Unsupervised approach for an optimal representation of the latent space of a failure analysis dataset

Abbas Rammal, Kenneth Ezukwoke, Anis Hoayek, Mireille Batton-Hubert

Please cite as :
Rammal, A., Ezukwoke, K., Hoayek, A. *et al.* Unsupervised approach for an optimal representation of the latent space of a failure analysis dataset. *J Supercomput* (2023). https://doi.org/10.1007/s11227-023-05634-0

Unsupervised approach for an optimal representation of the latent space of a failure analysis dataset

Abbas Rammal[1] · Kenneth Ezukwoke[1] · Anis Hoayek[1] · Mireille Batton-Hubert[1]

[1] Ecole des Mines de Saint-Etienne, Mathematics and Industrial Engineering, Organisation and Environmental Engineering, Henri FAYOL institute, 42023 Saint-Étienne, France

Abbas Rammal rammal_abbass@hotmail.com
Kenneth Ezukwoke ifeanyi.ezukwoke@emse.fr
Anis Hoayek anis.hoayek@emse.fr
Mireille Batton-Hubert batton@emse.fr

Abstract

Microelectronics production failure analysis is an important step in improving product quality and development. In fact, the understanding of the failure mechanisms and therefore the implementation of corrective actions on the cause of the failure depend on the results of this analysis. These analyses are saved under textual features format. Then such data need first to be preprocessed and vectorized (converted to numeric). Second, to overcome the curse of dimensionality caused by the vectorisation process, a dimension reduction is applied. A two-stage variable selection and feature extraction is used to reduce the high dimensionality of a feature space. We are first interested in studying the potential of using an unsupervised variable selection technique, the genetic algorithm, to identify the variables that best demonstrate discrimination in the separation and compactness of groups of textual data. The genetic algorithm uses a combination of the K-means or Gaussian Mixture Model clustering and validity indices as a fitness function for optimization. Such a function improves both compactness and class separation. The second work looks into the feasibility of applying a feature extraction technique. The adopted methodology is a Deep learning algorithm based on variational autoencoder (VAE) for latent space disentanglement and Gaussian Mixture Model for clustering of the latent space for cluster identification. The last objective of this paper is to propose a new methodology based on the combination between variational autoencoder (VAE) for the latent space disentanglement, and genetic algorithm (GA) to find, in an unsupervised way, the latent variables allowing the best discrimination of clusters of

failure analysis data. This methodology is called VAE-GA. Experiments on textual datasets of failure analysis demonstrate the effectiveness of the VAEGA proposed method which allows better discrimination of textual classes compared to the use of GA or VAE separately or the combination of PCA with GA (PCA-GA) or a simple Auto-encoders with GA (AE-GA).

Keywords Failure analysis · Unsupervised variable selection · Genetic algorithm · Feature extraction · Variational auto-encoder · Natural language processing · Artificial intelligence

Introduction

In the development of microelectronic technologies, failure analysis allows the study of potential failure mechanisms to define corrective solutions. Failure analysis is useful at all stages of a product's life, from design to use. At each of these stages, the analysis results will make it possible to optimize the products and ensure greater reliability. It is therefore crucial to be able to draw information from each analysis and therefore to ensure a high success rate of the analyses. This rate has also become one of the key indicators monitored by laboratories. One of the most fundamental challenges of Failure Analysis (FA) 4.0 project for the digital world is to ensure that increasingly complex electronic systems operate reliably and securely [4]. This is essential in safety-critical applications such as autonomous vehicles and in digitized industrial production (Industry 4.0).

The Failure Reporting, Analysis and Corrective Action System (FRACAS) is used by many organizations to track failures associated with their products [R7]. The fundamental tasks of FRACAS method include: Recording and capturing information about failures and problems; Identifying, selecting and prioritizing failures and problems; Identifying, implementing and verifying corrective actions to prevent recurrence of failures; Providing information from failure analysis and corrective actions in order to support reliability data analysis; Providing report summaries of incidents counts, and providing failure data used for reliability and quality metrics [23].

One of the fundamental challenges facing our digital world is to provide an innovative method based on artificial intelligence to quickly analyze and detect failures during the development and manufacture of electronic components and systems, using the final report generated by FRACAS. The Natural Language Processing (NLP) has been applied on the FRACAS final report to transform unstructured text from documents and databases into normalized structured data suitable for analysis or driving machine learning algorithms (ML) [37].

Text mining is an important and popular data-mining topic, where a fundamental goal is to enable users to extract informative data from text assets and perform related operations on text, like recovery, clustering or classification and synthesis. For text clustering, one of the most important steps is feature selection and dimension reduction, because not all features in the text dataset are useful and valuable for unsupervised classification. Before applying these techniques of dimension reduction or variable selection, we first need to convert textual data into numeric data using methods called vectorization which are known in NLP as word embeddings. This technique generally tries to map a word using a dictionary to its vector form [27].

Dimension Reduction techniques seek to find a transformation function that can transform a data such that most information of the high-dimensional data is kept in a much lower-dimensional subspace [46]. Some feature extraction methods have been successfully used in text categorization, such as principal component analysis (PCA) [44], Auto-Encoders and Variational Auto-Encoders (VAE) [15]. On the other hand, variable selection approach reduces the dimension of a dataset of variables potentially relevant with respect to a given phenomenon by finding the best minimum subset without transforming data into a new set. For this, we implement the selection of important variables that exclude the noninformative variables in order to improve the performance of complex regression and classification [24].

Mathematically speaking, various techniques have been employed to select optimal subsets of variables: successive projections algorithm [17], backward/forward selection algorithm [12], reweighted adaptive competitive sampling, importance of variables for projection, elimination of non-informative variables [7], interval partial least squares regression [26], Monte Carlo-elimination of non-informative variables [12], competitive adaptive reweighted sampling partial least squares [26], simulated annealing, methods based on artificial neural networks, ant colony optimization [2], variable importance in partial least squares projection, loading weight, regression coefficient [26], sequential search, particle swarm optimization [18], etc. However, most of these techniques require additional a priori complementary information, which may not be well suited to our textual datasets. Since not all data instances X in the dataset have known response values Y, the feature selection process is called “Unsupervised Selection Variable.” A growing class of evolutionary-optimization computations that became popular through the work of Holland is the Genetic Algorithms (GA) [38]. They contain an essential category of interrogative methods depending on the stochastic algorithms that imitate the principles of biological evolution in nature. As an advantage, they have the capability of analyzing the space of variables in a large but reasonable amount of time. In addition, the GA are deeply powerful methods employed in several band selection problems, including text clustering.

In the mechanism of GA, a solution (i.e., a point in the research area), called a “chromosome”, is constructed with chosen variables positioned as genes. The GA needs an objective function, called “fitness” function. The role of this function is to compute a quantitative value defining the fitness of each chromosome. The mechanism of evolution allows the algorithm to handle a finite set of chromosomes (the population). These chromosomes evolve by certain operators in each generation, such as crossover and mutations, which simulates the developments that result in natural reproduction [38]. The fitness function is arguably the most important part of a GA having the role to measure the quality of the chromosome in the population according to the given optimization objective. Since we want to classify the textual data in separate and compact clusters, we propose to test the K-means clustering with different well-known and widely used validity indices as a fitness function to evaluate the quality of the obtained clusters.

There are several studies on the application of GA on textual data. Genetic algorithm-oriented latent semantic features are proposed to obtain better representation of documents in text classification [45]. Genetic algorithm-based unsupervised feature selection technique with mean absolute difference as a fitness function was applied to improve text clustering [1]. GA is proposed to select optimal text features using term frequency-inverse document frequency (TFIDF) to reduce document term relationships [36]. GA has been applied for text clustering using ontology and evaluating the validity of various semantic similarity measures [40].

In this study, a two-stage feature selection and feature extraction is used to: (1) reduce the high dimensionality of a feature space composed of a large number of terms; (2) Remove redundant and irrelevant features from the feature space; (3) Reduce the computational complexity of the machine learning algorithms used in text classification and improve their performance. In the first stage, we apply a feature extraction technique using Variational Autoencoders (VAE) to represent complex textual data dimension via a low-dimensional latent space, which is learned in an unsupervised way. In the second stage, we apply an unsupervised variable selection technique using a genetic algorithm to identify more informative and useful features from the textual dataset containing a very large number of words, and to show whether the features selected by the proposed method can significantly improve the performance of unsupervised textual classification. This new methodology VAE – GA makes it possible to project the textual data on a new latent space and then to select the discriminant latent variables by GA to classify the textual data in an unsupervised way. This methodology will be compared with the application of VAE and GA separately, and the combination of PCA-GA and AE-GA on real datasets. This article will be structured as: in Sect. 2, mathematical methods, brief overview of the proposed research methodologies and the experimental setting will be presented. In Sect. 3, the application of proposed methodology and the experimental results obtained will be discussed. The last section presents conclusion and perspectives.

1 Mathematical methods

1.1 Dimensional reduction by transformation techniques

1.1.1 Auto-encoders

An auto-encoder (AE) is a neural network, which map their input to a latent representation typically of lower dimension, through nonlinear transformations, and reconstruct their input through this intermediate representation. An auto-encoder is composed of an encoder part and a decoder part. Both of which can have multiple layers for deeper AEs, and the decoding part has a layout that is symmetric to its encoding counterpart.

Consider $X = \{(x_i)\}^n_{i=1}$ being a dataset matrix with $x_i \in \mathbf{R}_D$ represents a sample from high-dimensional spaces $\mathbf{R}_D$ and n denotes the number of samples. The auto-encoder learns compressed features in a low-dimensional space L^d (with $d<D$) for high-dimensional data with minimum reconstruction loss [25]. The output of the neuron is the transformation of the input by set of layers L, which are accompanied by a weight matrix, a bias vector and a nonlinear activation function. The embedding of the dataset L^d is an output of the encoding part of the network denoted by$Z = \{(z_i)\}^n_{i=1}$ who performed by the function denotes as f, with $z_i = f(x_i), i = 1,\ldots,n$. Figure 1 shows the basic structure of Auto encoder (AE).

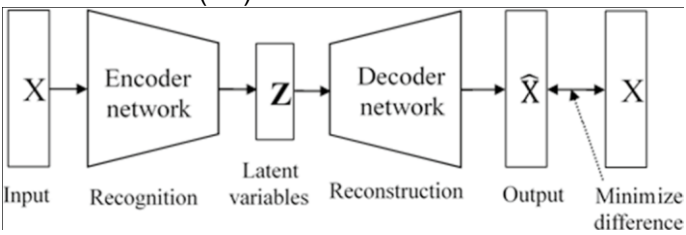

Fig. 1 Illustration of the dimensionality reduction principle with encoder and decoder

Formally, if $x_i^{(l-1)}$ denotes the i th output sample of the $(l-1)th$ layer, which is the i th input sample of the l th layer of the network, and $x_i^{(l)}$ is the ith output sample of the lth layer produces for this input, given by:

$$x_{i(l)} = f(A_{(l)} \times x_{i(l-1)} + b_{(l)}). \quad (1)$$

Where $A^{(l)}$ is the weight matrix and $b^{(l)}$ is a bias vector which accompany the lth layer, and $x_i^{(1)} \equiv x_i$. Generally, to guarantee the learned representation

adequately represents the input information, the following reconstruction error is minimized during the training phase:

$$\text{argmin}_{i} \sum_{i=1}^{n} \| x_i^{(L)} - x_i \|_2^2 . \quad th \tag{2}$$

Where $x_{i(L)}$ denotes the output of the final layer of the AE for the i input sample and can be considered as a reconstruction of that input [39].

1.1.2 Variational auto-encoders

A variational auto-encoder (VAE) is a type of generative deep learning model capable of unsupervised learning. It is a nonlinear continuous latent variable model, intended to learn a latent space, with an efficient gradient-based training procedure based on variational principles [35]. The framework of variational autoencoders (VAEs) provides a computationally efficient way for optimizing the deep latent-variable models (DLVMs) jointly with a corresponding inference model using stochastic gradient descent (SGD).

We introduce a parametric inference model $q_\phi(z/x)$ called an encoder or recognition model and $p_\theta(x/z)$ called a decoder or generative model where ϕ and θ are the variational parameters of the encoder and decoder, respectively. The goal of VAE is to find a probability distribution $q_\phi(z/x)$ of the latent variable z, which we can sample from $z \sim q_\phi(z/x)$ to generate new samples $x \sim p_\theta(x/z)$. For any choice of inference model $q_\phi(z/x)$, including the choice of variational parameters ϕ , we have [9]:

$$\log p_\phi(x) = \mathbb{E}_{q q_{\phi\phi}(z z/x x)}[[\log p_\theta(x)^{]}]$$

$$= \mathbb{E}_{(/)} \log p^{\theta}(x,z)\, p_\theta(x/z) \Big[\quad] \tag{3}$$

$$= \mathbb{E}_{q_\phi(z/x)}$$

$= L_{\theta,\phi}(x)+D_{KL}(q_\phi(z/x) \parallel p_\theta(x/z))$. The second term in equation (3) is the Kullback–Leibler (KL) divergence between $p_\theta(x/z)$ and $q_\phi(z/x)$, which is non-negative:

$$D_{KL}(q_\phi(z|x) \parallel p_\theta(x|z)) \geq 0. \tag{4}$$

The first term in equation (3) is the variational lower bound, also called the evidence lower bound (ELBO):

$$L_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x,z) - \log q_\phi(z|x)]. \tag{5}$$

Due to the non-negativity of the KL divergence, the ELBO is a lower bound on the log-likelihood of the data.

$$L_{\theta,\phi}(x) = \log p_\theta(x) - D_{KL}(q_\phi(z|x) \parallel p_\theta(x|z)) \leq \log p_\theta(x). \tag{6}$$

The loss or objective function to be maximized is given by,

$$L_{VAE}(\theta,\phi) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x) \parallel p_\theta(z)). \tag{7}$$

For two multivariate normal distributions $^{N}(\mu_0,\Sigma_0)$ and $^{N}(\mu_1,\Sigma_1)$the closed-form for the objective loss function of VAE is given by,

L_{VAE}

1 Σ^1+m

(8)

The term $\mu_1 - \mu_0\ \Sigma^2{}_{0-1}$ is a compact form for  , closely related to the squared- Mahalanobis distance if $\parallel \parallel \mu_1$ is a random variable. We then train the VAE to find the optimal parameters using Bernoulli distribution as the reconstruction loss.

$$(\phi^*,\theta^*) = \text{argmax}_{(\phi,\theta)} L_{VAE}(\theta,\phi). \tag{9}$$

1.1.3 Principal component analysis

Principal Component Analysis (PCA) is a classical method of multivariate statistics. It is a dimension-reduction tool that has been developed to reduce a large set of variables to a small set that still contains most of the pertinent information from the large set [22]. The reduced variables are called the "principal components." The principal components are calculated from the covariance matrix. If we assume that a matrix X of data has been centralized to have zero mean, the PCA requires the computation of the eigenvalues and eigenvectors of the covariance matrix, which is the product $C = \frac{1}{n-1} X \times X^T$. Since the covariance matrix is symmetric, it is diagonalizable, and the eigenvectors can be normalized such that they are orthonormal. We can decompose C into the product of three matrices:

$$C = W \times D \times W^T. \tag{10}$$

where W is a matrix of eigenvectors (each column is an eigenvector) and D is a diagonal matrix with eigenvalues λ in decreasing order on the diagonal. The eigenvectors are called principal axes or principal directions of the data. Projection of the data onto the principal axes is called principal components (PC). Figure 2 shows the Principal Component Analysis technique for Dimensionality Reduction.

1.2 U nsupervised selection variable using genetic algorithm

Genetic Algorithms (GA) are a class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection and crossover [48]. The steps of proposed GA methodology are shown in Fig. 3 These algorithms are based on the concept of natural selection of solutions by copying its main principles. Each solution may be considered a population, where each element is represented by a chromosome built with selected variables positioned as genes [16]. The GA steps reproduce various evolutionary operations, such as crossover and mutation, allowing the method to select for each generation the best chromosomes

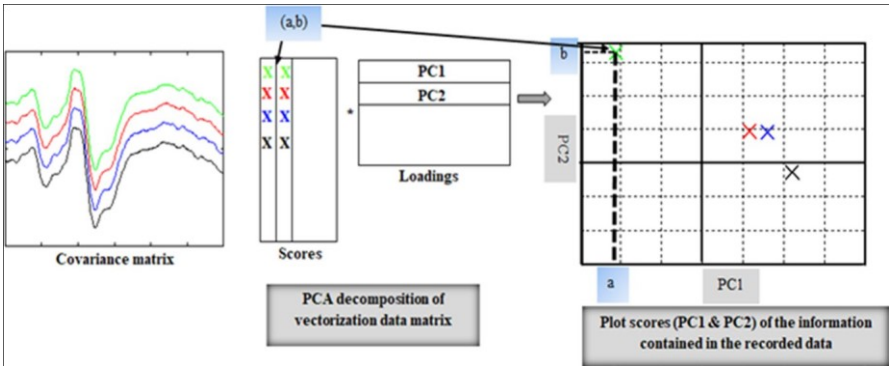

Fig. 2 Illustration of the dimensionality reduction principle with Principal Component Analysis

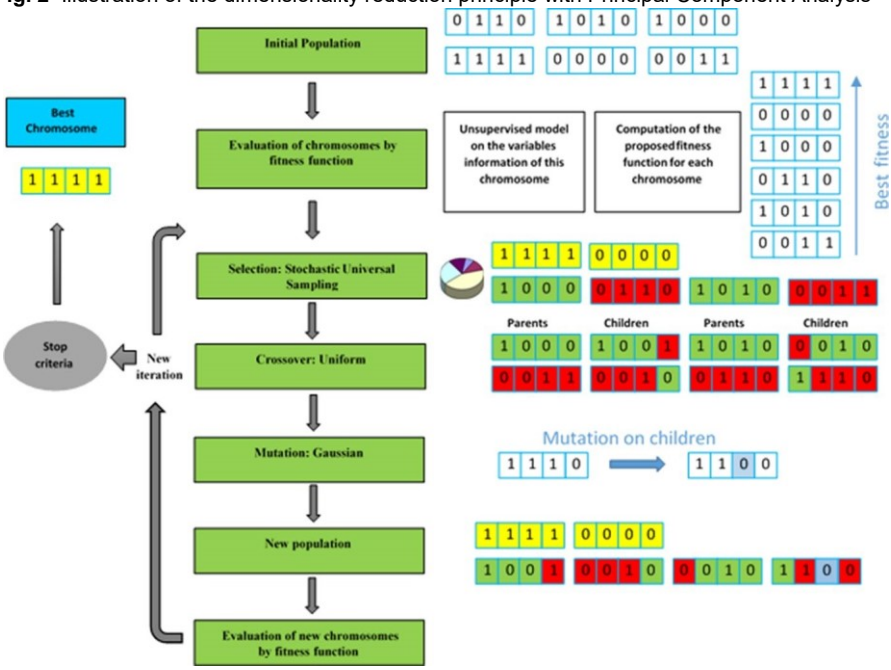

Fig. 3 Synoptic of the proposed GA methodology with binary genes. In our case, genes correspond to variables of numerical dataset

and to identify at the end an optimal chromosome with respect to an optimization criterion defined by a fitness function [28].

1.2.1 Genetic algorithm steps

Let $X = \{x_j\}^n_{j=1}$ be a matrix of numerical data of dimension $\mathbb{R}^{S\times n}$ where each column corresponds to a sample extracted from S variables $W = [w_1,\ldots,w_S]^T \in \mathbb{R}^S$. This dataset is recorded on several samples belonging to a set of classes $C=\{c_1,\ldots,c_k,\ldots,c_K\}$. The idea behind GA is the use of a population of solutions, each one represented in the form of a chromosome with selected variables positioned as genes. These algorithms are a type of evolutionary optimization computation based on the concept of natural selection of solutions in each generation for reproduction and various evolutionary operations such as crossover and mutation. The GA steps are briefly described thereafter; they are detailed in several articles, such as [16] and [28].

The initialized parameters are: the chromosome size L (the number of genes corresponding to the variable to be selected); the population size N (the number of chromosomes per generation); the number of elites N_e (the number of chromosomes with the best fitness values in the current generation that are guaranteed to survive to the next generation); and the fraction $FRAC_c$ (the number of chromosomes selected to perform crossover). The stopping parameters are the maximal number of iterations MAX_{it} , and the tolerance ϵ for the chosen fitness function. The first step of a GA is the creation of the starting population $P(0)$. N chromosomes are generated by randomly selecting L variables from W ($L<S$ is the size of the chromosomes):

$$P(0) = \{z_{i(0)} = [z_{i1}\ldots z_{il}\ \ldots z_{iL}] \quad . \quad (11)$$

The initial population $P(0)$ is selected randomly from the set of uniformly distributed variables ranging over their maximum and minimum limits [28]:

$$z_{0i} \sim U(z_{min_i}, z_{max_i}) . \quad (12)$$

where z^0_i signifies the initial l th variable of the i th population; z^{min}_i and z^{max}_i are the minimum and maximum limits of the lth decision variable; $U(z^{min}_i, z^{max}_i)$ signifies a uniform random variable ranging over $[z^{min}_i, z^{max}_i]$. Then each chromosome z_i in the current population is evaluated using a fitness function $F(.)$ that assigns a fitness value F_i:

$$F_i = F(z_i), \forall i = 1\ldots N. \quad (13)$$

The fitness function is arguably the most important part of a GA. The role of a fitness function is to measure the quality of the chromosome in the population according to the given optimization objective. Since we want to classify samples within the K unknown classes, we propose to use the K-

means or Gaussian Mixture Models Clustering methods with well-known and widely used validity indices as fitness function to evaluate the quality of the obtained clusters such as: Davies Bouldin (DB), Calinski-Harabasz (CH), Xie Beni (XB), Dunn Index (DI), Alternative Silhouette Width Criterion (ASWC) and Pakhira–Bandyopadhyay–Maulik (PBM). All of these indices measure the compactness and the separability of the clusters [41].

For each fitness function F_i, the values are ordered in ascending order and the best Ne chromosomes are selected based on this ordering. These are the surviving chromosomes that will be copied unchanged in the next population.

Once this selection has been completed, other N_p chromosomes are selected in pairs to replicate and to form $N_c = N_p$ "child" chromosomes in the next population. The selection is performed probabilistically so that an individual's selection probability is proportional to the individual's fitness. There are several schemes for the selection process: roulette-wheel selection and its extensions, scaling techniques, tournament models, elitist models, ranking, proportional selection, stochastic methods and stochastic universal sampling selection, etc. [31]. In our application, we have chosen the stochastic universal sampling selection procedure since this method is a single-phase sampling algorithm that has no deviation between the expected reproduction rate and the algorithmic sampling frequency and has a minimum spread [30]. First, we compute the probability p_i of selecting the chromosome z_i and the cumulative probability q_i:

$$p_i = \frac{F_i}{\sum_{i} F_i}. \tag{14}$$

$$q_i = \sum_{i=k} p_k. \tag{15}$$

Next, we generate a uniform random number $r \in [0, N-1]$. If $r < q_1$ then we select the first chromosome z_1, otherwise we select the chromosome z_i such that $q_{i-1} < r \leq q_i$.

Once the selection of N_p chromosomes has been completed, to form child chromosomes, we apply the crossover technique that is a structured yet randomized information exchange between chromosomes. This process allows the production of two children from two parents. In this process, genetic material from one parent is combined with genetic material from another parent to discover better children. Crossover operators can be realized by many methods, including single-point, k-point, Flat, uniform and/or order-based, discrete, and nonlinear methods. This work used a uniform crossover since it gives good results and has been used successfully in a majority of

cases [8, 19]. A uniform crossover operator combines a uniform blend of data from each parent [8], promoting greater exploration. In uniform crossover, each gene is randomly selected either from the first or from the second parent.

To explain the uniform crossover of genetic algorithms, the parent's chromosomes $p_1[z_{iq}]$, $p_2[z_{iq}]$ and the children chromosomes $o_1[z_{iq}]$, $o_2[z_{iq}]$, $q=1\cdots L$ are gene arrays. The most popular crossover variant between real numbers is the uniform crossover. Genes situated in the q position of the children chromosomes z_i are calculated as it follows [20]:

- α is a random vector of real numbers uniformly distributed with the same size as

  p_1 , p_2 , o_1 , o_2 where $\alpha_q \in [0,1]$.
- Children are copied from parents and crossover is obtained with equations 16 and 17:

$$o_1[z_i] = p_1[z_i] \text{ for each } \alpha_q > 0.5, o_1[z_{iq}] = p_2[z_{iq}]. \quad (16)$$

$$o_2[z_i] = p_2[z_i] \text{ for each } \alpha_q > 0.5, o_2[z_{iq}] = p_1[z_{iq}]. \quad (17)$$

After crossing, the mutation operator involves the modification of the value of 'gene' to maintain the genetic diversity from one generation of a population to the next. Mutation may be implemented by different methods: flip bit, random, boundary, uniform, nonuniform, or Gaussian. We have chosen here the Gaussian operator since it produced the best results for most fitness functions and gives good results [20, 21]. Essentially, this operator adds a unit Gaussian-distributed (with zero mean and unit variance) random value to the genes of each of the remaining $(N-(N_c+N_e))$ chromosomes. To implement the mutation, we add a random number generated by a Gaussian distribution with zero mean to all positions of the genes in each selected chromosome [11]. The new values of the genes, representing wavenumber positions in spectra, are then rounded to the nearest integer. The standard deviation of this distribution is the parameter called "scale" which is equal to 1 in the first generation, but this parameter is controlled during the next generations by another parameter called "shrink." The standard deviation at the *t th* generation, σ_t is the same at all coordinates of the parent chromosome, and is given by the recursive formula [11]:

$$\sigma_t = \sigma_{t-1} \times \left(1 - \text{shrink } T\_t\right). \quad (18)$$

Where T the number of generations. A low value of “shrink” produce a small decrease in the amplitude of the mutation on the indices of gene positions.

These different steps are repeated until one of the termination criteria is satisfied: the maximum number of iterations MAX $_{it}$ has been reached or the weighted-average change in the fitness values over all generations is less than a tolerance ϵ.

1.3 U nsupervised classification

1.3.1 Gaussian mixture model (GMM)

The Gaussian mixture model (GMM) is a probabilistic soft clustering algorithm used to estimate the parameters of a distribution of random variables by modeling them by a mixture density. GMM considers that each component of the mixture characterizes a class. These models have a main advantage: It is a probabilistic method for obtaining a classification of observations. A probability of belonging to each of the classes is calculated and a classification is generally obtained by assigning each of the observations to the most probable class [32]. The GMM parameters are estimated using the iterative estimation of Expectation-Maximization (EM) algorithm. Let X be a mixture random variable with a probability density function $f(x)$ defined as the weighted sum of the underlying components densities $f_k(x)$:

$$f(x/\theta) = \sum_{k=1}^{K} \pi_k f_k(x/\theta_k). \quad (19)$$

Where π_k represents the prior probability of component k satisfying [redacted] and $0 \leq \pi_k \leq 1$. K is the number of component Gaussian densities, θ and θ_k denote, respectively, the parameters of the Gaussian model. We assume that the underlying random variables are following d-dimensional Gaussian distributions. Therefore:

[redacted]

where μ_k is the d-dimensional vector mean, | | Σ_k is the covariance matrix of dimen-κ sion $d \times d$, Σ_k denotes the determinant of Σ_k and $\theta = \{\mu_k, \Sigma_k, \pi_k\}(k = 1)$ represents the parameters of GMM to be estimated. The Expectation-Maximization (EM) | | algorithm is the most widely used method for estimating

the parameters of a mixture model. It is an iterative technique maximizing the likelihood of the parameters of probabilistic models in the presence of latent (unobservable) variables. Given the sample $X = \{x_1, \cdots, x_n\}$, the corresponding GMM likelihood relative to the model with parameter θ is given by:

$$f(X/\theta) = \prod_{i=1}^{n} f(x_i/\theta). \tag{21}$$

It is easier to maximize the log-likelihood instead of the likelihood function itself. The log-likelihood function is given by:

$$\log(f(X/\theta)) = \sum_{i=1}^{n} \log(f(x_i/\theta)). \tag{22}$$

Suppose that with each sample x, a latent variable $z \in \mathbb{R}^K$ is associated. The loglikelihood function of X by introducing the latent variables Z takes the following form:

$$\begin{aligned} \log(f(X/\theta)) &= \sum_{z} \log(f(X, Z/\theta)) \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \{\log(\pi_k) + \log(\mathrm{N}(x_i/\mu_k, \Sigma_k))\}. \end{aligned} \tag{23}$$

Where z_{ik} denotes the kth component of z_i . To estimate the parameters of a Gaussian mixture, the expectation-maximization (EM) algorithm iterates between the following two steps until convergence: the Estimation step and the Maximization step.

Estimation step: the current parameters are used to calculate the expectation of z_{ik} , denoted by $\gamma(z_{ik})$, as follows:

$$\pi_k N(x_i/\mu_k,\Sigma_k)\ \gamma(z_{ik}) = \blacksquare. \quad (24)$$

Maximization step: after calculating the expectations of z_{ik} in previous step, the quantity $Q(\theta,\theta^{\text{old}})$ must be maximized with respect to the different parameters of the model.

$$\theta^{\text{new}} = \text{argmax}_{(\theta)} \quad Q(\theta,\theta^{\text{old}})\blacksquare X,\theta^{\text{old}})\log(f(X,Z/\theta^{\text{old}})). \quad (25)$$

The new estimated parameters θ^{new} are given by:

- The proportions: π_k^{new} $\blacksquare_n$
- The averages: $\mu_k$new $\blacksquare_{\sum_n \gamma(z)x}$ new new
- The covariance matrices: $\blacksquare_k$ $\gamma(z_{ik})(x_i-\mu_k)(x_i-\mu_k)_t$

$\blacksquare$

1.3.2 K-means clustering

K-method is one of the plainest alone unsupervised classification algorithms disposed partitioning a dataset $X=\{x_i/x_i\in\mathbb{R}^d\}^n_{i=1}$ of n samples collected from d variables into K disjoint partition or clusters, $P(X,K) = \{P_k\}^K_{k=1}$ expressed by their centroids, $C=\{c_k : c_k\in\mathbb{R}^d\}^K_{k=1}$. These clusters are estimated by minimizing thetotal
intra-cluster variation defined as [42]:

$$D(C) = I(x_i \in P_k)\|x_i - c_k\|_2. \quad (26)$$

$i=1\ k=1$

where I is the indicator function defined as:

$n \quad K$

$$\{ \quad 1, \text{ if } x^i \in P^k \quad . \quad (27)$$
$$I(x_i \in P_k) = 0, \text{ otherwise}$$

and $x_j - c_k^2$ the Euclidean distance between the i th sample x_i and the k^{th} center c_k . The K-means objective function ‖ ‖ 26 is optimized by the following iterative algorithm [42]:

• Step 1: Initialize the K cluster centers C by choosing randomly K different samples of X. • Step 2: For each $i = \{1,\ldots,n\}$, attribute the i th sample x_i to the m th cluster P_m such that

$$m = {}_k\text{argmin}_{=\{1,\ldots,K\}} \| x_i - c_k \|^2 \qquad {}_k$$

• Step 3: For each $k=\{1,\ldots,K\}$ update the k th cluster centroid c using:

$$\frac{\sum_{x_i \in P_k} x_i}{=N_k} c_k \quad . \qquad (28)$$

where N_k is the number of samples belonging to the k^{th} cluster P_k

• Step 4: Repeat steps 1 to 3, until the algorithm converges.

1.3.3 Partition assessment by validity index

A validity index is a mathematical function that can be used to compare numerous partitions predicted by the same clustering algorithm using different parameter values, such as the number of clusters. The majority of existing validity indices were defined as a function of compactness and separation measures of clusters, where compactness (or within-cluster distance) indicates the closeness of objects in the same cluster, and separation (or between-clusters distance) indicates the isolation of one cluster from another. The optimal partition maximizes cluster separation while minimizing cluster compactness. The Calinski-Harabasz (CH) [6], Dunn index (DI) [13], Davies-Bouldin (DB) [10], Pakhira–Bandyopadhyay–Maulik (PBM) [29], XieBeni (XB) [47], and Alternative Silhouette Width Criterion (ASWC) [34] are the most common validity indices described in the literature. These validity indices are briefly presented, using the mathematical notations, as follows:

Calinski–Harabasz [6]. The ratio of inter and intra cluster variances is used to calculate this index:

$$CH(P(X,K)) = W \cdot \frac{B}{} \quad (29)$$

where $B = \sum_{k}^{N} \| - c \|_2$ is the inter (or between) cluster variance, $W = \sum_{n-k} \| x_i - c_j \|_2$ is the intra (or within) cluster variance, N_j is the number of samples belonging to the *jth* cluster, and $c = \sum_{n} x_i$ is the dataset's barycenter. The best partition maximizes the CH index.

Dunn index [13]. This indicator is defined as the rate between the minimum intercluster distance and the largest cluster perimeter:

$$DI(P(X,K)) = \frac{\min_{p,q \in \{1,\ldots,K\}, p \neq q} \{\delta(P_p, P_q)\}}{\max_{l \in \{1,\cdots,K\}} \{\Delta(P_l)\}} \quad . \quad (30)$$

where $\delta(P_p, P_q) = \min_{x_i \in P_p, x_r \in P_q} \{\| x_i - x_r \|\}$ is the closet distance between two samples across clusters p and q, and $\Delta(P) = \max \{ x - x \}$ is the maximum distance between two samples in the l cluster. The best partition maximizes the DI.

Davies–Bouldin [10] This index is defined as a function of the ratio of the sum of within-cluster scatter to between cluster separations. When it has a small value it exhibits a good clustering:

$$DB(k) = \frac{}{} \frac{1}{k} \sum_{l=k1} R_l. \quad (31)$$

where $R_i = \max_{l \neq m, 1 \leq l, m \leq k} \frac{SM_l + l, S_m}{m}$. The term $\frac{}{} {}_{l,m}S_j = \frac{1}{N_j} \sum_{pN=j} 1 \| x_p - c_j \|_2$ represents the dispersion measure of the jth cluster. The M is the Euclidean distance between the centroids of l_{th} and mth clusters.

Pakhira–Bandyopadhyay–Maulik [29] This index is defined as the square ratio between the largest normalized inter-cluster distance D_N and the normalized sum of
intracluster distances E_N:

$$\text{PBM}(k) = \left(\frac{1}{K} \cdot \frac{D}{E}\right)^2 \quad (32)$$

where

$$D_N = \max_{l,m=1,\ldots,k} \|c_l - c_m\| \quad (33)$$

and

$$E_N = \sum_{i=1}^{n} \|x_i - \bar{x}\| \quad (34)$$

and $\bar{x}$ is the mean of dataset. The best partition maximizes the PBM index.

Xie–Beni [47] This index is defined as the ratio between the compactness π and the separation S of clusters. Mathematically, the XB index may be formulated as:

$$\text{XB}(k) = \frac{\pi}{S} \quad (35)$$

where $\pi = \frac{1}{n}\sum_{i=1}^{n} \|x_i - c_i\|^2$ the average compactness of the clusters, and

$S = \min_{l \neq m; 1 \leq l,m \leq k} \{\|c_l - c_m\|^2\}$ is the minimum separation measure between the clusters. K is defined as the number of clusters which minimizes the XB index.

Alternative Silhouette Width Criterion [34]. It is a variant of the original silhouette criterion obtained by considering the average silhouette of all points of the dataset:

$$ASWC(k) = N1 \sum_{i=}^{k}{}_{1} \; - \; \sum_{j}^{N}{}_{=i1} s(x_{i(j)}) \; - \; \overline{\qquad} = 1 \sum_{i=}^{k}{}_{1} \sum_{j}^{N}{}_{=i1} \; {}_{i(}b_{j)}{}^{i(j)} \; - \; . \qquad (36)$$

$$N \qquad a + \epsilon$$

The term $a_{i(j)} = {}_{N1i} \sum_{Nl=i1,li\neq(jj)} \| x_{l(j)} - x_{i(j)} \|_2$ is the mean distance of this ${}_{i(j)\ m=1\ldots k\ m\neq k}{}^{1}$

${}_{Nm-6}\, jth$ object to all 2 other samples in the i th cluster. And $\overline{\epsilon}$ is a small constant

(e.g. 10) used to avoid division by zero when $a = 0$. The term $b = \min_{opt;\ Ni}$

$$l \sum = 1 \; x_{i(j)} - x_{m(j)}$$

is the average distance of $x_{i(j)}$ to its nearest cluster. K is defined as the number of clusters which maximizes the ASWC index.

1.4 R esampling method: bootstrappingn

The general principle of the Bootstrap method is to resample the initial samples a large number of times, the statistical inference being based on the results of the samples obtained. Moreover, the Bootstrap resampling method is the most commonly used in most applications because it considers that all observations $X_1,\cdots,X_n$ are independent and identically distributed [14]. Suppose we are looking for information on a parameter θ of the population. A random selection of the population of a sample of size n is carried out, denoted by $(X_1,\cdots,X_n)$. Then, we calculate an estimator $\hat{\theta}$ of θ which is based on a drawn sample and therefore its numerical value varies according to different random draws. For this, we force to calculate the bootstrap standard error of $\hat{\theta}$ which is the following:

$$\text{SEB}\ (\theta) = \hat{\theta}^{b)})^2. \qquad (37)$$

Then pick a large number of bootstrap draws T, and repeat for $b=1\cdots T$. In our application, we consider the set of preprocessed data, $x_1,\cdots,x_n$, which belong to the classes $c_1,\cdots c_j,\cdots c_n$, with $c_j \in \{1\cdots K\}$ and K the number of classes known a priori. We realize T random draws in this dataset. For each new draw, $t \in \{1\cdots T\}$ we obtain a new set of n samples: $x_1(t(1)),\cdots,x_n(t(1))$ which belong to the classes $c_1,\cdots c_j,\cdots c_n$. The GMM (or K-means) classification is applied to this resampled set to estimate K classes. The result obtained gives us the membership of each observation to the classes $\hat{c}_1,\cdots \hat{c}_J,\cdots \hat{c}_n$. For each random draw, we measure the confidence interval and the number of well-classified samples, which allows us to calculate the average of good classifications over all T iterations.

$E_{moy} =$ [redacted] _ _

| .(38)

The combination of the resampling method with the classification method allows to obtain a more precise estimation of the classifications of the samples and limits the influence of the random initialization in the unsupervised classification algorithm (GMM or K-means).

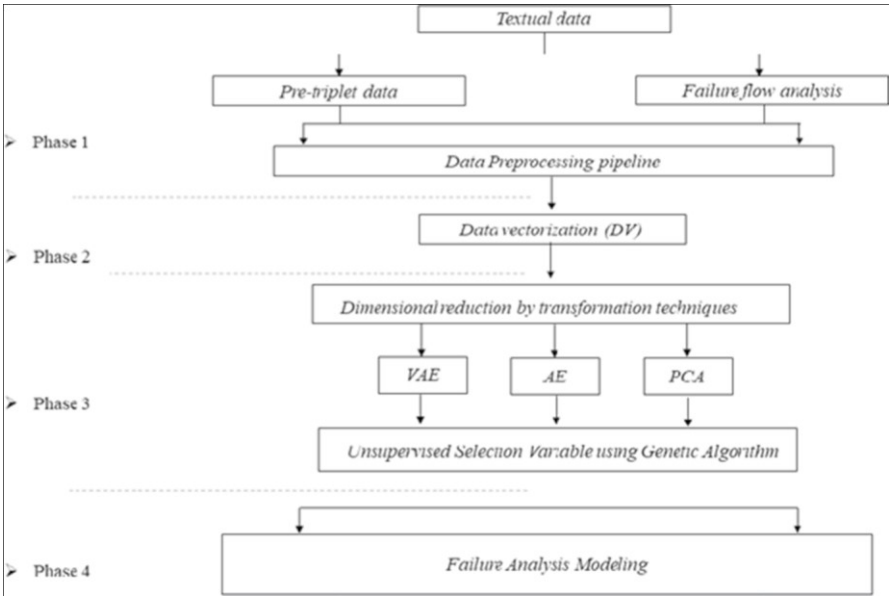

Fig. 4 Synoptic representation of the proposed methodology for the failure analysis of textual dataset

1.5 Proposed methodology for evaluation of failure analysis discrimination

In this section, we present the methodology proposed for the selection of variables by the genetic algorithm combined with K-means or GMM clustering and the validity index applied to textual data. Figure 4 shows the steps of the failure analysis modeling methodology by reducing the dimension using transformation and unsupervised variable selection techniques and representing the failure classes for this analyzed data.

The first phase represents the pipeline preprocessing of the textual data. The data vectorization using the Word2Vec method is applied in phase 2 to convert our textual data into numerical dataset [33]. These are a set of neural network models that have the aim to represent words in the vector space. These models are highly efficient and performant in understanding the context and relation between words.

The Phase 3 consists of two parts: the first is based on the application of dimensional reduction techniques on numerical data such as Auto-Encoders (AE), Variational Auto-Encoders (VAE) or Principal Component Analysis (PCA) methods to transform the original dataset X in order to obtain a new p-dimensional latent space X .Then, we apply on these reduced data the unsupervised variable selection method using the genetic algorithm combined with the clustering method such as K-means or GMM evaluated by different fitness functions as validity indices to identify discriminant variables. This process allows us to obtain an L-dimensional dataset instead of p-dimensional with $L<p<d$.

For representation purposes, Phase 4 shows the application of principal component analysis (PCA) on the final reduced space of variables. Useful information from the PCA is then represented in the form of scores (PC1 & PC2). The representation of plot scores in terms of PC1 and PC2 gives a qualitative indication of the separation process. Finally, to quantify the separability and compactness with respect to text clusters, we compute the validity indices DI, DB and CH on the L-dimensional dataset.

2 Background

2.1 Data formalization

Data description and analysis is an essential step that precedes modeling process of defected microelectronic samples. An accurate representation of the data is necessary to understand the failure analysis flow using one or more triplets (Step type, Sub-step technique, Equipment) proposed by experts. We have a set of textual data on microelectronic production failure analysis. This dataset consists of two parts [15]: the first is the description of failure $X^{\neg\lambda}$ and the second is the analysis flow paths of failure λ. In formalizing

our approach, we use the following notations. Let $X = \{x_{ij}\}^{n}_{i=',m}{}_{1,j=1}$ represents the input space of given dataset where n is the number of instances and m is the number of variable span. We define the description of failure $X^{\neg\lambda} = \{x_{ij}{}^{\neg\lambda}\}^{n}_{i=',d}{}_{1,j=1}$ as pre-triplet data, where $d \leq m$. This indicates on demand the

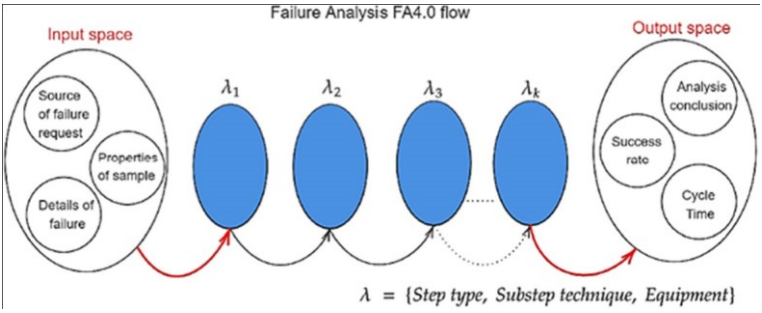

Fig. 5 Expert decision failure analysis process

information given before an expert proposes the analysis to determine the failure mode (FM). Figure 5 represents an illustration of expert decision of failure analysis.

Let $\Lambda = \{\lambda_{ij}\}^{n}_{i=',q}{}_{1,j=1} \in \mathbb{R}^{n \times q}$ represents the Failure analysis decision path recorded by expert per sample. It includes the whole process of diagnostics stacked horizontally [15]: Equipment, Sub-step technique, Step type. The purpose of analyzing faults is to find the source of the failure using one or more triplets (Step type, Substep technique, Equipment). Then the input space X can be described using a matrix representing a joint observation space of $x_{ij}{}^{\neg\lambda}$ and λ_{ij}:

$$\begin{vmatrix} x^{\neg\lambda} \cdots x^{\neg\lambda} \lambda \cdots \lambda_q x \\ \left(\blacksquare \right) \\ \cdots \lambda \lambda \cdots 2 \end{vmatrix}$$

$$X = \begin{vmatrix} \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \end{vmatrix}^{21} \qquad 2 \qquad q$$

$$\begin{vmatrix} \\ \\ x_{n\neg 1\lambda} \cdots x_{nd\neg\lambda} \lambda_{n1} \cdots \lambda_{nq} \end{vmatrix}$$

2.2 P reprocessing pipeline

Preprocessing text includes eliminating noise by removing whitespace and punctuation, correcting spelling errors, deleting duplicate instances, converting text to lowercase, and removing stop words and words with less than three letters. Here, we will go over the phases in the preprocessing pipeline [15]:

- Symbol and alphanumeric removal: This technique eliminates words from the document that do not contribute to the intelligent pattern or the sample of analysis, such as symbols and sometimes-alphanumeric words. They are just stop words and inflexions that are employed to highlight meaning, hence their removal [5].
- Tokenization and Thresholding: Tokenization is the process of utilizing a separator to convert or break a sentence into a token. Thresholding is a method for removing words that are shorter than a specific length [5]. The threshold in this paper is set at two.
- Stemmatization and Lemmatization: this is the process of removing affixes (prefixes and suffixes) from textual features [15].
- Abbreviation: abbreviations are common in FRACAS, thus, it is important to replace them with their full meaning. To help with this, we have put up an abbreviation dictionary.

2.3 Vectorization data

The technique of translating text into numerical form is known as text vectorization. Vectorization can be done in a variety of ways, the most common of which are Bag of Words, TF-IDF (Term Frequency–Inverse Document Frequency), Word2Vec, and GloVe. The Word2Vec method was chosen because it is an effective approach for converting text to numerical data in the context of FRACAS [15]. Word2vec is built on the deep learning concept of a fixed-length real-valued vector representing words. Word2vec is a prediction-based approach that comes in two flavors [49]: continuous bag-of-words (CBOW) and skip-gram (SG). Small neural networks are used in both the CBOW and SG models to learn the mapping of words to a vector space point. Therefore, each word represents a vector. Finally, we construct a high dimension matrix. Each row in matrix represents every training instance and the columns are the generated word vectors.

3 A pplication & results

First, we propose to apply our VAE-GA methodology on real data such as the E-Mail dataset of raw textual email messages [R50]. Then, we compare this

proposed methodology with other methods such as VAE, GA, AE, PCA, and the combination between AE or PCA with GA.

3.1 Real dataset: E-mail-classification (spam/no spam)

The considered dataset plays a crucial role in assessing the performance of any spam filter.The E-Mail dataset of raw textual email messages is mainely used to divide emails into two categories Spam (122 emails) and No spam (835 emails) [43].

The necessary stages that must be checked during the data mining of email messages are summarized into the following:

NLP preprocessing : This is the first stage that is executed whenever an incoming mail is received. This step consists of tokenization, stemming, noise removal and stop word removal.

Text Vectorization : This is the second stage that is the process of converting text into numerical representation. We have chosen the Word2Vec since this method is an efficient method for transforming text to numerical data.

Table 1 Performance of VAE, GA, AE, PCA, and the combination between VAE, AE and PCA with GA evaluated based on K-means or GMM clustering methods applied on the E-Mail textual dataset

Methods	Clustering	Accuracy (%)
VAE	K means	60.8
VAE	GMM	**77.5**
GA	K means	66.7
GA	GMM	**72.4**
PCA	K means	60.8
PCA	GMM	65
AE	K means	58.6
AE	GMM	65.8
VAE- GA	K means	68.4
VAE- GA	GMM	**80.7**
AE- GA	K means	64.5
AE - GA	GMM	71.4
PCA- GA	K means	62.3
PCA- GA	GMM	68.4

The Bold in the tables means the best results

After these stages, we test the performance of the methods proposed above on this dataset by calculating the precision of correct classifications. Table 1 shows the results of the application of VAE, GA, AE, PCA, and the combination between VAE, AE, and PCA with GA evaluated based on K-means or GMM clustering methods.

We can see that the three methods VAE-GA, GA and VAE give the best GMM clustering accuracy. The VAE-GA method gives the best performance

with a GMM clustering accuracy percentage of 80.7% . After that, we propose to apply Bootstrap resampling method on these three methods with T=1000 draws and calculate the mean of correct classification and confidence interval (CI) of GMM clustering. Table 2 shows that our proposed VAE-GA methodology yields better and much stable GMM clustering results than applying dimension reduction method like VAE and variable selection method like GA separately.

3.2 F ailure analysis dataset

STMicroelectronics provided the dataset for this study, covering a three-year period from 2019 to 2021. The data have the same format as defined in the FRACAS

Table 2 Average accuracy and confidence interval of GMM clustering for three methods: VAE-GA, GA and VAE after applying Bootstrap resampling technique to textual data observations

Methods	CI	Mean (%)
VAE	[68.5 80.8]	74.3
GA	[67.1 76.4]	71.5
VAE - GA	**[77.8 81.5]**	**79.8**

The Bold in the tables means the best results

system. Following the previous methodology, we transform the data from vertical stacking to horizontal stacking. i.e., each failure description (objective, context, etc.) followed by the whole path of triplet analysis is considered as one observation and represented on the same row in the dataset. The data are reduced to 12033 observations after the transformation and filtration, and we preserve 19 preprocessed variables excluding dates. We vectorize these data using Word2vec technique after acquiring clean processed data based on the pre processing pipeline. The Gensim's Word2vec settings are kept the same, with the exception that the vocabulary size is set to 1000 and the minimum word count is set to three. Finally, a matrix of dimensions 12033×1000 is obtained with 50.7% positively skewed features and 49% negatively skewed features.

3.3 C hoice of the parameters for genetic algorithm

A critical phase of GA is the right choice of its parameters, presented in Subsection 2.2, in order to ensure the convergence of the algorithm to the optimal solution.

The initial parameters have been fixed as follows: the maximum number of generations T=100 , the fraction of crossover F_c = 0.8 , the elites number N_e =2 , the tolerance ϵ =10^{-6} . These values have been used for several implementation of GA since they give good results [3]. In this paper, a grid research was used to choose the GA optimal parameters from the following

sets of possibilities: the size chromosome $L \in \{10,20,\cdots,100\}$, the population size $N \in \{100,200,\cdots,500\}$, and the number of cluster $K \in \{2,3,\cdots,20\}$. The fitness function can be handled with K-means or GMM Clustering methods evaluated by the validity indices commonly used to quantify class separability of failure analysis data in microelectronics field.

For each setting of the parameters, the variability of GA with different fitness functions is evaluated using the validity indices such as DI, DB and CH on the PC score plots obtained by PCA method. Table 3 displays the results of optimizing GA parameters: fitness function, number of clusters and number of variables. We find

Table 3 The values of validity indices DB, DI and CH for subdata matrix to variables selected by different fitness function of GA algorithm. A higher value of DI and CH and lower value of DB means better discrimination within the group of failure analysis data

Fitness function of GA	DI	DB	CH	*L* & *K*
K-means with DB	0.70	0.68	12350	*L*=20 & *K*=16
GMM with DB	**0.75**	**0.67**	**13340**	*L*=**20** & *K*=**16**
K-means with DI	0.42	1.03	4421	*L*=20 & *K*=17
GMM with DI	0.48	0.98	4854	*L*=20 & *K*=17
K-means with XB	0.68	0.77	10320	*L*=20 & *K*=16
GMM with XB	0.72	0.74	11810	*L*=20 & *K*=16
K-means with CH	0.55	0.78	11850	*L*=20 & *K*=17
GMM with CH	0.65	0.71	12240	*L*=20 & *K*=17
K-means with ASWC	0.59	0.77	10580	*L*=20 & *K*=17
GMM with ASWC	0.62	0.76	10780	*L*=20 & *K*=17
K-means with PBM	0.70	0.68	12670	*L*=20 & *K*=16
GMM with PBM	0.68	0.67	12560	*L*=20 & *K*=16

The Bold in the tables means the best results

that K means-DB or GMM-DB fitness function with L=20 and K=16 give the best values of the validity indices DI, DB and CH for the both application of GA combined with k-means or GMM clustering. These values, presented in Table 3, confirm that the GA with GMM-DB fitness function allows a better clustering of the analyzed observation of textual data compared to other GA algorithms based on other clustering techniques. Note that the application GA algorithms with K meansASWC, K means-CH, K means-XB, GMM-PBM and GMM-XB as fitness function give quite good discriminating results among the clusters of failure analysis data.

3.4 R esults and discussion of the proposed methodology for discriminating failure analysis

After selecting the best parameters of the GA algorithm, we propose to apply our methodology of combination between the VAE dimension reduction method and the GA variable selection method. The objective of this

methodology VAE-GA is to reduce the dimension of textual data by the VAE method, which projects the

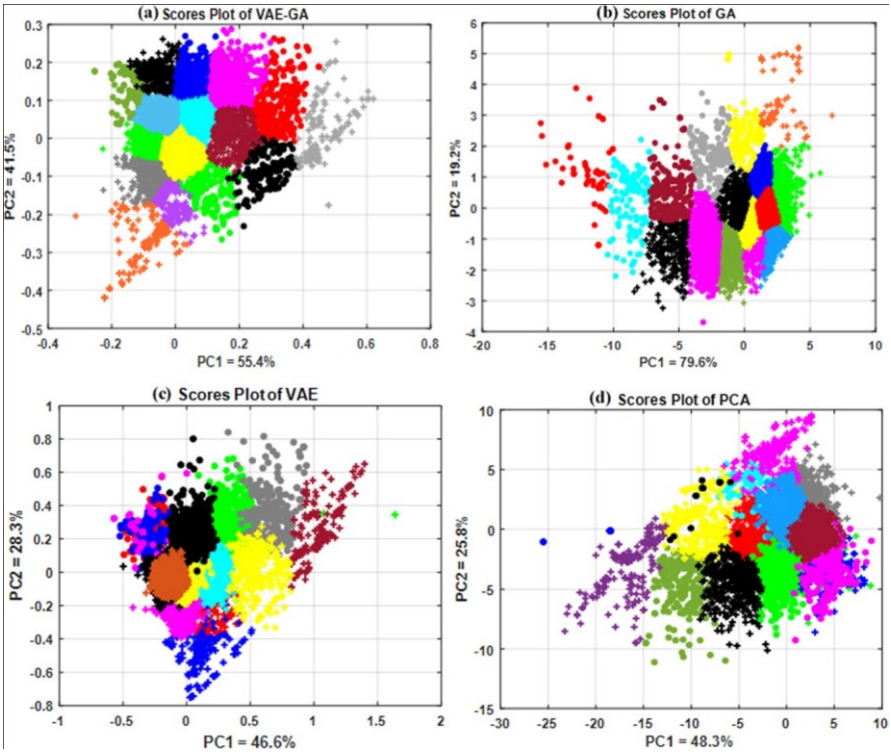

Fig. 6 Score plots showing the discrimination according to the clusters of textuals data set in terms of PC1 vs. PC2. PCA was applied on **a** the numerical sub-data matrix of the latent variables selected by the VAE-GA, **b** the numerical sub-data matrix of the variables selected by the GA, **c** the numerical sub-data matrix of the latent variables extracted by the VAE, **d** the numerical data matrix over all variables

Table 4 The values of validity indices DB, DI and CH applied on sub-data matrices of latent variables selected by the VAE-GA, the GA, the VAE and the full data, respectively. A higher value of DI and CH and lower value of DB means better discrimination within the group of failure analysis data

Methods	DI	DB	CH
VAE	0.84	0.65	14720
GA	0.75	0.67	13340
VAE - GA	**1.22**	**0.58**	**17080**
Over all variablesl	0.11	1.62	10300

The Bold in the tables means the best results

Table 5 The values of validity indices DB, DI and CH for sub-data matrix of variables selected by the VAE-GA, AE-GA, and PCA-GA techniques, respectively. A higher value of DI and CH and lower value of DB means better discrimination within the group of failure analysis data

Methods	DI	DB	CH
PCA-GA	0.78	0.66	14250
AE-GA	0.55	0.78	10520
VAE - GA	**1.22**	**0.58**	**17080**

The Bold in the tables means the best results

data on a new latent space, and then selects the discriminant latent variables in an unsupervised way by the GA method. Figure 6 shows the PC score plots for the analysis of the discrimination of failure analysis data. These figures show that the use of latent variables selected by VAE-GA (Fig. 6a) allows better discrimination and good representation of failure analysis data than the use of variables selected by GA solely (Fig. 6b) or the latent variables extracted by the VAE solely (Fig. 6c) or using the all-entire variables (Fig. 6d). The result of the embedding space visualization is a confirmation of the good disentanglement of VAE-GA latent space.

To confirm Fig. 6 results, we calculate the validity indices as metrics in order to quantify the results of compactness and separation between clusters of failure analysis data. Table 4 shows the DB, CH and DI validity indices scores obtained on the numerical sub-data matrix for the latent variables selected by the VAE-GA, the numerical subdata matrix for the variables selected by the GA, the numerical sub-data matrix for the latent variables extracted by the VAE, and the numerical data matrix over all variables. These values confirm that the VAE-GA provides better GMM clustering of failure analysis data compared to applying the GA and AVE methods separately.

To complete this study, we propose to compare our proposed VAE-GA methodology with other combination techniques of variable selection and dimension reduction such as principal component analysis with genetic algorithm (PCA-GA) and Autoencoders with genetic algorithm (AE-GA). Table 5 shows the DB, CH and DI validity indices scores obtained on the numerical sub-data matrix for the variables selected by the VAE-GA, PCA-GA and AE-GA techniques, respectively. These values confirm that the VAE-GA provides better GMM clustering of failure analysis data compared to applying the PCA-GA and AE-GA methods.

Now, as we got a good representation of our decision space one can define families of our decision space and select one or more representatives of each class, which may be used later for further predictive and inferential studies.

4 C onclusion & perspective

First, we proposed a methodology based on the association of a genetic algorithm with a k-means or GMM clustering evaluated by different fitness functions for the identification of discriminating variables for the study of textual data of failure analysis in microelectronics field. We have proposed to

apply a preprocessing pipeline to handle the different representation of the textual data and convert it to numerical data using Word2vec algorithm.

Based on several metrics used to quantify the separation and compactness between clusters of failure analysis data, we showed that the variables selected by the GA using GMM clustering and Davies Bouldin provide the best discrimination of the textual data compared to other combinations of variables selection algorithm. Secondly, we proposed a new algorithm, denoted by VAE-GA, combining a dimensional reduction technique using variational autoencoders (VAE) and a variable selection technique based on genetic algorithm. The PCA score plots on latent variables selected by VAE-GA allow better discrimination and good representation between clusters of failure analysis data than using GA and VAE separately or even using the whole set of original data. Thirdly, the variables selected by our VAE-GA methodology are also giving better representation of the space than the variables selected by the combinations PCA-GA and
AE-GA.

A first perspective of this work is to validate our results and optimize the model by exchanging with experts from STM. A second perspective is to extract representatives of the obtained clusters and associate them with a certain probability distribution for further predictive and inferential analysis about the decision space.

Acknowledgements This study was carried out by Mines Saint-Etienne in partnership with STMicroelectronics Reliability and Failure Analysis Lab in Grenoble, France.

Author contributions All named authors contributed equally to the construction of the paper. A.R. designed the structure of this article and managed to run the new algorithms and interpreted the results. A.H. and M.B. contributed in the explanation of mathematical methods and discussion of the results. He also reviewed the article for faults and added some other explanations and also revised the manuscript for linguistic check and some other explanations. K.E. were responsible of the data collection and illustration part. They gathered data from different sources and check for its reliability. Authors have read and agreed to the published version of the manuscript.

Funding This project has been funded with the support of european project FA4.0.

Data availibility All data, models, and code generated or used during the study appear in the submitted article and are provided upon request by contacting Abbas Rammal via email: abbas.rammal@emse.fr.

Declarations

Conflict of interest The authors declare that there are no competing interests.
Ethical approval Not applicable.

Consent to participate Not applicable.

Consent to publish Not applicable.

References

1. Abualigah L, Khader A, AlBetar M (2016) Unsupervised feature selection technique based on genetic algorithm for improving the text clustering, 2005. In: Paper Presented at the 7th International Conference on Computer Science and Information Technology, pp 13–14
2. Ani A (2005) Ant colony optimization for feature subset selection. Trans Eng Comput Technol 4:35–389
3. Ayad A (2013) Parametric analysis for genetic algorithms handling parameters. Alex Eng J 52:99–111
4. Bazu M, Bajenescu T (2011) A practical guide for manufacturers of electronic components and systems. failure analysis: a practical guide for manufacturers of electronic components and systems. Chennai, John Wiley and Sons
5. Bharti K, Singh P (2015) Hybrid dimension reduction by integrating feature selection with feature extraction method for text clustering. Expert Syst Appl 42:3105–3114
6. Calinski T, Harabasz J (1974) A dendrite method for cluster analysis. Commun Stat theor Methods 3:1–27
7. Centner V, Massart D, de Noord O, de Jong S, Vandeginste B, Sterna C (1996) Elimination of uninformative variables for multivariate calibration. Anal Chem 68(21):3851–3858
8. Chawdhry P, Roy R, Pant R (2012) Soft computing in engineering design and manufacturing. Springer, Berlin, Heidelberg
9. Dai B, Wipf D (2019) Diagnosing and enhancing vae models, 2019. Paper Presented at the International Conference on Learning Representations arXiv: 1903. 05789
10. Davies D, Boldin D (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intel 2:224–227
11. Deep K, Thakury M (2007) A new mutation operator for real coded genetic algorithms. Appl Math Comput 193:211–230
12. Derksen S, Keselman H (1992) Backward forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. Br J Math Stat Psychol 45(2):265–282
13. Dunn J (1974) Well-separated clusters and optimal fuzzy partitions. J Cybernet 4:95–104
14. Efron B (1979) Bootstrap methods: another look at the jackknife. Ann Stat 7:1–26
15. Ezukwoke K, Toubakh H, Hoayek A, Batton-Hubert M, Boucher X, Gounet P (2021) Intelligent fault analysis decision flow in semiconductor industry 4.0 using natural language processing with deep clustering, 2021. In: Paper Presented at the 17rd International Conference on Automation Science and Engineering, pp 23–27
16. Forrest S (1993) Genetic algorithms: principles of natural selection applied to computations. Science 261:872–878
17. Galvão R, Araújo M, Fragoso W, Silva E, José G, Soares S, Paiva H (2008) A variable elimination method to improve the parsimony of mlr models using the successive projections algorithm. Chemom Intell Lab Syst 92(1):83–91
18. Güney A, Bozdogan H, Arslan O (2021) Robust model selection in linear regression models using information complexity. J Comput Appl Math 398:113679
19. Gonçalves J, Mendes M, Resende M (2005) A hybrid genetic algorithm for the job shop scheduling problem. Eur J Oper Res 167:77–953
20. Hinterding R (1995) Gaussian mutation and self-adaption for numeric genetic algorithms, 1995. In: Paper Presented at the IEEE International Conference on Evolutionary Computation
21. Hinterding R, Michalewicz Z, Peachey T (1996) Self-adaptive genetic algorithm for numeric functions. Parallel Probl Solv Nat 1141:420–429
22. Jolliffe I (2002) Principal component analysis. Springer, Berlin, Heidelberg
23. Lee JH, Chan S, Jang JS (2010) Process-oriented development of failure reporting, analysis, and corrective action system. J Qual Reliab Eng 2010:8
24. Liu L, Kang J, Yu J, Wang Z (2005) A comparative study on unsupervised feature selection methods for text clustering, 2005. In: Paper Presented at the International Conference on Natural Language Processing and Knowledge Engineering, pp 30–31

25. Lore K, Akintayo A, Sarkar S (2017) A deep autoencoder approach to natural low-light image enhancement. Pattern Recognit 61:650–662
26. Mehmood T, Liland K, Snipen L, Sæbog S (2012) A review of variable selection methods in partial least squares regression. Chemom Intel Lab Syst 118:62–69
27. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Burges CJ, Bottou L, Welling M, Ghahramani Z, Weinberger KQ (eds) An overview and empirical comparison of natural language processing (NLP) models and an introduction to and empirical application of autoencoder models in marketing. Curran Associates Inc
28. Mitchell M (1995) Genetic algorithms: an overview. Complexity 1:31–39
29. Pakhira M, Bandyopadhyay S, Maulik U (2004) Validity index for crisp and fuzzy clusters. Pattern Recognit 37:487–501
30. Picek S, Goluba M (2010) Comparison of a crossover operator in binary-coded genetic algorithms. WSEAS Trans Comput 9:1064–1073
31. Ranjini A, Zoraida B (2013) Analysis of selection schemes for solving job shop scheduling problem using genetic algorithm. Int J Res Eng 2:775–779
32. Reynolds D (2009) Gaussian mixture models. Springer, Boston, US
33. Rong X (2014) word2vec parameter learning explained. *arXiv e-prints*
34. Rousseeuw P (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53–65
35. San Martin G, López Droguett E, Meruane V, das Chagas Moura M (2019) Deep variational autoencoders: a promising tool for dimensionality reduction and ball bearing elements fault diagnosis. Struct Health Monitor 18:1092–1128
36. Shamsinejadbabki P, Saraee M (2012) A new unsupervised feature selection method for text clustering based on genetic algorithms. J Intel Inf Syst 38:1–16
37. Shankar V, Parsana S (2022) An overview and empirical comparison of natural language processing (nlp) models and an introduction to and empirical application of autoencoder models in marketing. J Acad Mark Sci 50(6):1324–1350
38. Sivanandam S, Deepa S (2008) Introduction to genetic algorithms. Springer, Berlin, Germany
39. Song C, Liu F, Huang Y, Wang L, Tan T (2013) Auto-encoder based data clustering. Springer, Berlin, Heidelberg
40. Song W, Li C, Park C (2009) Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures. Expert Syst Appl 36:9095–9104
41. Starczewski A (2017) A new validity index for crisp clusters. Pattern Anal Appl 20:687–700
42. Teknomo K (2006) K-means clustering tutorials. Medicine 100:3
43. Thashina S (2020) Email based spam detection. International Journal of Engineering and Technical Researchs, 9
44. Uguz H (2011) A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. Knowl Based Syst 24(7):1024–1032
45. Uysal A, Gunal S (2014) Text classification using genetic algorithm oriented latent semantic features. Expert Syst Appl 41:5938–5947
46. Wyse N, Dubes C, Jain A (1980) A critical evaluation of intrinsic dimensionality algorithms
47. Xie X, Beni G (1991) A validity measure for fuzzy clustering. IEEE Trans Pattern Anal Mach Intel 13:841–847
48. Yangn M, Yang Y, Su T (2014) An efficient fitness function in genetic algorithm classifier for landuse recognition on satellite images. Sci World J. https:// doi. org/ 10. 1155/ 2014/ 264512
49. Yilmaz S, Toklu S (2020) A deep learning analysis on question classification task using word2vec representations. Neural Comput Appl 32:2909–2928