Deep Infinite Mixture Models for Fault Discovery in GPON-FTTH Networks

Amine Echraibi, Joachim Flocon-Cholet, Stephane Gosselin, Sandrine Vaton

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 23, 2021, accepted June 9, 2021, date of publication June 23, 2021, date of current version June 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3091328

Deep Infinite Mixture Models for Fault Discovery in GPON-FTTH Networks

AMINE ECHRAIBI[1,2], JOACHIM FLOCON-CHOLET[1], STÉPHANE GOSSELIN[1], AND SANDRINE VATON[2]

[1]Orange Labs, 22300 Lannion, France
[2]Department of Computer Science, Institut Mines Télécom Atlantique, 29238 Brest, France

Corresponding author: Amine Echraibi (amine.echraibi@imt-atlantique.fr)

This work was supported in part by the French Government (Association Nationale de la Recherche et de la Technologie) under CIFRE Agreement 2018/1563.

ABSTRACT Fault diagnosis in telecommunication networks requires extensive expert knowledge and is key to efficient network operations and high service availability. Specifically, discovering and identifying new faults occurring in the network is a challenging task. Some dominant methods in industry are based on expert systems or Bayesian networks. Both of these methods require considerable expert knowledge and time resources to construct and maintain the diagnosis system. In this paper, we propose a data driven approach for the clustering and identification of new faults, based on existing knowledge, using neural networks and infinite mixture models. In our approach deep infinite mixture models are capable of extracting interesting features from labeled data, which are then leveraged in the clustering process to identify new relevant faults in unlabeled data. We apply our method to real operational data from Fiber-to-the Home services based on Gigabit-capable Passive Optical Networks. We show that our approach can be trained end-to-end, and allows to identify and interpret new faults.

INDEX TERMS Network fault diagnosis, deep learning, infinite mixture models, variational inference.

I. INTRODUCTION

For any Internet service provider or network operator, it is crucial to quickly and efficiently diagnose the problems that occur on the network. The benefits of a good fault diagnosis system are mainly to minimize the costs of network and service operations and to enhance the customer's quality of experience. An accurate diagnosis for a particular failure will be helpful to optimize the technical mitigation process and to reduce the downtime of a service for the end user.

In network management, the fault diagnosis task can be divided in several steps. First, the detection step, which can be done proactively, aims at deciding if a customer experiences a problem and if further investigations are required. Second, in the isolation step, the goal is to identify the root cause of the problem given the current technical status. Third, the mitigation step covers all the actions required to fix the problem [1]–[4].

In this work, we study, as one particular application of Machine Learning techniques, the fault diagnosis of Fiber-to-the Home (FTTH) services based on Gigabit-capable Passive Optical Networks (GPON) [5]–[7]. In our setting, we assume that the detection of the issue has been triggered by a customer call. We thus do not need the detection step of the fault diagnosis process. This means that Anomaly Detection techniques [8], which find unexpected behaviors in data, have limited usefulness for our application. Instead, we need to perform reactive fault diagnosis where the goal is to identify or classify a failure *after* it has occurred. This goal is called Root Cause Analysis (RCA) in the literature [1]. In this framework, based on a data vector describing the customer's network and service features (alarms, optical powers, error rates, electric measures…) with multiple value types (continuous, binary, and categorical), a model will infer what type of failure the customer is facing. The intuition is that there is a specific pattern in the data that is linked with a particular type of failure.

Many machine learning–based systems were designed to perform Root Cause Analysis. One approach is to consider the traditional supervised learning scheme, where a classifier is trained using data labeled according to the different known types of failures [9]–[12]. A well-known limitation of supervised learning approaches is that they are not able to deal with zero-day anomalies, that is to say anomalies which are not embedded in the training dataset. Another model widely used for communication network management

The associate editor coordinating the review of this manuscript and approving it for publication was Claudio Cusano.

is Bayesian Networks [13]–[16], [17]. In the latter methodology, a directed acyclic graph, called a Bayesian graph, is built to model the network topology and dependencies between network features: each node represents an observed variable and connections between nodes represent conditional dependencies of random variables representing the features.

While all these approaches are appealing, they might not be able to keep up with the large size, complexity and highly dynamic nature of a network. A lot of changes can happen on a real network: new equipment or services can be deployed, the configuration of some parts of the network can change, we can collect new data that were not available previously, or more importantly, new types of failures can appear due to all these changes. The ever-evolving essence of a network leads to a situation where, at some point, there is a need to update the diagnosis system in order to maintain the best quality of service.

Using standard approaches such as classifiers, we would need to retrain our classification model with new features and new labeled data for new types of faults, which generally are not available, especially when the fault is very recent. With the Bayesian network framework, taking into account new types of failures, new data or new equipment would mean updating the Bayesian graph by adding and/or removing nodes and edges as well as changing conditional probabilities of the edges. This would require a high network expertise and may be unrealistic if the network is too complex. Though it is possible to derive the Bayesian graph from data [18], it becomes unfeasible in practice when dealing with large datasets composed of hundreds or thousands of features.

On the other hand, in order to discover new diagnoses, one needs to perform some data exploration and to find clusters corresponding to new types of faults. Although good performance can be reached with unsupervised clustering methods in a low-dimensional space, as shown in [19], problems may arise when using a distance metric in a high-dimensional space where the concept of proximity becomes meaningless [20], [21]. Furthermore, in unsupervised methods, the clusters may be relevant from a technical point of view (e.g. grouping individuals because they use the same type of equipment), but this type of clustering does not provide any new information that could be used to highlight a new fault. The obvious way to bypass this issue is to resort to cumbersome feature selection and preprocessing of the data by hand, which requires deep expert knowledge of the environment that is constantly changing, and can be expensive time-wise.

The closest machine learning paradigm to the fault discovery problem is semi-supervised clustering, where the clustering is guided by prior knowledge [22]–[25]. In this framework, a partial information on the data is provided in the form of labels. Clusters formed using semi-supervised approaches attempt to respect the constraints defined by the labels. However, the proposed algorithms were always used to improve the learning process based on a finite set of classes. Therefore, the number of clusters is known in advance and there is no clear evidence that these methods will be able to discover new classes.

In this paper, we propose a machine learning algorithm based on deep neural networks and infinite Gaussian Mixture Models (GMM). Our approach combines the power of neural networks, and the exploration potential of infinite mixture models. The neural network identifies useful features for the fault discovery process (based on the labeled faults), then the infinite mixture model clusters the data based on the neural network representation. Thus, the fault discovery process is done on a more representative space of the faults.

We demonstrate our approach on data derived from GPON-FTTH networks, we show that the neural network is capable of identifying relevant features for the diagnosis task. Furthermore, we show that the clustering approach automatically identifies clusters of relevant faults. The remainder of the paper is organized as follows:

- In section II, we frame the problem of discovering new types of faults as a machine learning problem and we give an overview of the overall system.
- In section III, we detail the feature extraction task as a deep learning classification task.
- In section IV we introduce the infinite Gaussian mixture model for clustering new faults based on the extracted features.
- Finally, in section V, we show how our approach can explore an unlabeled part of a GPON-FTTH dataset in order to identify new types of faults that were previously considered as unidentified faults by an expert system.

II. PROBLEM DESCRIPTION & THE DIAGNOSIS SYSTEM

A. PROBLEM STATEMENT

Discovering and diagnosing faults from network data is extremely challenging due to three key properties of large telecommunication networks:

- Scale: when dealing with large telecommunication networks, the scale of the data in terms of dimensionality and the number of instances adds to the complexity of the diagnosis system, and restricts further the approaches that one can adopt to solve the diagnosis problem.
- Novelty: a second issue is the adaptability of the diagnosis system. Given a dynamic corpus of data gathered from the network, the system needs to identify previously known faults if they occur, in addition to clustering and identifying new patterns of faults.
- Uninformative (noisy) features: when dealing with network data, not all features and data dimensions will be relevant for a specific fault, hence adding noise to the relevant information. Therefore, an efficient diagnosis system needs to filter out the noise in some way in order to identify relevant features for the fault in question.

A learning-based efficient diagnosis system thus needs to identify and learn previously known patterns of faults, discover new patterns unknown to the experts or the expert

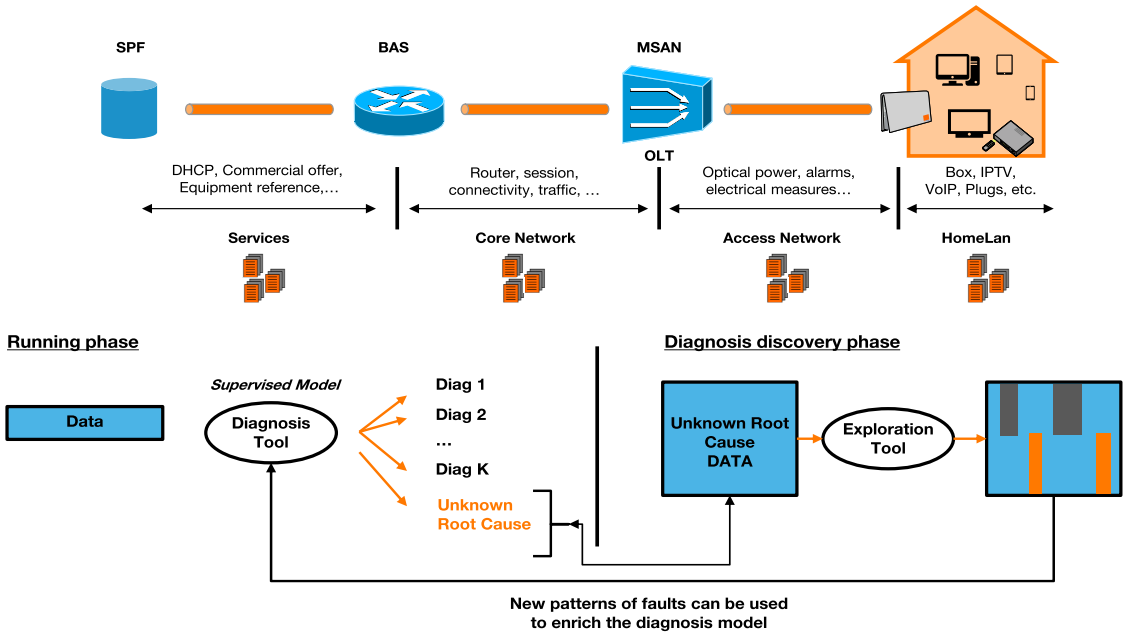

FIGURE 1. Illustration of the diagnosis process and its application to FTTH network services. Where, SPF: Service Platform; DHCP: Dynamic Host Control Protocol; BAS: Broadband Access Server; MSAN: Multi-Service Access Node; OLT: Optical Line Termination; Home LAN: Home Local Area Network; VoIP: Voice over IP.

system, and finally identify patterns relevant to actual faults on the network. An illustration of the overall diagnosis process as applied to FTTH network services is given in Figure 1.

B. DATA SPECIFICATION

The first challenge encountered when dealing with network data is the multi-typed nature of variables. For example, when treating data gathered from the optical network, one finds continuous variables such as power levels, and variables taking discrete, categorical, or binary values such as alarms of some specific device. In order to create the data corpus, we start by normalizing the continuous variables by subtracting the minimum and then scaling the values (dividing by the maximum minus the minimum value). For the categorical variables, we use the one-hot encoding i.e, if a variable X_i takes values in a set $\{c_1, \ldots, c_K\}$, then we create a representation in $\{0, 1\}^K$, for which the k^{th} column is equal to 1 if the variable X_i is equal to c_k. The resulting training data $\mathbf{X} \in \mathbb{R}^{N \times D}$ contains N lines, each representing a customer's data vector, and D columns, each representing a variable of the network. Given the large amount of data gathered D is often very high ($D \gg 1$, with often hundreds of variables). In addition to the data vector, for some customers, expert systems have identified the corresponding fault, thus providing us with a label. We denote these labels $\mathbf{y} \in \mathbb{R}^N$, each line represents the label given to a customer. A label $\mathbf{y}_n$ takes values in $\{-1, 1, \ldots, K\}$, where $\{1, \ldots, K\}$ represent the known classes, and the value -1 represents an unknown class. The key problem is to identify the labels of the class -1. This label could belong to the set of existing known faults (i.e. to the set of known classes $\{1, \ldots, K\}$) or a completely new set of faults occurring in the network. For what follows, we will denote $\mathbf{x}_n \in \mathbb{R}^D$ the data vector corresponding to the n^{th} customer, and suppose that the samples are independent and identically distributed.

C. OVERVIEW OF THE DIAGNOSIS SYSTEM

As shown in Figure 2, the diagnosis system is composed of three main blocks. The first one is the feature extractor, which will be a multi-layer perceptron or a neural network, responsible for compressing the data into a low dimensional feature vector containing the relevant information. The second one is the softmax unit responsible for classifying the known faults, its output is fed to the loss function in order to train the feature extractor. The final block is the clustering model responsible for identifying new patterns of faults based on the feature vector extracted.

The feature extractor represents the backbone of the system. Learning a good feature extractor is crucial for the performance of the diagnosis system. The feature extractor allows us to solve the scale and noise challenges simultaneously. The scale problem is solved by compressing the high dimensional raw data vector into a compact low dimensional manifold. The learned low dimensional manifold holds the information to correctly classify the known classes, thus creating noiseless features containing only the important information for each class.

The novelty challenge is tackled by the clustering model, which is in our case an infinite Gaussian mixture model. Based on learned features, the clustering model tries to identify new patterns or clusters of customer features, each cluster corresponding to a new unknown class. Then a final process

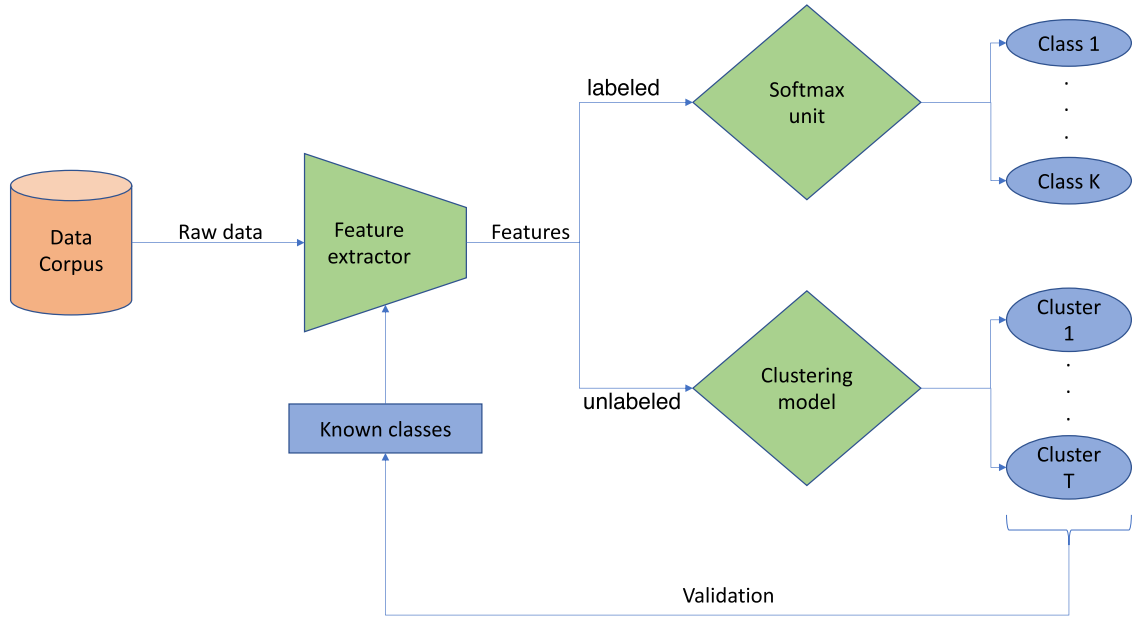

FIGURE 2. Overview of the diagnosis system.

of validation is done by an expert to identify relevant or correct new fault clusters. These correct clusters are then treated as labeled classes, the -1 labels being changed into positive values representing the new known classes. These new known classes are then re-fed to the feature extractor. In the following sections, we detail the model architectures of each block.

III. FEATURE EXTRACTION & CLASSIFICATION

A. FEATURE EXTRACTOR ARCHITECTURE

The feature extractor in our case is a deep feed forward neural network, that takes as input a data vector $\mathbf{x}_n \in \mathbb{R}^D$ and outputs a low dimensional feature vector denoted $\mathbf{f}_n \in \mathbb{R}^p$, where $p \ll D$ represents the dimension of the feature manifold. As a remainder, we can think of a neural network as a nonlinear function, built by successive applications of linear layers and nonlinear activation functions. Formally, we denote the overall parameters of the neural network θ, and:

$$\begin{aligned}\mathbf{f}_n &= g_\theta(\mathbf{x}_n) \\ &= \sigma \circ f^{(L-1)} \circ \ldots \circ \sigma \circ f^{(2)} \circ \sigma \circ f^{(1)}(\mathbf{x}_n)\end{aligned} \tag{1}$$

where, each function $f^{(l)}$ represents the l^{th} application of a linear or fully connected layer, defined by:

$$f^{(l)}(\mathbf{h}) = \mathbf{h}^T \mathbf{W}_l + \mathbf{b}_l \tag{2}$$

$\mathbf{W}_l$, and $\mathbf{b}_l$ represent the weight matrix and bias vector for the l^{th} layer. $\circ$ represents the composition of functions operator. Between each application of two linear layers, a non linear function is applied, denoted by σ, representing the ReLU activation function defined by

$$\sigma(\mathbf{h}) = \max(0, \mathbf{h}) \tag{3}$$

where the max is taken term by term.

For our application, we consider a neural network with 4 hidden layers ($L - 1 = 4$), where the dimension of each layer is about half of that of the previous layer. The previous choices are based on common practices in the deep learning community. The choice of the number of layers is done by cross-validation, where we choose the lowest number of hidden layers for which the classification error rate does not change. An overview of the architecture of the feature extractor is given in Figure 3. We detail the loss function and the learning process in the next subsection.

B. LEARNING FEATURES BY CLASSIFICATION

In order to learn the parameters of the feature extractor (step 1 of Algorithm 1), we define a classification task based on the K known classes. Given the feature vector of a customer denoted by $\mathbf{f}_n$, the softmax unit outputs the probability of each customer being in class k as:

$$\mathbb{P}[\mathbf{y}_n = k|\mathbf{f}_n] = \frac{e^{\mathbf{f}_n^T \mathbf{W}_{L,k} + \mathbf{b}_{L,k}}}{\sum_{j=1}^{K} e^{\mathbf{f}_n^T \mathbf{W}_{L,j} + \mathbf{b}_{L,j}}}. \tag{4}$$

where $\mathbf{W}_L, \mathbf{b}_L$ are the weights and biases of the last layer of the whole neural network (feature extractor followed by the softmax unit). The optimization of the parameters of the neural network is done by minimising the cross-entropy loss

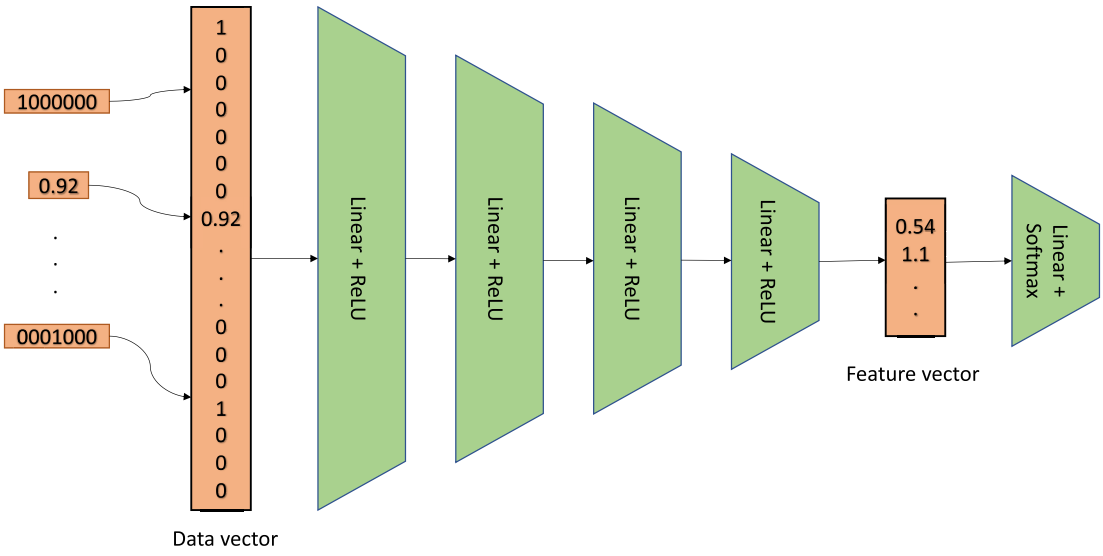

FIGURE 3. Architecture of the feature extractor followed by the softmax unit.

Algorithm 1 Deep Infinite Gaussian Mixture Model

Input: X, y
Initialize parameters
 # Step 1: Training of the feature extractor on labeled data
compute $\boldsymbol{\theta}^*$ by minimizing (5)
 # Step 2: Feature computation of unlabeled data
compute $\mathbf{f}_n$ using the last hidden layer
 # Step 3: Clustering process on unlabeled data
compute q^* by minimizing (10)
return $\hat{\mathbf{z}}_n = \arg\max_k q^*(\mathbf{z}_n = k) \quad \forall n$

function defined as:

$$l(\boldsymbol{\theta}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{1}[\mathbf{y}_n = k] \log \mathbb{P}_\theta[\mathbf{y}_n = k|\mathbf{f}_n], \qquad (5)$$

where θ represents the set of weights of the neural network. Learning θ is done by gradient descent and the standard back-propagation algorithm [26]. Following the previous learning process, we learn a feature extractor and a classifier of the known classes $\{1, \ldots, K\}$. In the next section, we show how to identify outlier clusters representing new patterns of faults.

IV. CLUSTERING MODEL

In the clustering process, the goal is to identify new patterns or clusters of unseen faults in unlabeled data (i.e. data with -1 label). The key trick in our method relies in step 2 of Algorithm 1, that consists in applying the clustering process to the feature manifold. The assumption is that the feature manifold is a more representative space in terms of filtered noise and relevant features than the original data space. Furthermore, applying the clustering process (step 3 of Algorithm 1) in the low dimensional feature manifold reduces considerably the complexity and execution time of the approach. Note that the clustering process is independent of the training of the neural network, and comes after the training process.

Given that the neural network learned highly expressive features for the known classes, it is highly likely that all information not relevant to the diagnosis task has been filtered. Furthermore, the low dimensional space of the last hidden layer is separated linearly with respect to each known class. Thus, outliers and new clusters should exist in a region outside the class regions.

The previous assumptions and deductions give guidance for the choice of the clustering model. The linearly separable nature of the feature space suggests that a simple Gaussian mixture model should be sufficient in order to identify clusters in it. The final challenge is the choice of the number of clusters. The number of clusters of faults is unknown a priori. Although methods of cross-validation can be used to identify a good candidate this always requires multiple re-iteration of the model, which could prove expensive for a scalable real-world diagnosis system. Ideally, we would like to learn the number of clusters jointly with the clusters themselves. This is made possible by a Dirichlet Process prior on cluster weights. In what follows, we detail the clustering model and learning of the number of clusters in a fully Bayesian approach.

A. LEARNING THE NUMBER OF FAULTS

Learning the number of clusters from the data has received considerable attention from the research community [27]. The main approach is to assume an infinite number of clusters, and then specify a probabilistic model of the creation of new clusters, namely the Dirichlet process (DP) [28]. The idea behind the DP is to suppose a potentially infinite number of clusters, and associate a prior with the creation of new clusters using a beta random variable. Then during inference, a posterior of the beta random variable is computed

to give a sense of the correct number of clusters needed. Thus, the number of clusters is estimated in the same fashion as the other parameters of the model. Formally, the DP prior is defined in its simplest form as:

$$\forall k \in \mathbb{N}^* \pi_k = \beta_k \prod_{l=1}^{k-1}(1-\beta_l)$$
$$\text{where } \beta_k \sim \text{Beta}(1, \eta) \tag{6}$$

where, π_k represents the prior probability of creating a cluster k. η is strictly positive and is called the concentration parameter of the DP. This parameter, as its name suggests, allows us to adjust our prior on the possible number of clusters created. The larger the concentration parameter η the faster the weights π_k tend to zero which results in a ''larger'' number of clusters (i.e, many clusters with smaller cluster weights). In the next subsection, we show how to couple the Dirichlet process with a Gaussian mixture model in the feature manifold to create a clustering model capable of identifying clusters of faults and learning the number of faults simultaneously.

B. DIRICHLET PROCESS LATENT GAUSSIAN MIXTURE MODEL

Gaussian mixture models have been considerably used to perform clustering on data. Coupled with the Dirichlet Process they enable the clustering and the learning of the number of clusters simultaneously. In our case, we consider a latent Gaussian mixture model, in the sense that the clustering is done on the feature manifold. We will denote by $\mathbf{z}_n$ the random variable representing the cluster of the n^{th} customer. The generative process defining our model is the following:

$$\mathbf{z}_n|\boldsymbol{\pi} \sim \text{Cat}(\cdot|\boldsymbol{\pi}) \text{ i.e } \mathbb{P}[\mathbf{z}_n = k]\pi_k$$
$$\mathbf{f}_n|\mathbf{z}_n = k, \quad \boldsymbol{\mu}, \Lambda \sim \mathcal{N}(\cdot|\boldsymbol{\mu}_k, \Lambda_k^{-1}) \tag{7}$$

where, we suppose that the features are Gaussian distributed with mean $\boldsymbol{\mu}_k$ and precision matrix Λ_k for cluster k. The mean and precision matrices are considered hidden (to be inferred) random variables with Gaussian and Wishart priors respectively. The cluster hidden random variable $\mathbf{z}$ follows a categorical distribution, with a prior probability π_k defined by the Dirichlet Process as shown in equation (6).

C. CLUSTER INFERENCE

In the inference process, the goal is to infer the hidden random variable distributions, that is the distributions of $\{\mathbf{z}_{1:N}, \boldsymbol{\mu}, \Lambda, \boldsymbol{\beta}\}$ knowing the feature values $\mathbf{f}_{1:N}$. This can be performed by computing or estimating the posterior distribution $p(\mathbf{z}_{1:N}, \boldsymbol{\mu}, \Lambda, \boldsymbol{\beta}|\mathbf{f}_{1:N})$. By marginalization we can then obtain the cluster assignments using maximum a posteriori estimation:

$$\hat{\mathbf{z}}_n = \arg\max_{\mathbf{z}_n} p(\mathbf{z}_n|\mathbf{f}_{1:N}) \tag{8}$$

A major challenge in the inference process, is the computation of $p(z_n|\mathbf{f}_{1:N})$. In simple parametric models, i.e. models where the number of clusters is known, inference can be done using the Expectation Maximization (EM) and the Maximum A Posteriori (MAP) algorithms [29]. For more complicated non-parametric (or infinite) models, this quantity can be estimated using Markov Chain Monte Carlo sampling methods [30]. Such approaches are well suited for small scale applications like uni-variate data, where the Markov chain explores a low dimensional space. However, in large scale applications these approaches do not scale well [31].

In our case, we adopt a recent inference paradigm called variational inference [32]. The idea behind variational inference is to transform the inference task into an optimisation task, thus leveraging the scalability and efficiency of treating such tasks. In the variational approximation paradigm, we introduce a variational distribution q to approximate the true posterior. One family of distributions leading to tractable solutions that can be efficiently implemented are factorized distributions of the form:

$$q(\boldsymbol{\mu}, \Lambda, \mathbf{z}_{1:N}, \boldsymbol{\beta}) = \prod_{k=1}^{T} q(\boldsymbol{\mu}_k|\Lambda_k)q(\Lambda_k)q(\beta_k)\prod_{n=1}^{N} q(\mathbf{z}_n) \tag{9}$$

The beta variational posterior is truncated using an upper bound on the number of clusters T by supposing $q(\beta_T = 1) = 1$ [27]. The variational optimization problem is the following one:

$$q^* = \arg\min_q \mathbb{D}_{KL}\left[q||p(\cdot|\mathbf{f}_{1:N})\right] \tag{10}$$

By solving equation (10), we attempt to identify, in the chosen family, the variational posterior q^* which is the closest, in terms of Kullback-Leibler divergence, to the true posterior $p(\cdot|\mathbf{f}_{1:N})$. After the computation of q^*, we can simply perform maximum a posteriori estimation based on the approximate distribution to obtain the cluster assignments, as follows:

$$\hat{\mathbf{z}}_n = \arg\max_k q^*(\mathbf{z}_n = k) \quad \forall n \tag{11}$$

The details of the computation of q^* are given in appendix B. A complete overview of the learning process is given in Algorithm 1.

V. EXPERIMENTAL EVALUATIONS

A. DATASET DESCRIPTION

We evaluate our approach on real network data. The dataset is built from the technical data for 86241 fiber customers. In one part of the dataset, thanks to a legacy diagnosis system (in our case an expert system based on deterministic rules), the customers have been classified into 8 known types of faults (including a normal behavior) as described in Table 1. This labelled sub-dataset corresponds to 64279 customers. Another part of the dataset (21962 customers) cannot be classified by the legacy diagnosis system, and the corresponding customers thus fall into the ''unknown faults''s category.

For each customer, we collected 1824 features coming from different parts of the network that describe the state of the customer's line. These variables mainly characterize

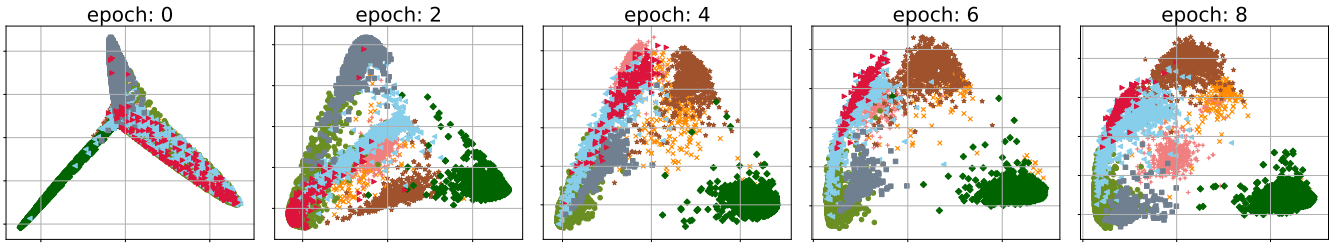

FIGURE 4. 2-D representation of the feature manifold across epochs of training (step 1 of Algorithm 1), each color represents a known fault of table 1.

TABLE 1. Description of the different classes in the dataset.

Fault type	# instances (N)
FTTH access OK	19604
Fiber cut	18282
Degraded fiber	6906
ONT-home gateway problem	6782
Gateway update problem	6050
Bad gateway configuration	3732
Account suspended	1527
TV problem	1396
Unknown Faults	21962
TOTAL	86241

TABLE 2. Overview of the network variables in the dataset.

Network scope	# variables	Description
Wide Area Network	652	GPON (OLT, ONT...)
Home LAN	446	Gateway, home devices
Internet Session	256	DHCP
Services	204	VoIP, TV
Offer	41	Customer offer profile
Miscellaneous	225	-
TOTAL	1824	
	8297	(after One-Hot Encoding)

the properties of the FTTH GPON optical system, the Home Gateway (HG), some services used by the customer (e.g. TV, VoIP) and the Internet session (see Table 2). For instance, we have a set of variables describing the GPON properties such as the optical powers received (Rx) or transmitted (Tx) by the Optical Line Termination (OLT) at the central office, and by the Optical Network Termination (ONT) at the customer side, in addition to the properties of the ONT and OLT (temperature, voltage, version etc.), the alarms encountered by the OLT, and so on.

The dataset is thus composed of a mix of categorical variables and numerical variables. The numerical features are normalized in $[0, 1]$ using min-max scaling. The categorical variables are encoded using the one-hot encoding, as explained in section II, in order to obtain a numerical dataset. Thus the final dataset is composed of $N = 86241$ instances and $D = 8297$ variables.

In order to evaluate the classification task on the known classes, the labeled dataset is split into a training set and a test set, where each represent 80% and 20% respectively. The metrics for evaluations are reported over 5 cross validation runs, we report the mean and the standard deviation for each metric. The clustering evaluation is done on the entire unknown fault dataset in a single run.

B. EVALUATION OF THE FEATURE EXTRACTOR

The first evaluation is that of the neural network representing the feature extractor. Extracting relevant features for the known classes is crucial for a good classification score, and for efficiently filtering noisy features for the subsequent clustering task. The key desirable feature of the hidden data manifold represented by the last hidden layer of the feature extractor is to be linearly separable for each known class. In order to represent this data manifold, we compute the hidden features on the test set after each epoch (full pass of stochastic gradient descent on the training set), we then transform the features into a 2D representation using kernel principal component analysis. This procedure shows plots of the hidden feature points in a 2D representation based on a Gaussian proximity measure.

As shown in Figure 4, the hidden representation is disorganised at the first epochs of training the neural network, the classes are not well separated based on the similarity measure. However, as we converge to the optimum parameters, the hidden feature manifold becomes more separable with respect to each class (epoch 8). This suggests that the features extracted are representative of each class modeled. In addition, the structure is in the form of ''lumps'' which is coherent with Gaussian model assumptions.

In order to demonstrate the effectiveness of the feature extractor further, we compare the raw two dimensional representation on the unknown faults dataset, and the two dimensional representation of the features extracted using the same dataset. As shown in Figure 6, the 2D representation on the extracted features is much more organized than the 2D representation on the raw data. Furthermore, the 2D representation of the features shows clear patterns of clusters unlike the raw data representations. This suggests that the feature extractor is capable of identifying features that permit us to separate and identify different clusters in the unknown fault dataset.

C. EVALUATION OF THE SOFTMAX UNIT

The softmax unit classifies the known faults. As a standard evaluation process we report the loss function (5) on the test

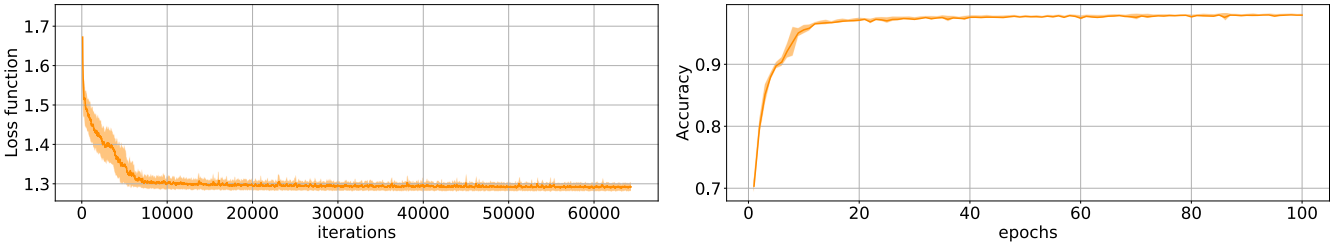

FIGURE 5. Loss function on the testset across iterations of gradient descent. Accuracy on the testset across epochs (step 1 of Algorithm 1).

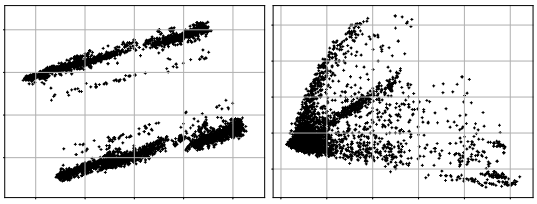

FIGURE 6. (Left) 2-D representation on the feature manifold of the unknown classes data (step 2 of Algorithm 1). (Right) 2-D representation of the raw unknown classes data.

TABLE 3. The confusion matrix on the known classes.

		Predicted classes							
		1	2	3	4	5	6	7	8
True classes	1	**1288**	96	10	39	63	13	14	4
	2	47	**3619**	0	0	0	45	14	7
	3	0	0	**6024**	3	12	5	2	4
	4	0	1	5	**18590**	0	304	621	83
	5	1	1	8	1	**18258**	7	6	0
	6	1	5	9	226	0	**6580**	74	11
	7	2	0	0	56	1	10	**6710**	3
	8	1	0	3	22	0	2	4	**1364**

set across training iterations. We report the mean and standard deviation over 5 different runs. As shown in Figure 5 (left) the loss function converges to the set of parameters representing of the minimum of (5).

In addition, we report the classification accuracy on the test set defined as:

$$\text{Accuracy} = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}[\hat{y}_n = y_n] \tag{12}$$

where, $\hat{y}_n$ represents the predicted label and y_n represents the ground-truth label in the n^{th} instance of the test set. As shown in Figure 5 (right) the classification accuracy increases across training iterations and reaches a high value, showing that the classifier is well trained and robust to errors. This is demonstrated also by the confusion matrix between the known classes, reported in table 3. The standard deviation across the different validation runs is very small, suggesting that the classifier generalizes very well on unseen test set samples.

D. EVALUATION OF THE CLUSTERING MODEL

In order to evaluate the clustering model, we begin by evaluating the fitting of the model to the unknown fault data. The latent infinite Gaussian mixture model is a probabilistic generative model, a standard approach of evaluating such models is to compute the log likelihood or the log probability of the data under the model across iterations. As shown in Figure 8, the log probability increases across iterations of updates of variational inference, and converges smoothly to a plateau, which suggests the convergence of the model.

The objective of the infinite latent Gaussian mixture model is to cluster patterns in the latent representation of unknown faults. In order to evaluate this clustering process, we report the clusters identified by the model in the 2D representation, across iterations of variational inference updates. As shown in Figure 7, the model starts with random clusters that do not fit the patterns in the unknown faults dataset. Across training iterations the ellipsoids representing the clusters fit with more accuracy the patterns seen in the 2D representation.

Furthermore, using the Dirichlet Process which learns the number of clusters, we see that across iterations of training of the clustering model, the number of clusters changes, in order to evaluate which number of clusters better fits the data. Thus, the model automatically identifies the number of clusters needed to converge to the optimal fit.

E. CLUSTER INTERPRETATION

In order to exploit the clusters and validate them, interpreting the faults discovered in each cluster is crucial. Interpretability of machine learning models is a hard task that is still an active research area. In this section, we provide an interpretation method relevant to data extracted from networks. The proposed method consists in identifying the decision rule over the features in order to reach the class assignment. One approach to accomplish this task, is to train a decision tree classifier [33] on the unknown fault data with the cluster labels as the true labels of the classifier. Thus, the resulting decision tree classifier gives us a path of logical statements on the features leading to each cluster discovered. This provides a readable interpretable rule that can be exploited by experts of the network domain to identify the fault.

In Figure 9, we give an example of such a decision path for the forth cluster discovered in the unknown fault dataset. As it can be seen the decision rule mainly involves transmission and reception powers of the optical network termination of the GPON-FTTH network. This suggests a problem with Optical Network Termination (ONT). Another example

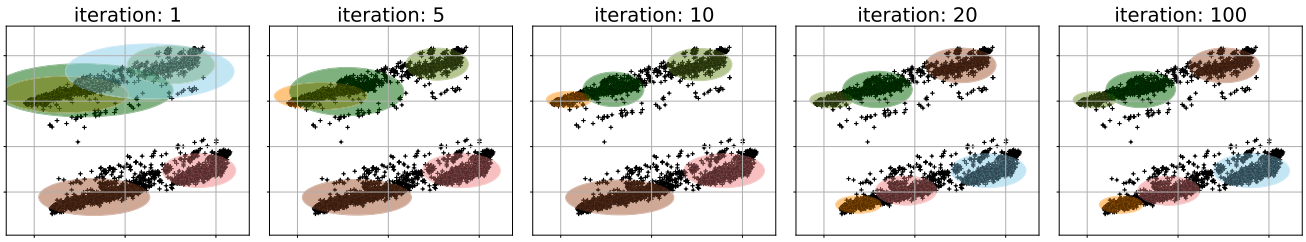

FIGURE 7. The clustering process of the infinite Gaussian mixture model on the latent feature manifold across iterations of variational inference (step 3 of Algorithm 1, i.e. Algorithm 2). The process is done on the unknown fault dataset, ellipsoids represent identified clusters of new faults.

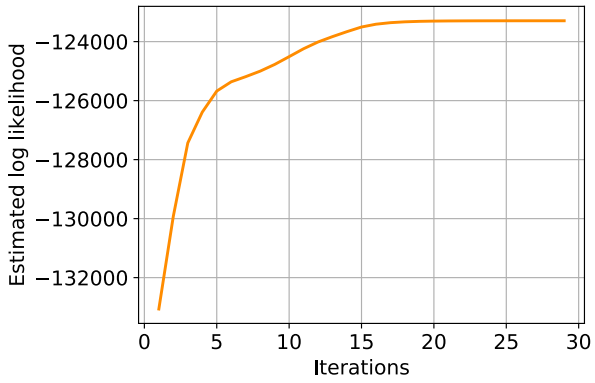

FIGURE 8. Log probability of the latent Dirichlet process Gaussian mixture model across iterations of variational inference (step 3 of Algorithm 1, i.e. Algorithm 2).

```
|--- delc.res.ftth.client.ols.mesure.ontrx <= 0.76
|   |--- olt.24.res.ont.data.vendor_1397572179 <= 0.50
|   |   |--- livebox.14.res.LANHosts <= 0.04
|   |   |   |--- olt.45.res.olt.temp <= 0.51
|   |   |   |   |--- delc.res.ftth.client.ols.mesure.onttx <= 0.77
|   |   |   |   |   |--- delc.res.ftth.histoalarms.sample.duration <= 0.70
|   |   |   |   |   |   |--- olt.24.res.ont.ols.tx <= 0.52
|   |   |   |   |   |   |   |--- class: 4
```

FIGURE 9. An example of a decision rule leading to cluster 4.

```
|--- delc.res.ftth.client.ont.lb.link_NOK >  0.50
|   |--- class: 3
```

FIGURE 10. Clear decision leading to cluster 3.

where the decision is quite clear is given in Figure 10, where the issue is the link between the livebox (lb) or home router of the client and the Optical Network Termination (ONT).

VI. CONCLUSION

In this paper, we proposed a new machine learning approach to identify unknown patterns of faults from network data. Our approach is based on infinite mixture models and neural networks. Using a neural network we learn interesting features for the unknown fault dataset where new classes of faults may exist. Then an infinite Gaussian mixture model is applied on the features extracted in order to identify new faults and to cluster the new data without knowing a priori the number of clusters. Experiments on FTTH-GPON real operational data show that our approach effectively identifies new patterns of faults. A simple decision tree analysis then allows to derive the root causes of each new fault discovered. Our approach is not restricted to the diagnosis task, it can be applied to various other network management applications such as Intrusion Detection Systems (IDS). Furthermore, one can generalize the overall model to an end-to-end deep probabilistic graphical model, thus merging the neural network training and cluster inference in a single model.

APPENDIX A
IMPLEMENTATION DETAILS

In this appendix, we give further details regarding the implementation of the different blocks of the algorithm. The neural network is a 4 hidden layer fully connected neural network, with Rectified Linear activation Units (ReLUs). In order to learn the neural network parameters, we use the ADAM optimizer [34], with a standard learning rate of $\alpha = 0.001$, decreasing exponentially as the iterations run longer. The loss function is the softmax cross-entropy loss defined by (5), we add an L2 regularization term with a regularization parameter $\lambda = 0.0005$. The regularization is added in order to limit the overfitting of the neural network.

The overall training is done using the Pytorch library [35], on a NVIDIA GPU GeForce RTX 2080 Ti, parallelizing the computations of the neural network and making the training process faster. The clustering process is done following Algorithm 2, the implementation is optimized for matrix operations making the iterations of equations (19, 20, 21, 22, 23) run efficiently.

APPENDIX B
VARIATIONAL INFERENCE FOR THE CLUSTERING MODEL

In this appendix, we provide a review of variational inference specifically the problem of solving equation (10):

$$q^* = \arg\min_q \mathbb{D}_{KL}\left[q||p(\cdot|\mathbf{f}_{1:N})\right] \tag{13}$$

The posterior distribution by Bayes rule is equal to the joint distribution divided by the likelihood: $p(\cdot|\mathbf{f}_{1:N}) = \frac{p(\cdot,\mathbf{f}_{1:N})}{p(\mathbf{f}_{1:N})}$. The likelihood does not contribute to the minimization criterion, Thus the minimization criterion of equation (13) is equivalent to:

$$q^* = \arg\min_q \mathbb{D}_{KL}\left[q||p(\cdot,\mathbf{f}_{1:N})\right] \tag{14}$$

The full generative model in mathematical form is given by:

$$\begin{aligned}
\forall k \quad \beta_k &\sim \text{Beta}(1, \eta) \\
\pi_k &= \beta_k \prod_{l=1}^{k-1}(1-\beta_l) \\
\boldsymbol{\mu}_k|\Lambda_k &\sim \mathcal{N}(\cdot|\mathbf{m}_0, (\kappa_0\Lambda_k)^{-1}) \\
\Lambda_k &\sim \mathcal{W}(\cdot; \mathbf{L}_0, \nu_0) \\
\mathbf{z}|\boldsymbol{\pi} &\sim \text{Cat}(\cdot|\boldsymbol{\pi}) \text{ i.e } \mathbb{P}[\mathbf{z}=k] = \pi_k \\
\mathbf{f}_n|z_n = k, \boldsymbol{\mu}, \Lambda &\sim \mathcal{N}(\cdot|\boldsymbol{\mu}_k, \Lambda_k^{-1})
\end{aligned}$$

where, $\{z_{1:N}, \boldsymbol{\mu}, \Lambda, \beta\}$ represent the hidden random variables to infer, and $\mathbf{f}_{1:N}$ represent the feature vector of the observable random variables. We denote by $\mathcal{W}$ the Wishart distribution, and $\mathcal{N}$ the Gaussian distribution. For simplicity of notation, we denote by $\boldsymbol{\zeta} = \zeta_{1:M} = \{z_{1:N}, \boldsymbol{\mu}, \Lambda, \beta\}$ and $\boldsymbol{\zeta}_{-m}$ the vector of all random variables except the m^{th} index:

$$\begin{aligned}
J(q) &= \mathbb{D}_{KL}\left[q(\boldsymbol{\zeta})||p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})\right] \\
&= \sum_{\boldsymbol{\zeta}} q(\boldsymbol{\zeta}) \log \frac{q(\boldsymbol{\zeta})}{p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})} \\
&= \sum_{\zeta_m}\sum_{\boldsymbol{\zeta}_{-m}} \underbrace{q(\zeta_m)q(\boldsymbol{\zeta}_{-m})}_{\text{eq (9) Factorisation}} \log \frac{q(\zeta_m)q(\boldsymbol{\zeta}_{-m})}{p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})} \\
&= \sum_{\zeta_m} q(\zeta_m)[-\sum_{\boldsymbol{\zeta}_{-m}} q(\boldsymbol{\zeta}_{-m}) \log p(\boldsymbol{\zeta}, \mathbf{f}_{1:N}) \\
&\quad + \log q(\zeta_m)] - \underbrace{\sum_{\zeta_m} q(\zeta_m)}_{=1} \mathbb{H}[q(\boldsymbol{\zeta}_{-m})] \\
&= \sum_{\zeta_m} q(\zeta_m)\left[-\mathbb{E}_{\boldsymbol{\zeta}_{-m}\sim q}\left[\log p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})\right]\right. \\
&\quad \left. + \log q(\zeta_m)\right] - \mathbb{H}[q(\boldsymbol{\zeta}_{-m})] \\
&= \mathbb{D}_{KL}\left[q(\zeta_m)||f(\zeta_m)\right] - \mathbb{H}[q(\boldsymbol{\zeta}_{-m}] \qquad (15)
\end{aligned}$$

where:

$$f(\zeta_m) \propto \exp\left(\mathbb{E}_{\boldsymbol{\zeta}_{-m}\sim q}\left[\log p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})\right]\right) \qquad (16)$$

Given the decomposition of $J(q)$ to a term depending on $q(\zeta_m)$ and a term depending on $q(\boldsymbol{\zeta}_{-m})$, we can minimize with respect to each $q(\zeta_m)$, $\forall m$:

$$\begin{aligned}
q^*(\zeta_m) &= \arg\min_{q(\zeta_m)} J(q(\zeta_m)q^*(\boldsymbol{\zeta}_{-m})) \\
&= \arg\min_{q(\zeta_m)} \left[\mathbb{D}_{KL}\left[q(\zeta_m)||f^*(\zeta_m)\right] - \mathbb{H}[q^*(\boldsymbol{\zeta}_{-m}]\right] \\
&= \arg\min_{q(\zeta_m)} \mathbb{D}_{KL}\left[q(\zeta_m)||f^*(\zeta_m)\right] \\
&\quad \text{where } f^*(\zeta_m) \propto \exp\left(\mathbb{E}_{\boldsymbol{\zeta}_{-m}\sim q^*}\left[\log p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})\right]\right) \\
&= f^*(\zeta_m)
\end{aligned}$$

Algorithm 2 Variational Inference

Input: $\mathbf{f}_{1:N}, T, \eta, \kappa_0, \mathbf{m}_0, \nu_0, \eta, \mathbf{L}_0$
Initialize ϕ
while J changes more than 10^{-6} **do**
  Compute: γ_1, γ_2 (19)
  Compute: κ, ν (20)
  Compute: $\mathbf{m}$ (21)
  Compute: $\mathbf{L}$ (22)
  Compute: $\phi_{n,k}$ $\forall n, \forall k$ (23)
  Compute: J (15)
end while
$z_n = \arg\max_k \phi_{nk}$ $\forall n$
return $z_{1:N}, \phi$

Therefore, the solution to equation (10) is in the form of fixed point equations often referred to as the mean field update equations:

$$\log q^*(\zeta_m) = \text{const} + \mathbb{E}_{\boldsymbol{\zeta}_{-m}\sim q^*}\left[\log p(\boldsymbol{\zeta}, \mathbf{f}_{1:N})\right] \quad \forall m \qquad (17)$$

These equations give us the form of each variational distribution and the fixed point update equations of its parameters. The variational distributions are the following:

$$\begin{aligned}
q(\beta_k) &= \text{Beta}(\beta_k; \gamma_{1,k}, \gamma_{2,k}) \\
q(\boldsymbol{\mu}_k|\Lambda_k) &= \mathcal{N}(\boldsymbol{\mu}_k; \mathbf{m}_k, (\kappa_k\Lambda_k)^{-1}) \\
q(\Lambda_k) &= \mathcal{W}(\Lambda_k; \mathbf{L}_k, \nu_k) \\
q(z_n) &= \text{Cat}(z_n; \phi_n) \qquad (18)
\end{aligned}$$

where the variational parameters are updated iteratively using the following equations:

$$\gamma_{1,k} = 1 + \sum_{n=1}^{N}\phi_{nk} \quad \gamma_{2,k} = \eta + \sum_{n=1}^{N}\sum_{l=k+1}^{T}\phi_{nl} \qquad (19)$$

$$\kappa_k = \kappa_0 + \sum_{n=1}^{N}\phi_{nk} \quad \nu_k = \nu_0 + \sum_{n=1}^{N}\phi_{nk} + 1 \qquad (20)$$

$$\mathbf{m}_k = \frac{\kappa_0\mathbf{m}_0 + \sum_{n=1}^{N}\phi_{nk}\mathbf{f}_n}{\kappa_k} \qquad (21)$$

$$\begin{aligned}
\mathbf{L}_k^{-1} &= \mathbf{L}_0^{-1} + \kappa_0(\mathbf{m}_k - \mathbf{m}_0)(\mathbf{m}_k - \mathbf{m}_0)^T \\
&\quad + \sum_{n=1}^{N}\phi_{nk}(\mathbf{f}_n - \mathbf{m}_k)(\mathbf{f}_n - \mathbf{m}_k)^T \qquad (22)
\end{aligned}$$

$$\begin{aligned}
\log\phi_{nk} &= -\frac{1}{2}\left[\frac{d}{\kappa_k} + \nu_k(\mathbf{f}_n - \mathbf{m}_k)^T\mathbf{L}_k(\mathbf{f}_n - \mathbf{m}_k)\right] \\
&\quad + \frac{1}{2}\left[\sum_{i=1}^{d}\psi\left(\frac{\nu_k + 1 - i}{2}\right) + \log\det(\mathbf{L}_k)\right] \\
&\quad + \psi(\gamma_{1,k}) - \psi(\gamma_{1,k} + \gamma_{2,k}) \\
&\quad + \sum_{l=1}^{k-1}\left[\psi(\gamma_{2,l}) - \psi(\gamma_{1,l} + \gamma_{2,l})\right] \qquad (23)
\end{aligned}$$

where ψ represents the digamma function, and the variables $\{\kappa_0, \mathbf{m}_0, \nu_0, \eta, \mathbf{L}_0\}$ are fixed hyperparameters. Algorithm 2 summarizes the learning procedure of the infinite mixture model using variational inference.

REFERENCES

[1] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities," *J. Internet Services Appl.*, vol. 9, no. 1, p. 16, Dec. 2018.

[2] S. Ayoubi, N. Limam, M. A. Salahuddin, N. Shahriar, R. Boutaba, F. Estrada-Solano, and O. M. Caicedo, "Machine learning for cognitive network management," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 158–165, Jan. 2018.

[3] M. Steinder and A. S. Sethi, "A survey of fault localization techniques in computer networks," *Sci. Comput. Program.*, vol. 53, no. 2, pp. 165–194, 2004.

[4] 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 Trans. Netw. Service Manage.*, vol. 16, no. 4, pp. 1537–1551, Dec. 2019.

[5] P. Chanclou, S. Gosselin, J. F. Palacios, V. L. 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 Commun. Mag.*, vol. 44, no. 8, pp. 29–35, Aug. 2006.

[6] *Gigabit-Capable Passive Optical Networks (GPON): General Characteristics*, Standard ITU-T G.984.1, 2008.

[7] *Gigabit-Capable Passive Optical Networks (G-PON): ONT Management and Control Interface Specification*, Standard ITU-T G.984.4, 2008.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009, doi: 10.1145/1541880.1541882.

[9] M. Y. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer, "Pinpoint: Problem determination in large, dynamic Internet services," in *Proc. Int. Conf. Dependable Syst. Netw.*, 2002, pp. 595–604.

[10] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. Brewer, "Failure diagnosis using decision trees," in *Proc. Int. Conf. Autonomic Comput.*, 2004, pp. 36–43.

[11] J. S. Baras, M. Ball, S. Gupta, P. Viswanathan, and P. Shah, "Automated network fault management," in *Proc. (MILCOM)*, vol. 3, 1997, pp. 1244–1250.

[12] M. Adda, K. Qader, and M. Al-Kasassbeh, "Comparative analysis of clustering techniques in network traffic faults classification," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 5, no. 4, pp. 6551–6563, 2017.

[13] S. R. Tembo, S. Vaton, J. L. Courant, S. Gosselin, and M. Beuvelot, "Model-based probabilistic reasoning for self-diagnosis of telecommunication networks: Application to a GPON-FTTH access network," *J. Netw. Syst. Manage.*, vol. 25, no. 3, pp. 558–590, Jul. 2017.

[14] L. Bennacer, Y. Amirat, A. Chibani, A. Mellouk, and L. Ciavaglia, "Self-diagnosis technique for virtual private networks combining Bayesian networks and case-based reasoning," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 1, pp. 354–366, Jul. 2014.

[15] R. Moštak, B. Spahija, and V. Z. Deljac, "Fault diagnosis in optical access network using Bayesian network," in *Proc. 18th Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, 2010, pp. 341–345.

[16] R. M. Khanafer, B. Solana, J. Triola, R. Barco, L. Moltsen, Z. Altman, and P. Lazaro, "Automated diagnosis for UMTS networks using Bayesian network approach," *IEEE Trans. Veh. Technol.*, vol. 57, no. 4, pp. 2451–2461, Jul. 2008.

[17] Y. Matsuo, Y. Nakano, A. Watanabe, K. Watanabe, K. Ishibashi, and R. Kawahara, "Root-cause diagnosis for rare failures using Bayesian network with dynamic modification," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[18] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[19] A. Echraibi, J. Flocon-Cholet, S. Gosselin, and S. Vaton, "An infinite multivariate categorical mixture model for self-diagnosis of telecommunication networks," in *Proc. 23rd Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Feb. 2020, pp. 258–265.

[20] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *Proc. Int. Conf. Database Theory*. Berlin, Germany: Springer, 2001, pp. 420–434.

[21] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Trans. Knowl. Discovery Data*, vol. 3, no. 1, pp. 1–58, 2009.

[22] S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2004, pp. 59–68.

[23] E. Bair, "Semi-supervised clustering methods," *Wiley Interdiscipl. Rev. Comput. Statist.*, vol. 5, no. 5, pp. 349–361, Sep. 2013.

[24] Z. Yu, P. Luo, J. Liu, H.-S. Wong, J. You, G. Han, and J. Zhang, "Semi-supervised ensemble clustering based on selected constraint projection," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2394–2407, Dec. 2018.

[25] Y. Ren, K. Hu, X. Dai, L. Pan, S. C. H. Hoi, and Z. Xu, "Semi-supervised deep embedded clustering," *Neurocomputing*, vol. 325, pp. 121–130, Jan. 2019.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[27] D. M. Blei and M. I. Jordan, "Variational inference for Dirichlet process mixtures," *Bayesian Anal.*, vol. 1, no. 1, pp. 121–143, Mar. 2006.

[28] J. Sethuraman, "A constructive definition of Dirichlet priors," *Statistica Sinica*, vol. 4, no. 2, pp. 639–650, 1994.

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *J. Roy. Stat. Soc. B, Methodol.*, vol. 39, no. 1, pp. 1–22, 1977.

[30] C. E. Rasmussen, "The infinite Gaussian mixture model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 554–560.

[31] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. Amer. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, Apr. 2017.

[32] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Mach. Learn.*, vol. 37, no. 2, pp. 183–233, 1999.

[33] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Boca Raton, FL, USA: CRC Press, 1984.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[35] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," Facebook AI Res., NIPS, Long Beach, CA, USA, Tech. Rep., 2017.

AMINE ECHRAIBI received the M.S. degree in information processing and computer science from Institut Mines-Télécom Atlantique (IMT Atlantique), Brest, France, with a focus on machine learning, and the master's degree in research from the University of Rennes 1, Rennes, France, with a focus on signal processing. He is currently pursuing the Doctor of Philosophy degree with IMT Atlantique.

In 2018, he joined the Orange Labs, Lannion, France, in the context of his Ph.D. work, as a Research Engineer working on probabilistic graphical models and deep learning for fault diagnosis in telecommunication networks. His research interests include probabilistic graphical models and inference methods for real-world data applications.

JOACHIM FLOCON-CHOLET received the M.S. degree in acoustics and signal processing from Université Pierre et Marie Curie (Paris VI), France, in partnership with the Telecom ParisTech and IRCAM, and the Ph.D. degree in audio classification for real-time applications from the Orange Labs, in 2016.

Since 2017, he has been conducting research on data mining applied to network diagnosis systems.

STÉPHANE GOSSELIN joined the Research and Development Center of France Telecom (currently, the Orange Labs), in 1993, where he first studied fast free-space optical switching systems. He worked on WDM transport networks, from 1997 to 2007. He led a research unit on optical transport network technologies, from 1999 to 2003, and an in charge of France Telecom prospective research on optical core, access, and home networks, from 2003 to 2006. He led several research projects on broadband communication systems, networks, and network management with the Orange Labs, Lannion, France, from 2007 to 2016. He is currently an in charge of a research program on cognitive operation of future networks and services with the Orange Labs, including research activities on the application of artificial intelligence to advanced network management. He was the Technical Leader of the European Project FP7 COMBO, from 2013 to 2016. He was involved in several French and European projects, such as ACTS/DEMON, IST/TOPRATE, IST/NoE/e-Photon/One, BONE, and ICT/COMBO dealing with optics, broadband communication networks, and fixed mobile network architecture. He has coauthored the second European Strategic Research Agenda in photonics, in 2010, and holds four international patents. He has authored or coauthored about 85 national or international articles or communications and two book chapters.

SANDRINE VATON received the Ph.D. degree in signal processing from the Telecom Paris. She is currently a Full Professor with the Department of Computer Science, IMT Atlantique. She also leads the Math and Net Research Group, Lab-STICC laboratory. The research group aims to design, describe, manage, secure, and control various aspects of operator communication networks, and relies on stochastic models and algorithms to capture the high variability of such networks. She is also a Gender Equality Advisor with IMT Atlantique. She has supervised 17 Ph.D. theses and has been involved in several collaborative research projects at European level (DEMONS, ETICS, and EuroNGI), French level (ANR VIPEER and ANR OSCAR), or in collaboration with South America. She coordinates actions aimed at improving gender balance in study and work conditions and promoting computer science initiatives that reach girls. She received an accreditation to supervise research in computer science from the University of Rennes 1, France.