it. This natural tree-like structure of hyperbolic spaces caused by their negative curvature makes them well suited for representing hierarchical data [170]. The rules of DELC's software are similar to decision trees and contain multiple families of diagnoses (FTTH, Wi-Fi, CPL, ONT, ...). This suggests a hierarchical nature of the diagnoses, which could be leveraged by using a hyperbolic representation. However, while hyperbolic neural networks have shown promising results for clustering in computer vision and natural language processing, they have not been extensively studied on tabular data and still suffer from instability during training [171, 172].

# Résumé en Français

Cette section est un résumé substantiel en Français de ce manuscrit de thèse. Cet exercice recommandé par l'école doctorale permet non seulement de rendre la thèse plus accessible aux lecteurs francophones qui ne maîtrisent pas l'anglais, mais aussi de promouvoir l'usage de la langue française dans le domaine académique et scientifique. Dans les pages suivantes, nous introduirons ainsi le thème de cette thèse et motiverons son importance, avant de résumer le contenu et les contributions de chacun des chapitres.

## Introduction

### L'importance de la fiabilité des réseaux de télécommunication

La dépendance de l'économie mondiale aux services Internet s'est beaucoup accentuée au cours de ces dernières années. La communication instantanée, la gestion de projets, le marketing ou le commerce électronique sont désormais des éléments essentiels des opérations commerciales modernes, facilitant les échanges à l'échelle mondiale. La fibre optique a joué un rôle particulièrement crucial dans ces avancées en améliorant considérablement les vitesses Internet. Selon l'ARCEP, le nombre de foyers français éligibles aux offres de Fibre-To-The-Home (FTTH) est passé de 10,9 millions en 2018 à 38,9 millions en 2023 [3], couvrant plus de 84% du marché de l'accès Internet fixe. La fibre optique a permis l'intégration d'applications à haut débit et faible latence telles que le streaming vidéo, les jeux en ligne, l'informatique en nuage (ou *cloud computing*), la télémédecine et les infrastructures de villes intelligentes dans la vie quotidienne. Cette évolution est appelée à se poursuivre avec la connexion de plus en plus d'appareils et de services gourmands en bande passante.

Les Fournisseurs d'Accès à Internet (FAI) sont responsables de la livraison et de la maintenance de la connectivité Internet. Ils construisent et maintiennent l'infrastructure du réseau en posant de la fibre optique, en installant des routeurs, des commutateurs, des centres de données et d'autres équipements nécessaires à la transmission des données. Ils gèrent également la qualité du service, garantissant aux utilisateurs des connexions

Internet fiables et à haute vitesse. Cependant, avec un réseau en constante évolution composé d'une grande variété d'équipements, il devient de plus en plus difficile de fournir des services de qualité avec une durée de fonctionnement optimale. Même avec du matériel et des logiciels fiables, des pannes se produiront inévitablement, affectant l'expérience utilisateur. Selon une étude de 2023 de l'Uptime Institute [5], 70% des pannes de centres de données coûtent aux entreprises plus de 100 000 dollars par panne. Certaines de ces pannes peuvent être spectaculaires, comme en 2016, lorsqu'une seule défaillance de routeur a conduit Delta Airlines à annuler 2 300 vols et à perdre 177 millions de dollars. À plus petite échelle, la résolution de ces pannes doit aussi être rapide, car plusieurs FAI sont souvent en concurrence pour les mêmes abonnés, et les utilisateurs peuvent passer chez des fournisseurs concurrents s'ils ne sont pas satisfaits. Par conséquent, une gestion efficace des pannes est un aspect clé de l'exploitation des réseaux de télécommunications, en particulier dans les réseaux d'accès.

## Le diagnostic automatique des pannes

Dans le passé, le diagnostic des pannes était fait manuellement par des experts du réseau qui examinaient chaque panne individuellement via un nombre limité d'attributs descriptifs. Cependant, l'évolution des réseaux de télécommunications au cours des deux dernières décennies a considérablement augmenté leur taille et leur complexité, et le diagnostic est aujourd'hui devenu trop complexe pour des experts humains. C'est pourquoi, en pratique, le diagnostic des pannes est principalement réalisé par des systèmes experts (souvent des algorithmes à base de règles) [9]. De la même manière qu'un expert étudie un cas spécifique, ces systèmes analysent un grand nombre d'indicateurs de performance avant de parvenir à une conclusion sur l'origine de la panne. Ces systèmes représentent les connaissances des experts du réseau sous forme de règles dont les décisions sont faciles à interpréter. Cependant, ces solutions sont très spécifiques et ne peuvent pas être transférées à d'autres domaines d'application ou architectures de réseau. De plus, elles nécessitent une maintenance manuelle et continue de la part des experts du réseau, ce qui devient de plus en plus difficile à mesure que le volume de données et le nombre de sources potentielles de pannes augmentent. Enfin, comme les règles ne couvrent pas tous les cas possibles, certaines pannes peuvent ne correspondre à aucune des règles et ne pas être diagnostiquées. Ces pannes non diagnostiquées nécessiteront un temps d'investigation supplémentaire et augmenteront les coûts d'exploitation du FAI.

*Orange*, l'entreprise dans laquelle j'ai réalisé ma thèse, est un Fournisseur d'Accès à

Internet qui développe rapidement ses services de fibre optique. Comme la plupart des FAIs, *Orange France* a conçu son propre système de diagnostic expert. Appelé DELC (pour Diagnostic Expert de la Ligne Client), il est capable de fournir un diagnostic pour la majorité des pannes, mais certaines restent non diagnostiquées et nécessitent souvent une investigation coûteuse de la part d'un technicien. Avec des millions d'interventions de techniciens chaque année, c'est un sujet d'une importance cruciale pour *Orange*.

L'objectif de cette thèse est donc d'essayer de découvrir automatiquement de nouveaux diagnostics pour les pannes pour lesquelles le système DELC n'a pas pu déterminer la cause racine. Dans ce manuscrit, nous considérons que nous disposons d'un ensemble de fautes diagnostiquées (les classes connues) et d'un ensemble de fautes non diagnostiquées (les nouvelles classes) que nous devons explorer. Idéalement, le système que nous allons concevoir ne sera pas spécifique à DELC et pourra être utilisé avec d'autres architectures de réseau ou pour d'autres problèmes. Nous allons ainsi tenter de créer une méthode qui peut être appliquée pour partitionner n'importe quel ensemble de classes inconnues, étant donné un ensemble de classes connues.

## Le diagnostic en tant que problème d'apprentissage automatique

Dans cette thèse, nous abordons le problème du diagnostic automatique des pannes en se focalisant uniquement sur les données. En effet, modéliser entièrement le réseau d'un Fournisseur d'Accès à Internet n'est pas faisable en pratique, car c'est une tâche monumentale et qui nécessite une maintenance continue. Ici, nous supposons que pour chaque panne, le FAI a collecté un certain nombre de caractéristiques décrivant l'état du réseau au moment de la panne. Les descriptions de pannes sur une période de plusieurs mois pourront alors être compilées pour former notre jeu de données.

Le système expert à base de règles d'*Orange*, DELC, diagnostique entre 80% et 90% des pannes FTTH. Les pannes non diagnostiquées sont étiquetées comme DNIs (pour Défauts Non Identifiés). Nous pouvons ainsi considérer que nous avons deux ensembles de données distincts : l'un composé des diagnostics connus où toutes les pannes ont été étiquetées par le système expert, et l'autre composé des DNIs non étiquetés. Les données collectées sont composées d'un grand nombre d'attributs descriptifs hétérogènes tels que des niveaux de signal, des versions logicielles, des horodatages, des statuts d'équipements, etc. Après pré-traitement, les données seront structurées sous forme de tableau, chaque ligne représentant une panne et chaque colonne une caractéristique.

En explorant les sous-domaines de l'*open-world learning* (voir Chapitre 2), nous avons

découvert le domaine émergent de la Découverte de Nouvelles Classes (*Novel Class Discovery*, NCD). Très similaire à notre problème, il y a pendant l'entraînement un ensemble étiqueté de classes connues et un ensemble non étiqueté de classes différentes et inconnues. Et l'objectif est de découvrir les classes sous-jacentes dans les données non étiquetées. Le premier article majeur de NCD a été publié en 2021 [17], et ce domaine est resté largement focalisé sur les données d'images depuis lors. Comme il existe une forte corrélation spatiale entre les pixels d'une image, des techniques telles que la convolution, l'augmentation des données ou l'apprentissage auto-supervisé (*Self-Supervised Learning*, SSL) peuvent être employées pour améliorer les performances. Dans notre cas, les attributs des données tabulaires sont hétérogènes et il n'y a pas de corrélation spatiale. Ainsi, ces techniques ne peuvent pas être utilisées et les méthodes de NCD de l'état de l'art ne peuvent pas directement être appliquées. Cependant, nous pensons que la philosophie générale et les concepts derrière le NCD peuvent être transférés à notre problème. Nous considérons donc dans cette thèse que notre problème de diagnostic de pannes est un problème de Découverte de Nouvelles Classes dans des données tabulaires.

## Chapitre 2 : État de l'art

Nous avons commencé par une étude de l'état de l'art en matière de Découverte de Nouvelles Classes. Après avoir défini formellement les hypothèses et composants clés des méthodes de NCD, nous les avons catégorisées en fonction de la manière dont les connaissances de l'ensemble étiqueté sont utilisées pour partitionner l'ensemble non étiqueté. Nous avons constaté que les approches à deux étapes, qui s'entraînent d'abord uniquement sur les données étiquetées avant de partitionner l'ensemble non étiqueté, ont rapidement perdu de leur popularité. Leur risque de sur-entraînement sur les classes connues a mené les chercheurs à définir des méthodes à une seule étape où les deux ensembles sont utilisés à la fois et qui sont maintenant largement adoptées. Cette taxonomie devrait aider à orienter les recherches futures en donnant un aperçu clair des familles d'approches et de techniques qui ont déjà été explorées. L'émergence du NCD a conduit à la création de nouveaux domaines où les chercheurs assouplissent les hypothèses du NCD et que nous avons également analysés. De plus, nous avons identifié et présenté les techniques et outils couramment utilisés en NCD. Enfin, étant donné qu'il s'agit d'un nouveau domaine qui se situe à l'intersection de plusieurs autres, il peut être difficile de le distinguer des autres domaines de recherche. Nous avons donc présenté les domaines les plus proches et précisé

leurs principales différences.

Malgré le nombre croissant de travaux dans ce domaine, nous avons identifié plusieurs questions restant sans réponse et certaines perspectives qui méritent d'être étudiées plus en détail. Comme nous l'avons déjà exprimé, la majorité des travaux de NCD sont appliqués uniquement aux données d'images et profitent de puissantes techniques telles que l'augmentation de données et l'apprentissage auto-supervisé qui reposent sur la structure unique des images. Ces techniques sont en partie responsables du succès des méthodes de NCD, et comme elles ne sont pas directement applicables à d'autres types de données, la plupart des travaux sont encore limités aux données d'images. Cependant, il serait intéressant d'explorer leur potentiel avec d'autres types de données comme le texte ou les tableaux. DTC [37] a montré que les méthodes de partitionnement à base de réseaux de neurones peuvent facilement être transférées au problème de NCD, et nous pensons que davantage d'entre elles pourraient être adaptées. Ensuite, les méthodes de NCD ne devraient pas supposer que le nombre de nouvelles classes est connu à l'avance, et même si certaines approches ont été proposées pour le déterminer automatiquement, leur précision est limitée. Nous pensons également qu'il est important d'avoir un protocole d'évaluation unifié, car des travaux ont montré que la sélection des classes connues/nouvelles a une influence sur la difficulté du problème de NCD [40]. Enfin, nous avons remarqué que le pseudo-étiquetage est une technique largement utilisée dans les méthodes en une étape. Leur performance est un facteur décisif pour le succès de ces méthodes et il y a encore des améliorations possibles dans ce domaine, par exemple, en tenant compte des données étiquetées, ou en s'inspirant de la théorie des graphes ou du partitionnement spectral.

# Chapitre 3 : Première méthode

Dans ce chapitre, nous avons défini une première méthode, TabularNCD, pour la découverte de nouvelles classes dans les données tabulaires. Inspirée par le travail d'AutoNovel [17], TabularNCD repose principalement sur la définition de pseudo-étiquettes et sur l'optimisation d'un objectif joint sur nos deux ensembles de données. Son entraînement commence par l'initialisation de son espace latent à l'aide d'une nouvelle technique d'apprentissage auto-supervisé pour les données tabulaires, VIME [55]. Le partitionnement des données non étiquetées est ensuite réalisé en se basant sur l'idée que le voisinage local d'une instance dans l'espace latent appartient probablement à la même classe. Ce processus est optimisé conjointement avec un classificateur sur les classes

connues afin d'inclure leurs caractéristiques importantes dans l'espace latent.

Les contributions que nous faisons dans ce chapitre sont les suivantes :

1. Proposition d'une nouvelle méthode pour la découverte de nouvelles classes dans des données tabulaires.
2. Évaluation empirique de cette méthode sur sept ensembles de données publics, démontrant sa performance supérieure par rapport à des méthodes entièrement non supervisées et une méthode exploitant les classes connues de manière naïve.
3. Introduction d'une approche originale pour la définition de pseudo-étiquettes, exploitant le voisinage local d'une instance dans un espace latent pré-entraîné.
4. Réalisation d'expériences pour comprendre les raisons de l'avantage de la méthode proposée par rapport à des approches plus simples.

Nous avons également comparé TabularNCD à une méthode de découverte de nouvelles classes pour les images, CD-KNet, sur l'ensemble de données MNIST, et avons obtenu des résultats comparables, démontrant que notre méthode est efficace même sans les avantages des réseaux de neurones convolutifs utilisés pour les images. Nous avons ensuite réalisé une étude d'ablation pour évaluer l'importance de chaque composant de notre méthode. Enfin, nous avons visualisé l'évolution de la représentation des données au cours de l'entraînement et observé que notre modèle sépare progressivement les échantillons en groupes de même classes.

En conclusion, notre travail démontre que le domaine de la Découverte de Nouvelles Classes est non seulement applicable aux images mais aussi aux données tabulaires hétérogènes, ouvrant ainsi de nouvelles perspectives pour l'apprentissage en monde ouvert dans divers domaines d'application.

# Chapitre 4 : Approches pratiques

Dans ce chapitre, nous focalisons notre travail sur les limitations de la méthode proposée dans le chapitre précédent. En effet, comme il est souvent le cas dans les articles de NCD, le nombre de nouvelles classes y est supposé connu à l'avance, et leurs étiquettes on été utilisées pour optimiser les hyperparamètres. Les méthodes qui font ces hypothèses deviennent alors difficiles à appliquer dans des scénarios pratiques. Nous nous concentrons donc ici sur la résolution du NCD dans les données tabulaires sans aucune connaissance préalable des nouvelles classes.

Pour ce faire, nous proposons d'adapter le processus de validation croisée à $k$ blocs en cachant certaines des classes connues dans chaque bloc pour optimiser les hyperparamètres des méthodes de NCD. Nous avons constaté que les méthodes avec trop d'hyperparamètres ont tendance à sur-apprendre ces classes cachées. Par conséquent, nous définissons un modèle simple de NCD à base de réseaux de neurones, composé uniquement des éléments essentiels au problème de NCD, et montrons qu'il offre des performances robustes dans des conditions réalistes. De plus, nous observons que l'espace latent de cette méthode peut être utilisé pour estimer de manière fiable le nombre de nouvelles classes.

Nous adaptons également deux algorithmes de partitionnement non supervisé ($k$-means et partitionnement spectral) pour qu'ils tirent parti des connaissances des classes connues. Des expériences sont menées sur sept ensembles de données tabulaires, démontrant l'efficacité des méthodes proposées et du processus d'ajustement des hyperparamètres. Ces expériences montrent également que le problème de NCD peut être résolu dans des conditions réalistes où aucune connaissances a priori des nouvelles classes ne sont disponibles.

## Chapitre 5 : Interface d'exploration de données

Les experts d'un certain domaine sont les plus à même d'interpréter les résultats des algorithmes de NCD ou de partitionnement non supervisé. Mais en pratique, ils ne disposent pas forcément des compétences requises en rédaction de code ou en science des données pour mettre en œuvre ces algorithmes. Dans ce chapitre, nous présentons une interface interactive d'exploration de données qui permet à ces experts d'appliquer facilement les algorithmes développés dans les chapitres précédents sur leurs ensembles de données sans avoir à rédiger de code. Dans cette interface, les utilisateurs peuvent sélectionner les attributs et les classes à utiliser, visualiser leurs données en deux dimensions à l'aide de t-SNE, configurer et exécuter des modèles de NCD et de partitionnement, et générer des règles interprétables sous forme d'arbres de décision pour décrire les classes découvertes.

L'idée d'impliquer des experts du domaine dans le processus de NCD est venue de l'observation que les attributs des ensembles de données tabulaires sont toujours spécifiques à l'application et peuvent être difficiles à interpréter pour les data scientists. En permettant aux experts de manipuler directement les données et de réentraîner les modèles avec différents paramètres, le processus de découverte de nouvelles classes est simplifié et beaucoup moins fastidieux que lorsqu'il est effectué en interaction avec un data scientist.

# Chapitre 6 : Application au diagnostic de pannes FTTH

Ce chapitre représente la synthèse et l'application des concepts et méthodes développés tout au long de cette thèse dans le contexte du diagnostic des pannes des réseaux FTTH (Fiber-To-The-Home). Après avoir introduit l'architecture des réseaux FTTH GPON, nous avons décrit les limitations du système de diagnostic automatique à base de règles d'*Orange*, appelé DELC (Diagnostic Expert de la Ligne Client). Depuis son introduction en 2012, le taux de diagnostic du système DELC est passé de 48% à 85% en 2023. Cependant, ce taux a récemment cessé de progresser, et les experts du réseau peinent à trouver de nouveaux diagnostics. Les 15% restants doivent souvent être investigués par des techniciens, entraînant des coûts d'exploitation élevés pour *Orange*.

Les outils de NCD que nous avons développés dans les chapitres précédents de cette thèse sont une approche que les experts du réseau n'ont pas encore explorée. Nous espérons ainsi qu'ils offriront une nouvelle perspective à la découverte de diagnostics. L'objectif n'est pas de remplacer le système actuel, mais de suggérer de nouveaux diagnostics aux experts du réseau à partir de groupes de pannes similaires non diagnostiquées.

Pour ce faire, nous avons commencé par collecter un grand nombre de pannes opérationnelles du réseau FTTH d'*Orange*. Ces données étaient désorganisées et contenaient des informations dans des formats très variés, un prétraitement approfondi était donc nécessaire pour transformer les données en un format utilisable. Nous avons structuré ces données en une table CSV, nettoyé les erreurs et les valeurs manquantes, et enrichi le jeu de données avec des informations supplémentaires pertinentes. Nous avons également filtré les attributs pour réduire la dimensionnalité du jeu de données, en utilisant des méthodes de sélection de caractéristiques non supervisées.

Dans les expériences, nous avons comparé tous les algorithmes de partitionnement et de NCD utilisés au cours de cette thèse en fonction de plusieurs métriques Les méthodes basées sur les réseaux neuronaux, telles que TabularNCD et Projection-Based NCD (PBN), ont significativement surpassé les autres solutions, avec PBN obtenant les meilleurs résultats. Enfin, nous avons validé notre processus de génération de règles en entraînant un arbre de décision pour prédire un groupe de données et avons constaté qu'il ressemblait étroitement aux règles réelles du logiciel DELC. Nous avons conclu que notre approche semi-automatique pour découvrir de nouveaux diagnostics pourrait aider les experts du réseau à identifier et corriger plus rapidement les pannes, réduisant ainsi les coûts opérationnels du réseau FTTH d'*Orange*.

# Conclusion et perspectives

Cette thèse a abordé le problème de la découverte de nouvelles classes dans les données tabulaires en développant des méthodes originales et en les appliquant à un cas réel de diagnostic de pannes des réseaux FTTH. Nous avons montré que les méthodes basées sur les réseaux de neurones, en particulier PBN, sont efficaces pour résoudre ce problème dans des scénarios réalistes où aucune connaissance a priori n'est disponible.

Pour les travaux futurs, nous proposons plusieurs directions de recherche. Tout d'abord, il serait utile de développer des méthodes pour sélectionner automatiquement les groupes de données les plus pertinents afin d'optimiser le temps limité que les experts du domaine peuvent consacrer à cette tâche. Ensuite, l'intégration de l'expert du domaine dans le processus d'apprentissage par des stratégies de *human-in-the-loop* pourrait améliorer les résultats. Enfin, il serait intéressant d'explorer l'utilisation des réseaux de neurones hyperboliques pour représenter les données, car ils sont adaptés pour la représentation de données hiérarchiques comme celles des réseaux Internet.

# Appendices

Appendix A

# Supplementary materials of Chapter 3

## A.1 Comparison of the pseudo-labeling methods

**Hyperparameter range**

In this section, we compare the performance and stability of two pseudo-labeling methods. The first is the threshold based method of Equation 2.9 (see Section 2.5.2), which is used in several NCD articles [45, 39]:

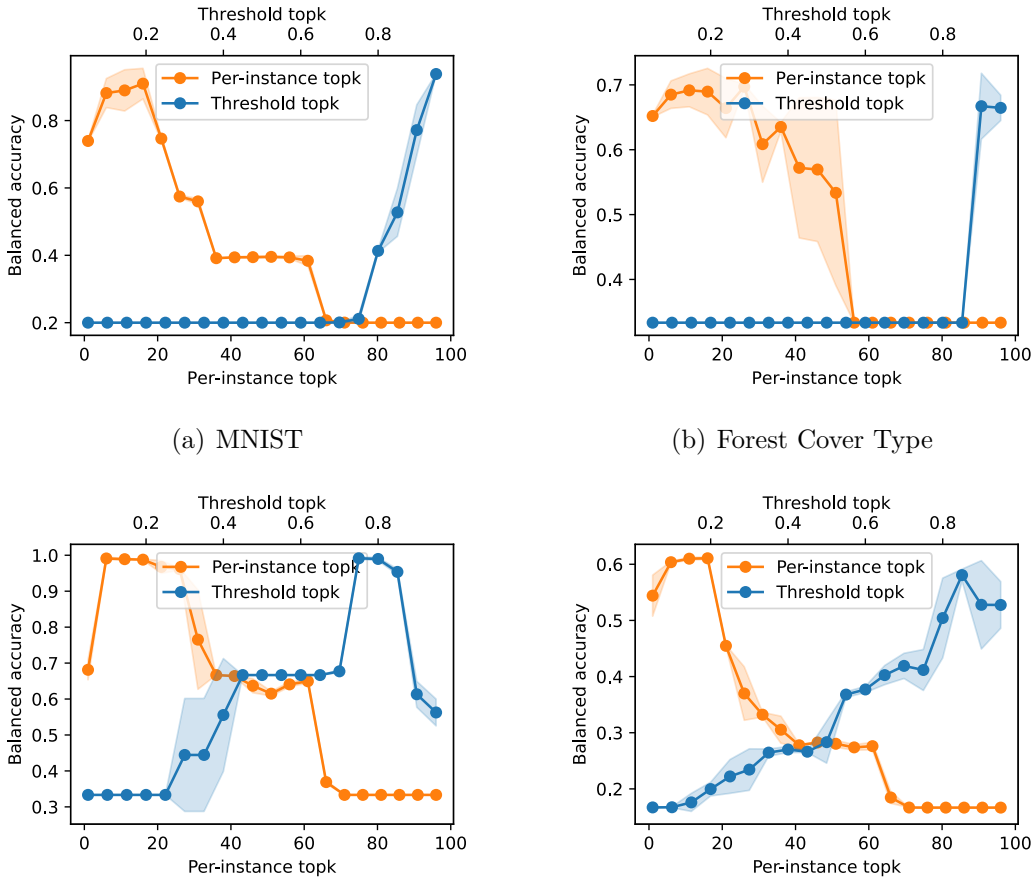
$$\hat{y}_{i,j} = \mathbb{1}[\delta(z_i, z_j) \geq \lambda]$$

And the second is the *top k* based method of Equation 3.6, that we propose in Section 3.3.2:

$$\hat{y}_{i,j} = \mathbb{1}[j \in \underset{\substack{r \in \{1,\dots,|Z|\} \\ r \neq i}}{\text{argtop}_k}\ \delta(z_i, z_r)]$$

To determine which pseudo-labeling methods is better, the performance of TabularNCD is evaluated with both methods for different values of their hyperparameter. For the threshold based method, the value of $\lambda$ is varied in $[0, 1]$, and for the *top k* based method, the percentage of the pairs that are positive is varied in $[0\%, 100\%]$.

The results of this experiment are displayed in Figure A.1 and are averaged over 10 executions for 4 different datasets. They reveal that the *top k* based method obtains similar or slightly superior performance over the threshold based method. However, we observe that the ideal range of values is larger for the *top k* based method, which makes hyperparameter tuning easier.




(a) MNIST

(b) Forest Cover Type

(c) Human Activity Recognition

(d) US Census 1990

Figure A.1 – Balanced accuracy w.r.t to the pseudo-labeling method, for all of their possible values.

**Clustering collapse**

During the experiments, we also noticed that with the threshold based method, the clustering network would sometimes *collapse*, and degenerate to a trivial solution where all instances are given the same cluster label (see Figure A.2). This problem arises due to the fact that the cosine similarity is measured in the latent space, which itself is updated during learning. By bringing all the data points very close to each other, the cosine similarity between all pairs of points can become higher that the threshold $\lambda$, and all the pairs will be regarded as positive in Equation 2.9. In this case, the model easily minimizes the loss and degenerates to a trivial solution with a single cluster.

In contrast to the threshold based method, the *top k* based method uses a *number* of positive and negative pairs which is constant during training. This collapse phenomenon

therefore cannot happen in this case, which is another advantage of the proposed pseudo-label definition method.

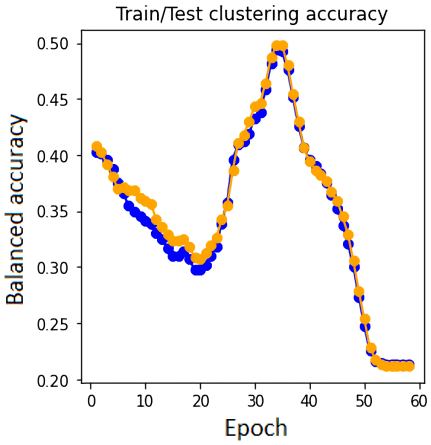



Figure A.2 – Evolution of the balanced accuracy during training, where the clustering collapses.

## A.2 Hyperparameters importance

In Chapter 3, a total of 8 hyperparameters were incrementally introduced for TabularNCD. And the optimal values chosen by the tuning process are reported in Appendix A.4. As stated throughout this manuscript, this large number of hyperparameters that must be tuned is the main weakness of TabularNCD. Here, we summarize these hyperparameters in Table A.1 and assess their importance according to the impact they have on the final clustering performance to determine if some can be excluded from the optimization process.

Table A.1 – Summary of the hyperparameters tuned in TabularNCD.

| Parameter | Description | Importance |
|---|---|---|
| *top k* | Number of positive pairwise labels for each point in a batch | High |
| $w_1$ | Trade-off between the classification and regularization terms | Low |
| $w_2$ | Trade-off between the clustering and regularization terms | High |
| $lr_{classif.}$ | Learning rate of the optimizer for the classification task | Medium |
| $lr_{cluster.}$ | Learning rate of the optimizer for the clustering task | Medium |
| $k$ neighbors | Number of points in the neighborhood of a sample in SMOTE | Low |
| dropout | Probability of zeroing inputs between each layers | Medium |
| activation fct. | Activation function used between each layer of the encoder | Low |

The importance of each hyperparameter depicted in Figure A.3 was calculated by the framework we used to automate the hyperparameter search, *Weights & Biases*. Its approach is to train a random forest with the hyperparameter configurations as inputs and the selected model's performance metric as the target output. The importance of each hyperparameter is then defined as the feature importance values of the trained random forest model. In Figure A.3, we report the average importance computed during the optimization of TabularNCD over the 7 datasets of Chapter 3.

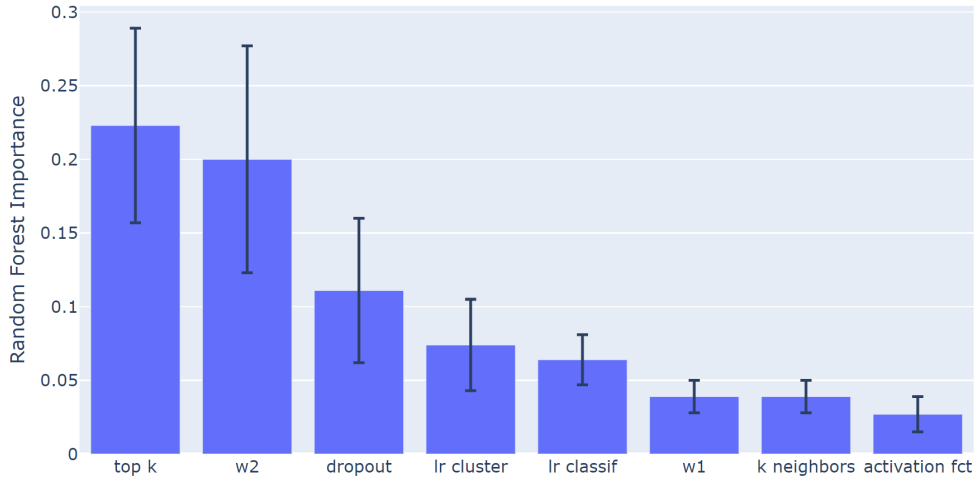



Figure A.3 – Average random forest hyperparameter importance for TabularNCD.

We observe three groups of hyperparameters:

— **High importance ($\geq$ 0.20):** *top* $k$ and $w_2$.

— **Medium importance (0.05 - 0.20):** dropout, $lr_{cluster.}$ and $lr_{classif.}$.

— **Low Importance ($<$ 0.05):** $w_1$, $k$ neighbors and activation function.

Unsurprisingly, *top* $k$ has the most impact on the performance since it controls the pseudo-labeling process, which directly determines the results produced by the clustering network. For similar reasons, $w_1$ is almost as important as *top* $k$. It balances the weight of the binary cross entropy loss of the clustering network with the regularization term.

On the other hand, $w_1$, $k$ neighbors and the activation function were always the least important hyperparameters across the 7 datasets. Given their low impact on performance, we recommend fixing $w_1 = 0.8$, $k$ neighbors $= 10$, and the activation function $=$ ReLU in future experiments. This should reduce the search space and save computational resources.

## A.3 Influence of data representation for the k-means

In table A.2, the quality of different data representations of MNIST are compared using $k$-means. We observe that after Self-Supervised Learning, the encoder learned a representation of the original data where the $k$-means algorithm performs slightly better than on the original data (+2.8% in accuracy). And after joint training, the novel classes are even more cleanly separated, as $k$-means gains +24.3% in accuracy over the original data representation.

Table A.2 – Clustering performance of $k$-means on the new classes of MNIST for different data representations.

| Data representation | BACC | ACC | NMI | ARI |
|---|---|---|---|---|
| (1) Original data | 60.1 | 61.1 | 48.1 | 38.2 |
| (2) Latent w/ SSL | 63.8 | 63.9 | 50.1 | 41.4 |
| (3) Latent w/ joint | 84.9 | 85.4 | 74.8 | 72.8 |

(1) is the original data, (2) is the data projected by an encoder trained with SSL (Section 3.3.1), and (3) is the data projected by an encoder jointly trained by the TabularNCD (Section 3.3.2).

## A.4 Hyperparameters values

We optimize the most important hyperparameters for each dataset and report their values in table A.3. The constants are: *batch size = 512*, *encoder layers sizes = [d, d, d]*, $\alpha$ *= 2.0*, *ssl lr = 0.001*, *epochs = 30* and *p_m = 0.30*.

Table A.3 – Best hyperparameters found.

| Parameter | MNIST | Forest | Letter | Human | Satimage | Pendigits | Census |
|---|---|---|---|---|---|---|---|
| cosine *top k* | 15.015 | 19.300 | 2.019 | 15.277 | 6.214 | 5.609 | 13.96 |
| w1 | 0.8709 | 0.1507 | 0.4887 | 0.4560 | 0.80 | 0.7970 | 0.8104 |
| w2 | 0.6980 | 0.8303 | 0.9350 | 0.6335 | 0.8142 | 0.8000 | 0.9628 |
| $lr_{classif.}$ | 0.009484 | 0.001876 | 0.009906 | 0.001761 | 0.007389 | 0.006359 | 0.005484 |
| $lr_{cluster.}$ | 0.0009516 | 0.007191 | 0.007467 | 0.001017 | 0.008819 | 0.009585 | 0.0008563 |
| *k neighbors* | 4 | 9 | 6 | 15 | 11 | 10 | 7 |
| dropout | 0.01107 | 0.09115 | 0.07537 | 0.2959 | 0.4210 | 0.01652 | 0.06973 |
| activation fct. | Sigmoid | Sigmoid | ReLU | ReLU | ReLU | ReLU | ReLU |

"cosine *top k*" is here a percentage of the max number of pairs of instances in a batch.

## A.5 Estimation of the *top k* value

### A.5.1 Objective

Lowering the number of hyperparameters that must be tuned is an important task. By reducing the size of the hyperparameter space, the optimization is faster and the optimum is easier to find. In the context of NCD, it also decreases the chances of overfitting on the known classes (see Figure 4.5 of Section **??**). Therefore, in this section, we explore a way of automatically estimating the optimal value of *top k* in the pseudo-labeling process. The general idea is to apply the pseudo-labeling method to all the points of a mini-batch (both labeled and unlabeled) and to compute the accuracy on the labeled data.

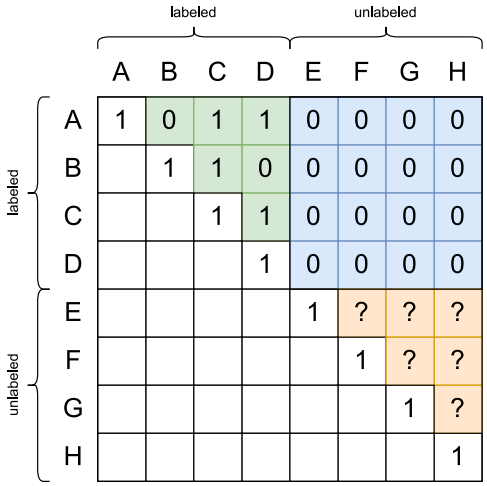



Figure A.4 – Ground truth pairwise pseudo-label matrix. Because it is symmetric, the lower triangle of the matrix can be ignored.

We illustrate this concept in Figure A.4, which represents the pairwise pseudo-label matrix for a batch of 8 data points where $\{A, B, C, D\}$ are labeled observations and $\{E, F, G, H\}$ are unlabeled observations. Since the ground truth labels of the known classes are available, we can easily define the true pairwise relationships of the green triangle. And we can then compare this ground-truth matrix with the result of our unsupervised pseudo-labeling process. Thus, by varying the *top k* hyperparameter in Equation 3.6 (see also Appendix A.1), we can find the value that minimizes the pseudo-labeling error on the labeled data.

This approach relies on the assumption that known and novel classes share some similarity in the distribution of their values. In other words, an optimal value of *top k* for

the known classes should also produce good results for the novel classes. Note that this approach can be used for all three pseudo-labeling methods (i.e. the $\lambda$ threshold, the *top* $k$ of Equation 3.6 or the $k$ in the RankStats method [17]).

To quickly find the optimal value of *top* $k$ for the labeled data, we explore different values using the **golden section search** (GSS). The GSS is a method to estimate the extremum (minimum or maximum) of a function in an interval. If the function is unimodal [1], it is guaranteed to find that extremum. As it incrementally narrows the range of the possible extremum, the GSS can be stopped either when the range is small enough or after a certain number of steps.

Proving that a function is unimodal is difficult, so we will only attempt to explain intuitively why the pseudo-labeling error is unimodal. In Figure A.5, we picture the pseudo-labeling error of Equation 3.6 according to the value of *top* $k$. When the value is too small, the pairwise pseudo-labeling will "miss" relations that should have been positive. And when it is too large there will be too many positive pairwise relations, some of which will link points between different classes. In practice, we find that when the latent space is reasonably well defined and the clusters are fairly separable, this unimodality holds true. And we find experimentally that after 10 iterations of the GSS, the precision is sufficient.

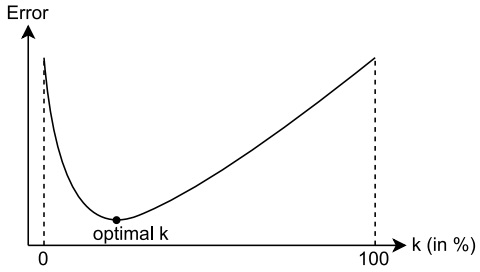



Figure A.5 – Example of a pseudo-labeling error curve in which we search the $k$ which minimizes the error with the golden section search.

### A.5.2 Experiments

In the experiments, we compare two estimation approaches. In the first, which was described above, only the *labeled-to-labeled* relations (the green of Figure A.4) are used

1. e.g. with $m$ the minimum, a unimodal function is monotonically decreasing for $x \leq m$ and monotonically increasing for $x \geq m$

to compute pseudo-labeling error. And in the second, we also use *labeled-to-unlabeled* relations (in blue). The *unlabeled-to-unlabeled* relations (in yellow) cannot be used as we have no information available regarding their labels.

These two estimation approaches are evaluated on a total of 4 pseudo-labeling methods, which define positive pairwise relations in different ways (see Figure 2.10 for a reminder):

1. **threshold** - cosine similarities above a certain threshold (see Eq. 2.9 of Sec. 2.5.2).
2. **top k per instance** - the $k$ largest cosine similarities of each instance (see Eq. 3.6 of Sec. 3.3.2).
3. **top k total** - the largest $k$ cosine similarities in the upper triangle of the pairwise cosine similarity matrix.
4. **top k per inst. agreeing** - if a pairwise relation is among the $k$ largest for both instances of the pair.

We compare the performance of TabularNCD when *top k* was optimized beforehand (called here the *baseline*) to its performance when *top k* was estimated during training. There are 2 estimation methods and 4 pseudo-labeling methods, so $1+2\times4 = 9$ approaches are compared in total. The numerical results for each approach and dataset in terms of clustering accuracy can be found in Table A.4. And in Figure A.6, the average rank of each approach is displayed on a critical diagram.

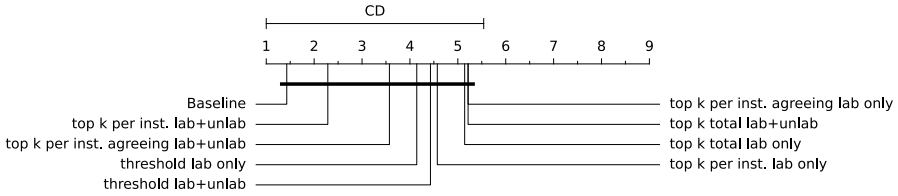



Figure A.6 – Critical diagram of the ACC of TabularNCD with different pseudo-labeling methods. The comparison is done using Nemenyi's test, which is conducted post-hoc on the results of a Friedman test.

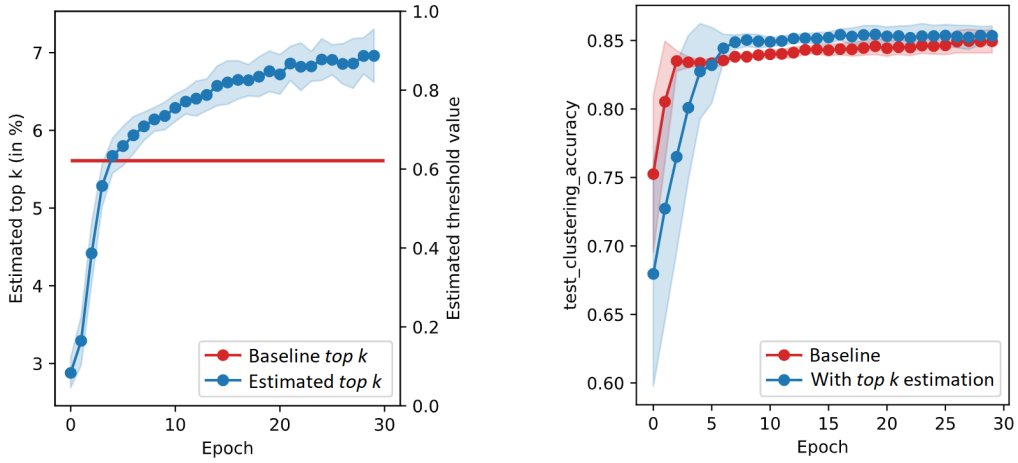
The first thing to note is that none of the estimation approaches reliably outperform the baseline. This could be explained by the fact that the hyperparameters were optimized for a fixed value of the "top k per instance" pseudo-labeling method. Thus, the comparison may be unfavorable to the other the other pseudo-labeling methods, as the hyperparameters are not independant of the pseudo-labeling method. Nevertheless, the "top k per instance" applied to both labeled and unlabeled data is almost on par with the

baseline, with an average ACC of 81.7% against 82.4% for the baseline. This suggests that if the hyperparameters were optimized using this estimation method, it could potentially outperform the baseline.

Table A.4 – Test clustering accuracy (×100) averaged over 5 executions. Best results are in bold.

| | | Forest | Human | Letter | Pendigits | Satimage | Census | MNIST |
|---|---|---|---|---|---|---|---|---|
| | baseline | **93.4±0.1** | **99.4±0.3** | **71.6±2.4** | 84.9±0.8 | **92.0±1.8** | 49.5±0.4 | 85.8±5.4 |
| lab+unlab | threshold | 91.0±3.1 | 88.5±13.0 | 60.7±5.4 | **88.1±1.7** | 76.1±3.4 | **52.2±4.5** | 21.1±0.0 |
| lab+unlab | *top k* total | 86.2±13.6 | 96.7±0.5 | 60.9±1.5 | 85.1±0.8 | 76.1±10.2 | 48.5±2.9 | 50.2±9.0 |
| lab+unlab | *top k* per instance | **93.2±0.4** | **98.8±0.2** | 65.3±3.3 | 85.3±0.7 | **92.9±0.8** | 45.1±1.1 | 91.2±4.9 |
| lab+unlab | *top k* per inst. agreeing | 89.9±7.0 | **98.1±0.5** | 37.4±8.8 | 84.5±0.9 | **92.5±1.8** | 48.9±1.1 | **93.6±0.2** |
| lab only | threshold | **92.7±0.3** | **99.5±0.1** | 69.3±5.9 | 76.2±8.3 | 77.0±1.7 | 48.7±0.9 | 21.1±0.0 |
| lab only | *top k* total | 88.5±5.0 | 74.8±2.7 | 61.0±4.6 | 82.5±4.7 | 83.2±7.9 | 49.4±2.0 | 25.1±7.9 |
| lab only | *top k* per instance | **93.1±0.3** | 97.9±0.3 | **69.3±2.5** | 81.6±1.1 | 74.4±0.3 | 44.2±0.6 | 87.9±3.3 |
| lab only | *top k* per inst. agreeing | **92.8±0.4** | 85.9±8.0 | 65.5±5.2 | 81.6±2.8 | 74.5±0.6 | 45.4±2.1 | 85.5±1.7 |

Finally, in Figure A.7, we plot the value of the estimated value of *top k* and the clustering accuracy during training of TabularNCD on the Pendigits dataset. We observe that the estimated value starts at a low value (around 3%), indicating that only the very close neighborhood of the points in the latent space belongs to the same class. However, as the training progresses and the latent space is refined by TabularNCD, the value rises since the neighborhood belonging to the same class as the observations expands.




(a) Estimated value of *top k*

(b) Clustering accuracy

Figure A.7 – Estimated threshold and clustering during the training of TabularNCD on the Pendigits dataset.

Appendix B

# Supplementary materials of Chapter 4

## B.1 Pseudocode of the methodologies

In this section, we provide the pseudocode of the NCD $k$-means and NCD Spectral Clustering methods proposed in Sections 4.3.2 and 4.3.3, respectively. With this addition, we hope to improve the clarity of our research and make it easier to understand the available open-source code.

**Algorithm 2** NCD $k$-means

**Require:** Labeled set of known classes $D^l = \{X^l, Y^l\}$, unlabeled set of novel classes $D^u = \{X^u\}$, number of clusters $k$.

**Initialize:** $L^l$ and $L^u$ empty lists of centroids for the known and novel classes

*Phase 1 - Initialization of the centroids*

*// Initialize the centroids of known classes using the ground truth labels:*
**for** each class $c$ in $D^l$ **do**
    Let $X_c^l = \{x_i^l \mid x_i^l \in X^l, y_i^l = c\}$ be the subset of points belonging to class $c$
    Calculate $\mu_c^l = \frac{1}{|X_c^l|} \sum_{x_i^l \in X_c^l} x_i^l$ the average point of class $c$
    Add $\mu_c^l$ to the list $L^l$ of centroids of the known classes
**end for**
*// Initialize the centroids of the novel classes in $L^u$ following k-means++:*
**while** $|L^u| < k$ **do**
    For each point $x_i^u$ in $D^u$, calculate the distance $d(x_i^u)$ between $x_i^u$ and its nearest centroid in $L^l \cup L^u$
    Choose the next centroid from the points in $D^u$ with probability proportional to $d(x_i^u)^2$
    Add the chosen centroid to $L^u$
**end while**

*Phase 2 - Follow the usual k-means algorithm*

Make the centroids of $L^u$ converge, using only the data of $D^u$
Partition the points in $D^u$ using only the centroids of the novel classes in $L^u$

**Algorithm 3** NCD Spectral Clustering

**Require:** Labeled set of known classes $D^l = \{X^l, Y^l\}$, unlabeled set of novel classes $D^u = \{X^u\}$, number of clusters $k$, number of hyperparameter optimization runs $n_{opt}$.

*Phase 1 - Determination of the optimal hyperparameters*

Initialize $ARI_{opt} = 0$ the best ARI obtained and $\{u_{opt} = 0, \sigma_{opt} = 0\}$ the optimal hyperparameters
**for** $n_{opt}$ random combinations of hyperparameters $\{u_{rand}, \sigma_{rand}\}$ **do**
    Compute the adjacency matrix $A$ of the points in $D^l \cup D^u$, where $A_{i,j} = \exp(-\|x_i - x_j\|_2^2/(2\sigma_{rand}^2))$
    Define the spectral embedding $U$ as the first $u_{rand}$ eigenvectors of the symmetric normalized Laplacian of $A$
    Partition all the points in $U$ using $k$-means into $k + C^l$ classes
    Calculate $ARI_{rand}$, the clustering ARI of this $k$-mean only on the points of $D^l$, using the ground truth labels $Y^l$
    **if** $ARI_{rand} > ARI_{opt}$ **then**
        $ARI_{opt} \leftarrow ARI_{rand}$, $u_{opt} \leftarrow u_{rand}$ and $\sigma_{opt} \leftarrow \sigma_{rand}$
    **end if**
**end for**

*Phase 2 - Clustering of the data of the novel classes*

Get $U_{opt}$ the spectral embedding of $D^l \cup D^u$ with the parameters $\{u_{opt}, \sigma_{opt}\}$
Partition the points of $D^u$ embedded in $U_{opt}$ into $k$ clusters with $k$-means

## B.2 Additional result metrics

In addition to the ACC results discussed in Section **??**, we present the NMI (Table B.1) and ARI (Table B.2) for the NCD methods when the number of novel classes $C^u$ is unknown and has to be estimated. Our results are consistent with those shown in Table 4.5: the PBN method largely outperforms its competitors in this realistic scenario. In particular, PBN achieves an average NMI higher than the baseline, NCD SC and TabularNCD by 23.5%, 8.6% and 13.5% respectively. Similarly, the ARI is 24.5%, 12.4% and 17.5% higher on average.

Table B.1 – Test NMI averaged over 10 runs. With $C^u$ estimated with the Silhouette coefficient. Best results are in bold.

| Dataset | Baseline | NCD SC | TabularNCD | PBN |
|---|---|---|---|---|
| Human | 56.1±8.9 | 45.8±3.0 | **75.2±0.0** | **75.2±0.0** |
| Letter | 54.4±3.3 | 52.6±1.0 | 37.2±3.2 | **59.2±2.4** |
| Pendigits | 41.0±6.7 | **79.5±1.4** | 48.0±6.5 | 73.4±2.5 |
| Census | 59.7±0.6 | 33.5±1.0 | 42.6±1.5 | **60.6±0.5** |
| m feat | 50.0±3.9 | 71.9±2.9 | 44.8±2.1 | **79.1±3.3** |
| Optdigits | 28.5±6.0 | 77.3±2.0 | **93.1±2.0** | 84.9±2.0 |
| CNAE-9 | 23.0±1.5 | **56.5±2.0** | 42.0±1.3 | 45.2±13.4 |
| Average | 44.7 | 59.6 | 54.7 | **68.2** |

Table B.2 – Test ARI averaged over 10 runs. With $C^u$ estimated with the Silhouette coefficient. Best results are in bold.

| Dataset | Baseline | NCD SC | TabularNCD | PBN |
|---|---|---|---|---|
| Human | 48.7±4.0 | 20.3±3.0 | **61.4±0.0** | **61.4±0.0** |
| Letter | 44.3±4.1 | 29.5±2.2 | 23.4±5.6 | **48.9±3.1** |
| Pendigits | 29.8±5.7 | **73.7±1.7** | 37.6±6.5 | 65.3±3.4 |
| Census | **42.6±3.7** | 23.0±3.8 | 26.7±0.8 | 35.5±0.2 |
| m feat | 40.9±5.3 | 64.1±4.5 | 21.5±0.2 | **79.1±4.2** |
| Optdigits | 16.9±6.5 | 75.6±3.6 | **94.1±2.7** | 84.4±4.4 |
| CNAE-9 | 11.6±1.9 | **33.2±4.0** | 18.8±0.4 | 31.5±1.7 |
| Average | 33.5 | 45.6 | 40.5 | **58.0** |

## B.3 Hyperparameters details

The Table B.3 shows the hyperparameters found by the full procedure described in Section 4.6. The type of distributions and value spaces explored for each hyperparameter are given in the "Range" and "Distribution" columns. The variable $d$ in the range of the *latent dim* refers to the dimension of the points in the datasets.

Table B.3 – Hyperparameters of the methods found when $C^u$ is estimated.

| | Parameter | Range | Distribution | Human | Letter | Pendigits | Census | m features | Optdigits | CNAE-9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Base-line | latent dim | [5, $d$] | int uniform | 560 | 9 | 9 | 30 | 34 | 42 | 569 |
| | lr | [0.0001, 0.1] | log uniform | 0.000154 | 0.001333 | 0.005517 | 0.002021 | 0.000118 | 0.003330 | 0.000167 |
| | dropout | [0.0, 0.6] | uniform | 0.168044 | 0.140095 | 0.052505 | 0.467836 | 0.564855 | 0.075180 | 0.257765 |
| NCD SC | $s_{min}$ | ]0.0, 1.0[ | uniform | 0.29396 | 0.98137 | 0.86147 | 0.63534 | 0.70005 | 0.20412 | 0.83132 |
| | $u$ | [1, 200] | int uniform | 144 | 14 | 18 | 53 | 16 | 26 | 14 |
| TabularNCD | $k$ neighbors | [2, 100] | int uniform | 70 | 29 | 73 | 45 | 15 | 48 | 7 |
| | latent dim | [5, $d$] | int uniform | 372 | 14 | 13 | 56 | 98 | 62 | 791 |
| | lr | [0.0001, 0.1] | log uniform | 0.00961 | 0.00036 | 0.00464 | 0.00124 | 0.00840 | 0.00755 | 0.00129 |
| | dropout | [0.0, 0.6] | uniform | 0.15436 | 0.06672 | 0.27823 | 0.27489 | 0.01384 | 0.09158 | 0.08220 |
| | *top k* | [0.0, 1.0] | uniform | 0.61097 | 0.16517 | 0.46739 | 0.93956 | 0.59044 | 0.99586 | 0.94210 |
| | $w_1$ | [0.0, 1.0] | uniform | 0.04745 | 0.41367 | 0.13992 | 0.31247 | 0.30193 | 0.78532 | 0.19631 |
| | $w_2$ | [0.0, 1.0] | uniform | 0.70265 | 0.97095 | 0.60544 | 0.88265 | 0.83427 | 0.93356 | 0.64128 |
| PBN | latent dim | [5, $d$] | int uniform | 504 | 22 | 12 | 13 | 172 | 53 | 579 |
| | lr | [0.0001, 0.1] | log uniform | 0.00010 | 0.00058 | 0.00107 | 0.00211 | 0.00467 | 0.00036 | 0.00440 |
| | dropout | [0.0, 0.6] | uniform | 0.50984 | 0.02745 | 0.01126 | 0.16777 | 0.23934 | 0.06606 | 0.08131 |
| | $w$ | [0.0, 1.0] | uniform | 0.46714 | 0.72241 | 0.10671 | 0.96086 | 0.67034 | 0.18917 | 0.69815 |

## B.4 PBN coupled with Spectral Clustering

In Section 4.3.3, we proposed the Projection Based NCD (PBN) method where a latent representation is first trained with a reconstruction loss on all the points, and a classification loss on the labeled data. After training of the representation, the simple $k$-means algorithm is used to cluster the unlabeled data of the novel classes. It was chosen mainly for its fast execution time, as the idea behind PBN is to obtain a representation where the novel classes are already well separated, so a sophisticated clustering algorithm should not be needed. But naturally, any other unsupervised clustering method can be employed and might improve the performance. Therefore, in this section, we investigate the performance of PBN when $k$-means is replaced with Spectral Clustering (SC).

In these experiments, the number of novel classes $C^u$ is considered to be known in advance. In Table B.4, three variations of the PBN method are compared:

— **KM PBN:** The model used in the article, where $k$-means is used in the latent space.

— **SC PBN default:** $k$-means is directly replaced by SC with default hyperparameters. The similarity $s_{min}$ is set to 0.6, and the number of components $u$ of the spectral embedding is set to the number of novel classes $C^u$, as is usually done in the literature on Spectral Clustering.

— **SC PBN full opt:** Here, SC is also used for clustering, and its hyperparameters $s_{min}$ and $u$ are optimized together with PBN's hyperparameters. So for each datasets, 6 hyperparameters are optimized (4 from PBN and 2 from SC).

Table B.4 – Test accuracy of the 3 variations of PBN averaged over 10 runs. Best results are in bold.

| Dataset | KM PBN | SC PBN default | SC PBN full opt |
|---|---|---|---|
| Human | 76.7±1.8 | **80.2±0.7** | 76.6±6.1 |
| Letter | 62.4±2.0 | **64.4±2.8** | 50.9±2.7 |
| Pendigits | **82.8±0.6** | 82.0±2.2 | 79.3±4.0 |
| Census | **62.4±0.9** | 61.1±2.0 | 53.0±2.4 |
| m feat | 91.7±0.8 | **92.6±1.6** | 89.7±1.5 |
| Optdigits | 92.6±2.3 | **94.8±2.1** | 94.6±3.6 |
| CNAE-9 | 72.6±4.6 | **73.5±6.1** | 68.0±4.2 |
| Average | 77.3 | **78.4** | 73.2 |

In Table B.4, it can be seen that using SC with its default hyperparameters results in a clustering accuracy that is on average 1.1% higher than PBN coupled with $k$-means. But when the hyperparameters of SC are also optimized, the average accuracy drops by 4.1% instead. This shows that, as expressed throughout this article, having too many hyperparameters in the context of the NCD problem makes it difficult to effectively transfer the results on the known classes to the novel classes, and the hyperparameters tend to overfit the known classes.

These results suggest that the best approach is to apply SC with its default hyperparameters to cluster the projection of the novel data. However, it must be noted that the number of novel classes is used to define the number of components of the spectral embedding. Thus, in the case where $C^u$ is not considered to be known in advance and has to be estimated, the performance may be degraded.

## B.5 Cluster Validity Indices results details

An estimate of the number of clusters in the 7 datasets considered in this paper can be found in Table B.5. Among the 6 CVIs reported here, the Silhouette coefficient performed the best. Furthermore, compared to the original feature space, its average estimation error significantly decreased in the latent space, validating our approach. For some datasets, the Davies-Bouldin index continued to decrease and the Dunn index continued to increase as the number of clusters increased, resulting in very large overestimations. Note that the estimates of the number of novel classes in Table B.5 are not needed in the experiments of Section **??**, since Algorithm 1 directly incorporates such estimates in the training procedure. This table has only helped us to identify the most appropriate CVI for our problem. The only exception is the TabularNCD method, which requires an a priori estimation of the number of novel classes in the original feature space.

Table B.5 – An estimation of the number of novel classes with some CVIs in the latent space of PBN.

| Dataset | Human | Letter | Pendigits | Census | m feat | Optdigits | CNAE-9 |
|---|---|---|---|---|---|---|---|
| Ground truth | 3 | 7 | 5 | 6 | 5 | 5 | 5 |
| PBN latent space | | | | | | | |
| Silhouette | 2 | 8 | 5 | 3 | 5 | 5 | 5 |
| CH | 2 | 3 | 5 | 4 | 2 | 2 | 5 |
| Dunn | 2 | 3 | 98 | 3 | 2 | 95 | 2 |
| KM ACC | 1 | 2 | 3 | 3 | 5 | 6 | 1 |
| Davies-B. | 2 | 63 | 6 | 3 | 99 | 4 | 96 |
| Elbow | 9 | 14 | 10 | 7 | 16 | 13 | 9 |
| Original feature space | | | | | | | |
| Silhouette | 2 | 45 | 5 | 3 | 2 | 9 | 2 |

## B.6 NCD k-means centroids convergence

In this appendix, we aim to determine how to achieve the best performance with NCD $k$-means. Specifically, after the centroid initialization described in Section 4.3.2, we investigate: (1) whether it is more effective to update the centroids of both known and novel classes, or only the centroids of novel classes; (2) whether the centroids need to be updated using data from both known and novel classes, or only using data from novel classes. The results are presented in Table B.6 and show that for 5 out of 7 datasets,

the best results are obtained when only the centroids of the novel classes are updated on the unlabeled data. Updating the centroids of the known classes always leads to worse performance, as the class labels are not used in this process. Thus, the centroids of the known classes run the risk of capturing data from the novel classes (and vice versa).

Table B.6 – ACC of NCD $k$-means averaged over 10 runs. Best results are in bold.

| Dataset | Converging the centroids... | On unlabeled data only | On labeled and unlabeled data |
|---|---|---|---|
| Human | novel only | 75.9±0.0 | **77.4±0.0** |
| | known and novel | - | 75.1±0.5 |
| Letter | novel only | **51.9±2.3** | 39.5±1.9 |
| | known and novel | - | 42.3±2.6 |
| Pendigits | novel only | **81.7±0.0** | 72.7±0.9 |
| | known and novel | - | 75.3±4.0 |
| Census | novel only | **50.4±1.1** | 50.4±4.8 |
| | known and novel | - | 44.6±8.3 |
| m feat | novel only | **89.7±0.4** | 69.1±0.2 |
| | known and novel | - | 84.1±7.0 |
| Optdigits | novel only | **94.2±0.0** | 70.8±7.8 |
| | known and novel | - | 74.0±14.7 |
| CNAE-9 | novel only | 61.2±4.5 | 48.3±8.5 |
| | known and novel | - | **68.1±7.5** |

# BIBLIOGRAPHY


[1] Y. Yang, D. Xu, F. Nie, S. Yan, and Y. Zhuang, "Image clustering using local discriminant models and global integration," *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2761–2773, 2010.

[2] H. W. Kuhn and B. Yaw, "The hungarian method for the assignment problem," *Naval Res. Logist. Quart*, pp. 83–97, 1955.

[3] "Fixed broadband and superfast broadband market in france," 2023. `https://en.arcep.fr/news/press-releases/view/n/fixed-broadband-and-superfast-broadband-market-071223.html`, Accessed: 2024-02-02.

[4] "Cisco annual internet report," 2023. `https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html`, Accessed: 2024-02-02.

[5] A. Lawrence and L. Simon, "Annual outage analysis 2023," tech. rep., Uptime Institute Intelligence, 2023.

[6] S. Cherrared, S. Imadali, E. Fabre, G. Gössler, and I. G. B. Yahia, "A survey of fault management in network virtualization environments: Challenges and solutions," *IEEE TNSM*, vol. 16, no. 4, pp. 1537–1551, 2019.

[7] J. Zaytoon and S. Lafortune, "Overview of fault diagnosis methods for discrete event systems," *Annual Reviews in Control*, vol. 37, no. 2, pp. 308–320, 2013.

[8] R. Barco, V. Wille, L. Díez, and M. Toril, "Learning of model parameters for fault diagnosis in wireless networks," *Wireless Networks*, vol. 16, pp. 255–271, 2010.

[9] A. Abraham, "Rule-based expert systems," *Handbook of measuring system design*, 2005.

[10] A. Gupta and P. Prabhat, "Novel approaches in network fault management," *International Journal of Next-Generation Computing*, vol. 8, no. 2, 2017.

[11] L. Bonniot, *Computer network modeling and root cause analysis with statistical learning*. Thesis, Université Rennes 1, June 2021.

[12] B. Cai, L. Huang, and M. Xie, "Bayesian networks in fault diagnosis," *IEEE Transactions on industrial informatics*, vol. 13, no. 5, pp. 2227–2240, 2017.

[13] M. Kordestani, M. Saif, M. E. Orchard, R. Razavi-Far, and K. Khorasani, "Failure prognosis and applications—a survey of recent literature," *IEEE transactions on reliability*, vol. 70, no. 2, pp. 728–748, 2019.

[14] L. Feng, L. Xiang, and W. Xiu-qing, "A survey of intelligent network fault diagnosis technology," in *Chinese Control and Decision Conference*, pp. 4874–4879, 2013.

[15] S. R. Tembo Mouafo, *Applications de l'intelligence artificielle à la détection et l'isolation de pannes multiples dans un réseau de télécommunications*. Thesis, Ecole nationale supérieure Mines-Télécom Atlantique, Jan. 2017.

[16] A. Echraibi, *Probabilistic graphical models : theory and applications to network diagnosis*. Thesis, Ecole nationale supérieure Mines-Télécom Atlantique, Dec. 2021.

[17] K. Han, S.-A. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, "Autonovel: Automatically discovering and learning novel visual categories," *TPAMI*, 2021.

[18] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: A survey," *IEEE transactions on neural networks and learning systems*, 2022.

[19] H. Chi, F. Liu, W. Yang, L. Lan, T. Liu, B. Han, G. Niu, M. Zhou, and M. Sugiyama, "Meta discovery: Learning to discover novel classes given very limited data," in *International Conference on Learning Representations*, 2022.

[20] C. Troisemaine, V. Lemaire, S. Gosselin, A. Reiffers-Masson, J. Flocon-Cholet, and S. Vaton, "Novel class discovery: an introduction and key concepts," *ArXiv*, 2023.

[21] C. Troisemaine, J. Flocon-Cholet, S. Gosselin, S. Vaton, A. Reiffers-Masson, and V. Lemaire, "A method for discovering novel classes in tabular data," in *ICKG*, pp. 265–274, 2022.

[22] C. Troisemaine, J. Flocon-Cholet, S. Gosselin, S. Vaton, A. Reiffers-Masson, and V. Lemaire, "Découvrir de nouvelles classes dans des données tabulaires," in *Extraction et Gestion des Connaissances: Actes de la conférence (EGC)*, vol. 39, 2023.

[23] C. Troisemaine, A. Reiffers-Masson, S. Gosselin, V. Lemaire, and S. Vaton, "A practical approach to novel class discovery in tabular data," *Data Mining and Knowledge Discovery*, May 2024.

[24] C. Troisemaine, J. Flocon-Cholet, S. Gosselin, A. Reiffers-Masson, S. Vaton, and V. Lemaire, "An interactive interface for novel class discovery in tabular data," in *Machine Learning and Knowledge Discovery in Databases: Applied Data Science and Demo Track*, vol. 14175, pp. 295–299, 2023.

[25] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," *International Journal of Computer Vision*, pp. 1–28, 2024.

[26] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *CVPR*, 2015.

[27] P. Nodet, V. Lemaire, A. Bondu, A. Cornuéjols, and A. Ouorou, "From weakly supervised learning to biquality learning: an introduction," in *IJCNN*, 2021.

[28] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

[29] Y.-C. Hsu, Z. Lv, and Z. Kira, "Learning to cluster in order to transfer across domains and tasks," in *International Conference on Learning Representations*, 2018.

[30] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Trans. Intell. Syst. Technol.*, 2019.

[31] M. Abavisani and V. M. Patel, "Deep multimodal subspace clustering networks," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1601–1614, 2018.

[32] K. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *ICCV*, pp. 5747–5756, 2017.

[33] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *CVPR*, pp. 5147–5156, 2016.

[34] Y. Li, M. Yang, D. Peng, T. Li, J. Huang, and X. Peng, "Twin contrastive learning for online clustering," *International Journal of Computer Vision*, vol. 130, 2022.

[35] F. Ntelemis, Y. Jin, and S. A. Thomas, "Information maximization clustering via multi-view self-labelling," *Knowledge-Based Systems*, vol. 250, p. 109042, 2022.

[36] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Generalized category discovery," in *CVPR*, pp. 7492–7501, 2022.

[37] K. Han, A. Vedaldi, and A. Zisserman, "Learning to discover novel visual categories via deep transfer clustering," in *ICCV*, 2019.

[38] Z. Wang, B. Salehi, A. Gritsenko, K. Chowdhury, S. Ioannidis, and J. Dy, "Open-world class discovery with kernel networks," in *ICDM*, pp. 631–640, 2020.

[39] Z. Zhong, E. Fini, S. Roy, Z. Luo, E. Ricci, and N. Sebe, "Neighborhood contrastive learning for novel class discovery," in *IEEE/CVF CVPR*, 2021.

[40] Z. Li, J. Otholt, B. Dai, D. Hu, C. Meinel, and H. Yang, "A closer look at novel class discovery from the labeled set," in *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

[41] K. Han, S.-A. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, "Automatically discovering and learning new visual categories with ranking statistics," in *International Conference on Learning Representations (ICLR)*, 2020.

[42] Y.-C. Hsu, Z. Lv, J. Schlosser, P. Odom, and Z. Kira, "Multi-class classification without multi-class labels," in *ICLR*, 2019.

[43] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *ICML*, vol. 48, pp. 478–487, 2016.

[44] Y. Qing, Y. Zeng, Q. Cao, and G.-B. Huang, "End-to-end novel visual categories learning via auxiliary self-supervision," *Neural Networks*, vol. 139, pp. 24–32, 2021.

[45] Z. Zhong, L. Zhu, Z. Luo, S. Li, Y. Yang, and N. Sebe, "Openmix: Reviving known knowledge for discovering novel visual categories in an open world," in *CVPR*, 2021.

[46] X. Jia, K. Han, Y. Zhu, and B. Green, "Joint representation learning and novel category discovery on single-and multi-modal data," in *ICCV*, pp. 610–619, 2021.

[47] J. Yagnik, D. Strelow, D. A. Ross, and R.-s. Lin, "The power of comparative reasoning," in *ICCV*, pp. 2431–2438, 2011.

[48] B. Zhao and K. Han, "Novel visual category discovery with dual ranking statistics and mutual knowledge distillation," in *NeurIPS*, 2021.

[49] K. Joseph, S. Paul, G. Aggarwal, S. Biswas, P. Rai, K. Han, and V. N. Balasubramanian, "Spacing loss for discovering novel categories," in *CVPR*, 2022.

[50] Y. Liu and T. Tuytelaars, "Residual tuning: Toward novel category discovery without labels," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[51] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *ICLR*, 2018.

[52] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *ICLR*, 2018.

[53] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *NeurIPS*, 2020.

[54] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *AISTATS*, pp. 297–304, 2010.

[55] J. Yoon, Y. Zhang, J. Jordon, and M. van der Schaar, "Vime: Extending the success of self- and semi-supervised learning to tabular domain," in *NeurIPS*, 2020.

[56] Q. Yu, D. Ikami, G. Irie, and K. Aizawa, "Self-labeling framework for novel category discovery over domains," in *Conference on Artificial Intelligence*, vol. 36, 2022.

[57] Y. Fei, Z. Zhao, S. Yang, and B. Zhao, "Xcon: Learning with experts for fine-grained category discovery," in *British Machine Vision Conference (BMVC)*, 2022.

[58] J. Zheng, W. Li, J. Hong, L. Petersson, and N. Barnes, "Towards open-set object detection and discovery," in *CVPR*, pp. 3961–3970, 2022.

[59] N. Chawla, K. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[60] M. Yang, Y. Zhu, J. Yu, A. Wu, and C. Deng, "Divide and conquer: Compositional experts for generalized novel class discovery," in *CVPR*, pp. 14268–14277, 2022.

[61] E. Fini, E. Sangineto, S. Lathuilière, Z. Zhong, M. Nabi, and E. Ricci, "A unified objective for novel class discovery," in *ICCV*, pp. 9284–9292, 2021.

[62] J. Zhuang, Z. Chen, P. Wei, G. Li, and L. Lin, "Discovering implicit classes achieves open set domain adaptation," in *ICME*, pp. 01–06, 2022.

[63] M. N. Rizve, N. Kardan, S. Khan, F. Shahbaz Khan, and M. Shah, "Openldn: Learning to discover novel classes for open-world semi-supervised learning," in *ECCV*, 2022.

[64] K. Joseph, S. Paul, G. Aggarwal, S. Biswas, P. Rai, K. Han, and V. N. Balasubramanian, "Novel class discovery without forgetting," in *ECCV*, 2022.

[65] S. Roy, M. Liu, Z. Zhong, N. Sebe, and E. Ricci, "Class-incremental novel class discovery," in *European Conference on Computer Vision*, pp. 317–333, 2022.

[66] Y. Zhao, Z. Zhong, N. Sebe, and G. H. Lee, "Novel class discovery in semantic segmentation," in *CVPR*, pp. 4340–4349, 2022.

[67] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European conference on computer vision*, pp. 649–666, 2016.

[68] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[69] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *ICCV*, 2021.

[70] D. Bahri, H. Jiang, Y. Tay, and D. Metzler, "Scarf: Self-supervised contrastive learning using random feature corruption," in *ICLR*, 2022.

[71] T. Ucar, E. Hajiramezanali, and L. Edwards, "Subtab: Subsetting features of tabular data for self-supervised representation learning," *NeurIPS*, vol. 34, 2021.

[72] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *IJCNN*, 2020.

[73] C.-C. Hsu and C.-W. Lin, "Cnn-based joint clustering and representation learning with feature drift compensation for large-scale image data," *IEEE Transactions on Multimedia*, vol. 20, no. 2, pp. 421–429, 2017.

[74] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *JMLR*, pp. 113–141, 2000.

[75] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, vol. 2, pp. 1735–1742, 2006.

[76] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[77] Y. Sun and Y. Li, "Opencon: Open-world contrastive learning," in *TMLR*, 2023.

[78] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, 2018.

[79] J. Wang, Z. Ma, F. Nie, and X. Li, "Progressive self-supervised clustering with novel category discovery," *IEEE Transactions on Cybernetics*, 2021.

[80] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. The MIT Press, 2006.

[81] D. Zhang, Z.-H. Zhou, and S. Chen, "Semi-supervised dimensionality reduction," in *2007 SIAM International Conference on Data Mining*, pp. 629–634, 2007.

[82] Z.-H. Zhou, M. Li, *et al.*, "Semi-supervised regression with co-training.," in *IJCAI*, vol. 5, pp. 908–913, 2005.

[83] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *International Conference on Machine Learning (ICML)*, 2002.

[84] J. Callut, K. Françoisse, M. Saerens, and P. Dupont, "Semi-supervised classification from discriminative random walks," in *ECML PKDD*, pp. 162–177, 2008.

[85] K. L. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained k-means clustering with background knowledge," in *ICML*, 2001.

[86] S. Basu, A. Banerjee, and R. Mooney, "Active semi-supervision for pairwise constrained clustering," *SIAM International Conference on Data Mining*, 2004.

[87] S. Mehrkanoon, C. Alzate, R. Mall, R. Langone, and J. A. Suykens, "Multiclass semisupervised learning based upon kernel spectral clustering," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 4, pp. 720–733, 2014.

[88] V. Lemaire, O. Alaoui Ismaili, and A. Cornuéjols, "An initialization scheme for supervized k-means," in *International Joint Conference on Neural Networks*, 2015.

[89] Y. Chen, X. Zhu, W. Li, and S. Gong, "Semi-supervised learning under class distribution mismatch," in *AAAI*, 2020.

[90] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," *NeurIPS*, vol. 31, 2018.

[91] L.-Z. Guo, Z.-Y. Zhang, Y. Jiang, Y.-F. Li, and Z.-H. Zhou, "Safe deep semi-supervised learning for unseen-class unlabeled data," in *ICML*, 2020.

[92] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, pp. 248–255, 2009.

[93] J. Tao and X. Fang, "Toward multi-label sentiment analysis: a transfer learning based approach," *Journal of Big Data*, vol. 7, no. 1, pp. 1–26, 2020.

[94] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang, "Heterogeneous transfer learning for image classification," in *AAAI*, 2011.

[95] S. J. Pan, Q. Yang, Y. Zhang, and W. Dai, *Transfer Learning in Activity Recognition*, p. 307–323. Cambridge University Press, 2020.

[96] X. Shi, Q. Liu, W. Fan, P. S. Yu, and R. Zhu, "Transfer learning on heterogenous feature spaces via spectral transformation," in *ICDM*, pp. 1049–1054, 2010.

[97] C. Wang and S. Mahadevan, "Heterogeneous domain adaptation using manifold alignment," in *IJCAI*, p. 1541–1546, 2011.

[98] W. Dai, Y. Chen, G.-r. Xue, Q. Yang, and Y. Yu, "Translated learning: Transfer learning across different feature spaces," in *NeurIPS*, vol. 21, 2009.

[99] P. Prettenhofer and B. Stein, "Cross-language text classification using structural correspondence learning," in *48th Annual Meeting of the Association for Computational Linguistics*, pp. 1118–1127, 2010.

[100] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller, "A unifying review of deep and shallow anomaly detection," *IEEE*, vol. 109, no. 5, pp. 756–795, 2021.

[101] M. Markou and S. Singh, "Novelty detection: a review—part 1: statistical approaches," *Signal processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[102] L. Shu, H. Xu, and B. Liu, "Unseen class discovery in open-world classification," *ArXiv*, vol. abs/1801.05609, 2018.

[103] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *TPAMI*, vol. 35, no. 7, pp. 1757–1772, 2013.

[104] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *ICMLA*, pp. 195–200, 2016.

[105] T. Xiao, C. Zhang, and H. Zha, "Learning to detect anomalies in surveillance video," *IEEE Signal Processing Letters*, vol. 22, no. 9, pp. 1477–1481, 2015.

[106] J. Van den Broeck, S. Argeseanu Cunningham, R. Eeckels, and K. Herbst, "Data cleaning: detecting, diagnosing, and editing data abnormalities," *PLoS medicine*, vol. 2, no. 10, p. e267, 2005.

[107] T. T. Dang, H. Y. Ngan, and W. Liu, "Distance-based k-nearest neighbors outlier detection method in large-scale traffic data," in *DSP*, pp. 507–510, 2015.

[108] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *SIGMOD*, pp. 93–104, 2000.

[109] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996.

[110] Y. Sun, Z. Shi, Y. Liang, and Y. Li, "When and how does known class help discover unknown ones? provable understanding through spectral analysis," in *ICML*, vol. 202, pp. 33014–33043, 2023.

[111] Y. Zhu, T. Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y. Evrard, J. Doroshow, and R. Stevens, "Converting tabular data into images for deep learning with convolutional neural networks," *Scientific Reports*, vol. 11, 2021.

[112] E. Fitkov-Norris, S. Vahid, and C. Hand, "Evaluating the impact of categorical data encoding and scaling on neural network classification performance: The case of repeat consumption of identical cultural goods," in *CCIS*, pp. 343–352, 2012.

[113] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *TKDE*, vol. 31, pp. 2346–2363, 2019.

[114] S. J. Pan and Q. Yang, "A survey on transfer learning," *TKDE*, 2010.

[115] Q. Qin, W. Hu, and B. Liu, "Text classification with novelty detection," *ArXiv*, vol. abs/2009.11119, 2020.

[116] H. Xu, B. Liu, L. Shu, and P. Yu, "Open-world learning and application to product classification," in *The World Wide Web Conference*, p. 3413–3419, 2019.

[117] X. Guo, A. Alipour-Fanid, L. Wu, H. Purohit, X. Chen, K. Zeng, and L. Zhao, "Multi-stage deep classifier cascades for open world recognition," *CIKM*, 2019.

[118] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.

[119] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *NeurIPS*, 2017.

[120] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," in *NeurIPS*, vol. 31, 2018.

[121] K. Cao, M. Brbic, and J. Leskovec, "Open-world semi-supervised learning," in *International Conference on Learning Representations*, 2022.

[122] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, 1997.

[123] L. Pascal, P. Michiardi, X. Bost, B. Huet, and M. A. Zuluaga, "Optimization strategies in multi-task learning: Averaged or independent losses?," *arXiv*, vol. abs/2109.11678, 2021.

[124] J. A. Blackard and D. J. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Computers and Electronics in Agriculture*, vol. 24, pp. 131–151, 1999.

[125] P. W. Frey and D. J. Slate, "Letter recognition using holland-style adaptive classifiers," *Machine Learning*, vol. 6, pp. 161–182, 2005.

[126] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.

[127] D. Dua and C. Graff, "Uci machine learning repository," 2017.

[128] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[129] U. von Luxburg, "A tutorial on spectral clustering.," *Stat. Comput.*, 2007.

[130] X. Wang and I. Davidson, "Active spectral clustering," in *ICDM*, pp. 561–568, 2010.

[131] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019.

[132] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, vol. 32, 2019.

[133] L. Zhang, L. Qi, X. Yang, H. Qiao, M.-H. Yang, and Z. Liu, "Automatically discovering novel visual categories with adaptive prototype learning," *TPAMI*, 2024.

[134] D. Arthur and S. Vassilvitskii, "K-means++ the advantages of careful seeding," in *ACM-SIAM SODA*, pp. 1027–1035, 2007.

[135] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NeurIPS*, pp. 849–856, 2001.

[136] A. A. Khan and S. K. Mohanty, "A fast spectral clustering technique using mst based proximity graph for diversified datasets," *Information Sciences*, 2022.

[137] W. Stuetzle, "Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample," *Journal of classification*, vol. 20, no. 1, pp. 25–47, 2003.

[138] L. Le, A. Patterson, and M. White, "Supervised autoencoders: Improving generalization performance with unsupervised regularizers," *NeurIPS*, vol. 31, 2018.

[139] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, "An extensive comparative study of cluster validity indices," *Pattern recognition*, 2013.

[140] M. Yang, L. Wang, C. Deng, and H. Zhang, "Bootstrap your own prior: Towards distribution-agnostic novel class discovery," in *CVPR*, pp. 3459–3468, 2023.

[141] U. Von Luxburg, R. C. Williamson, and I. Guyon, "Clustering: Science or art?," in *ICML workshop on unsupervised and transfer learning*, pp. 65–79, 2012.

[142] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," in *ICDCS Workshops*, 2011.

[143] K. Arvai, "kneed," Apr. 2023.

[144] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, 2022.

[145] T. Soukup and I. Davidson, *Visual data mining: Techniques and tools for data visualization and mining*. John Wiley & Sons, Inc., 2002.

[146] P. Chanclou, S. Gosselin, J. Palacios, V. Alvarez, and E. Zouganeli, "Overview of the optical broadband access evolution: a joint article by operators in the ist network of excellence e-photon/one," *IEEE Communications Magazine*, 2006.

[147] A. Gladisch, C. Lange, and R. Leppla, "Power efficiency of optical versus electronic access networks," in *European Conference on Optical Communication*, 2008.

[148] J. Baliga, R. Ayre, K. Hinton, and R. S. Tucker, "Energy consumption in wired and wireless access networks," *IEEE Communications Magazine*, pp. 70–77, 2011.

[149] ITU-T, "Gigabit-capable passive optical networks (gpon): General characteristics," standard, International Telecommunication Union, Geneva, CH, Mar. 2008.

[150] ITU-T, "10-gigabit-capable symmetric passive optical network (xgs-pon)," standard, International Telecommunication Union, Geneva, CH, Feb. 2023.

[151] I. Cale, A. Salihovic, and M. Ivekovic, "Gigabit passive optical network - gpon," in *International conference on information technology interfaces*, pp. 679–684, 2007.

[152] U. Khurana, H. Samulowitz, and D. Turaga, "Feature engineering for predictive modeling using reinforcement learning," in *AAAI*, vol. 32, 2018.

[153] F. Horn, R. Pack, and M. Rieger, "The autofeat python library for automated feature engineering and selection," in *ECML PKDD*, pp. 111–120, 2019.

[154] R. Bellman, R. Bellman, and R. Corporation, *Dynamic Programming*. Rand Corporation research study, American Association for the Advancement of Science, 1957.

[155] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *ICDT*, pp. 420–434, 2001.

[156] S. Solorio-Fernández, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "A review of unsupervised feature selection methods," *Artificial Intelligence Review*, 2020.

[157] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[158] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[159] M. Gevrey, I. Dimopoulos, and S. Lek, "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecological modelling*, vol. 160, no. 3, pp. 249–264, 2003.

[160] R. Varshavsky, A. Gottlieb, M. Linial, and D. Horn, "Novel unsupervised feature filtering of biological data," *Bioinformatics*, vol. 22, no. 14, pp. e507–e513, 2006.

[161] V. Roth and T. Lange, "Feature selection in clustering problems," *NeurIPS*, 2003.

[162] Y. Li, B.-L. Lu, and Z.-F. Wu, "A hybrid method of unsupervised feature selection based on ranking," in *ICPR*, vol. 2, pp. 687–690, 2006.

[163] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," *NeurIPS*, 2005.

[164] D. Yan, L. Huang, and M. I. Jordan, "Fast approximate spectral clustering," in *International conference on Knowledge discovery and data mining*, 2009.

[165] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the nystrom method," *TPAMI*, vol. 26, no. 2, pp. 214–225, 2004.

[166] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, 1987.

[167] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, pp. 224–227, 1979.

[168] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," *IJPRAI*, vol. 25, no. 03, pp. 337–372, 2011.

[169] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Á. Fernández-Leal, "Human-in-the-loop machine learning: a state of the art," *Artificial Intelligence Review*, vol. 56, no. 4, pp. 3005–3054, 2023.

[170] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," *Advances in neural information processing systems*, vol. 30, 2017.

[171] N. Monath, M. Zaheer, D. Silva, A. McCallum, and A. Ahmed, "Gradient-based hierarchical clustering using continuous representations of trees in hyperbolic space," in *25th ACM SIGKDD International Conference*, pp. 714–722, 2019.

[172] F.-Y. Lin, B. Bai, K. Bai, Y. Ren, P. Zhao, and Z. Xu, "Contrastive multi-view hyperbolic hierarchical clustering," in *IJCAI*, 2022.

COLLEGES SCIENCES
BRETAGNE POUR L'INGENIEUR
LOIRE ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

**Titre :** Découverte de Nouvelles Classes dans les Données Tabulaires : une Application au Diagnostic de Pannes des Réseaux

**Mot clés :** Découverte de nouvelles classes, Partitionnement, Diagnostic, Données tabulaires

**Résumé :** Cette thèse porte sur la découverte de nouvelles classes dans le contexte de données tabulaires. Le problème de Novel Class Discovery (NCD) consiste à extraire d'un ensemble étiqueté de classes déjà connues des connaissances qui permettront de partitionner plus précisément un ensemble non étiqueté de nouvelles classes. Bien que le NCD ait récemment fait l'objet d'une grande attention de la part de la communauté, il est généralement résolu sur des problèmes de vision par ordinateur et parfois dans des conditions irréalistes. En particulier, le nombre de nouvelles classes est souvent supposé étant connu à l'avance, et leurs étiquettes sont parfois utilisées pour ajuster les hyperparamètres. Les méthodes qui reposent sur ces hypothèses ne sont pas applicables aux scénarios du monde réel. C'est pourquoi dans cette thèse nous nous concentrons sur la résolution de découverte dans les données tabulaires lorsqu'aucune connaissance a priori n'est disponible. Les méthodes développées au cours de la thèse sont appliquées à un cas réel : le diagnostic automatique des pannes dans les réseaux de télécommunication, spécifiquement les réseaux d'accès à fibre optique. Le but est d'avoir une gestion efficace des pannes, en particulier au stade du diagnostic lorsque des pannes inconnues (nouvelles classes) peuvent apparaitre.

**Title:** Novel Class Discovery in Tabular Data: an Application to Network Fault Diagnosis

**Keywords:** Novel class discovery, Clustering, Fault diagnosis, Tabular data

**Abstract:** This thesis focuses on Novel Class Discovery (NCD) in the context of tabular data. The Novel Class Discovery problem consists in extracting knowledge from a labeled set of already known classes in order to more accurately partition an unlabeled set of new classes. Although NCD has recently received a lot of of attention from the community, it is generally addressed in computer vision problems and sometimes under unrealistic conditions. In particular, the number of novel classes is often assumed to be known in advance, and their labels are sometimes used to tune hyperparameters. Methods based on these assumptions are not applicable to real-world scenarios. Thus, in this thesis we focus on discovery resolution in tabular data when no a priori knowledge is available. The methods developed in the thesis are applied to a real-world case: automatic fault diagnosis in telecommunication networks, with a focus on fiber optic access networks. The aim is to achieve efficient fault management, particularly at the diagnosis stage when unknown faults (new classes) may appear.