scientific reports

OPEN

Root cause prediction for failures in semiconductor industry, a genetic algorithm–machine learning approach

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

Failure analysis has become an important part of guaranteeing good quality in the electronic component manufacturing process. The conclusions of a failure analysis can be used to identify a component's flaws and to better understand the mechanisms and causes of failure, allowing for the implementation of remedial steps to improve the product's quality and reliability. A failure reporting, analysis, and corrective action system is a method for organizations to report, classify, and evaluate failures, as well as plan corrective actions. These text feature datasets must first be preprocessed by Natural Language Processing techniques and converted to numeric by vectorization methods before starting the process of information extraction and building predictive models to predict failure conclusions of a given failure description. However, not all-textual information is useful for building predictive models suitable for failure analysis. Feature selection has been approached by several variable selection methods. Some of them have not been adapted for use in large data sets or are difficult to tune and others are not applicable to textual data. This article aims to develop a predictive model able to predict the failure conclusions using the discriminating features of the failure descriptions. For this, we propose to combine a Genetic Algorithm with supervised learning methods for an optimal prediction of the conclusions of failure in terms of the discriminant features of failure descriptions. Since we have an unbalanced dataset, we propose to apply an F1 score as a fitness function of supervised classification methods such as Decision Tree Classifier and Support Vector Machine. The suggested algorithms are called GA-DT and GA-SVM. Experiments on failure analysis textual datasets demonstrate the effectiveness of the proposed GA-DT method in creating a better predictive model of failure conclusion compared to using the information of the entire textual features or limited features selected by a genetic algorithm based on a SVM. Quantitative performances such as BLEU score and cosine similarity are used to compare the prediction performance of the different approaches.

The development of microelectronic technologies provides new opportunities to improve the maintenance of production equipment from both the technical and managerial perspective. To establish this improvement in production, it is necessary to focus on an important step that is the failure analysis. This process is a technical procedure for studying how materials and products fail. It is important to understand how and why a component fails when it no longer performs its intended function[1]. The main goal of failure analysis is to find the underlying root cause of the failure, ideally with a view to removing it and identifying ways to prevent it from happening again. Objective failure analysis can have a number of good outcomes such as obtaining an information database that can be put to good use in preventing future failures, enhancing the quality and extending the life of products and services, and making the most of the economic aspects[2]. To meet these principal fundamental challenges in our digital world, it is important to build an information database to describe failures and their conclusions, allowing to ensure that increasingly complex electronic systems operate reliably and securely.

Many organizations use the Failure Reporting, Analysis, and Corrective Action System (FRACAS) to keep track of product problems. The FRACAS technique's main tasks are[3]: recording and capturing information about failures and problems, provide new information to support future reliability analyses, provide report summaries of incident counts and provide failure dataset and metrics to measure quality parameters. Developing a novel

Ecole des Mines de Saint-Etienne, Mathematics and Industrial Engineering, Organisation and Environmental Engineering, Henri FAYOL Institute, 42023 Saint-Etienne, France. ✉email: abbas.rammal@emse.fr

technique based on artificial intelligence (AI) to swiftly assess and discover faults during the development and manufacture of electronic components and systems, using the final report generated by FRACAS, is one of the key difficulties facing our digital world. The incorporation of AI and multi-structured data sources is critical to the success of data-driven maintenance. When an AI-enhanced technique is introduced and integrated into a reliability-centered maintenance analysis of complex production systems, failure rates are reduced, and availability is improved[4].

Text mining is an artificial intelligence (AI) technique that applies natural language processing (NLP) to convert unstructured text in documents and databases into normalized, structured data that can be analyzed or used to train machine learning (ML) algorithms[5]. Text mining is also a technique for extracting information from unstructured documents and identifying novel and previously unknown patterns. Then, the next step is feature or attribute selection. This step focuses on deleting elements that are not significant to the mining process[6]. In addition, this step has several advantages: reduce computational complexity; get fewer noises in the decision space and reduce the dimension to have more consistent and homogeneous dataset[7].

In our study, we have a textual dataset that consists of the description of the failure analysis and the failure conclusion for microelectronic technologies' products. Our objective is to construct a model able to predict the failure conclusion features from the failure analysis description features. However, not all textual information's are valuable for the construction of predictive model, while the use of limited number of a priori features may be tricky. Feature selection reduces dimensionality by selecting a subset of original input textual variables. In other words, the textual variable selection strategy decreases the dimension of textual features that may be relevant to a specific phenomenon by identifying the best minimum subset without transforming the data into a new set[8]. In order to achieve complicated models for prediction and classification algorithms, we implement the selection of relevant textual variables while excluding non-informative variables.

Various mathematical techniques have been used to select optimal subsets of variables: successive projections algorithm[9], backward/forward selection algorithm[10], reweighted adaptive competitive sampling, importance of variables for projection, elimination of non-informative variables[11], interval partial least squares regression[12], Monte Carlo-elimination of non-informative variables[13], Particle Swarm Optimization and Deep Learning Approach[14], feature learning enhanced convolutional neural network (FLE-CNN)[15], competitive adaptive reweighted sampling partial least squares[12], etc. However, most of these techniques are not well suited to textual datasets. On the other hand, the application of these methods leads to the loss of a lot of information during the analysis.

The genetic algorithm (GA) belongs to research techniques that emulate the principle of natural selection. GA performs a search in complex, large and multimode landscapes, and provides near-optimal solutions for objective or fitness function of an optimization problem[16]. However, the cost of computational time is high because its long string representation evolves in high dimensional space typical for textual data. A genetic algorithm is a bottom-up strategy that chooses the best features subset based on the "survival of the fittest" principle, with each chromosome competing with the others[16]. That is, the quality of chromosomes is assessed using a predetermined fitness function. 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. Supervised learning methods can be used to derive new fitness functions that can transform a textual data in a much lower-dimensional subspace more adequately regarding a specific application[17]. Different types of supervised methods exist in the literature. The best-known are Decision Tree model (DT), and Support Vector Machine model (SVM). A study was conducted to demonstrate that the combination of genetic algorithm and support vector machine method improves the textual classification accuracy of the spam dataset[18]. Another study shows that the efficiency of feature selection based on information gain and genetic algorithm can reduce the dimension of the text vector and improve the accuracy of text classification[19]. A recent paper proposes the genetic algorithm-oriented latent semantic feature methodology to achieve better representation of documents in text classification[20].

Therefore based on all the above, one can summarize the motivation of combining GA and supervised learning methods by the following:

The combination of Genetic Algorithms (GA) and supervised learning methods has been a popular topic of research in the field of machine learning and optimization. For example, in a study by Fernández et al. (2002), the authors used a GA to optimize the parameters of a support vector machine (SVM) for a classification task and showed that the combination of these two approaches led to improved performance compared to using only the SVM. Another study by Liu et al. (2011) proposed a GA-based approach for feature selection in conjunction with a decision tree classifier, showing that the combination of these two methods outperformed individual methods in several benchmark datasets. In addition to parameter optimization, GAs have also been used to search for optimal network architecture in deep learning. For instance, Real et al. (2017) proposed a method called "Large-Scale Evolution of Image Classifiers" where they used a GA to evolve the architecture of Convolutional Neural Networks (CNNs) and showed that the evolved architectures outperformed manually designed ones in the CIFAR-10 and CIFAR-100 image classification benchmarks. These studies demonstrate the potential of combining GA and supervised learning methods for improving performance in various applications, and highlight the need for further research in this area.

On the other hand, the research gaps and challenges and how we are overcoming these points can be summarized by the following:

The most challenging issues in this study are likely related to the task of developing a predictive model that can accurately predict the failure conclusions based on the failure descriptions. This is a challenging task as it requires the model to learn the relationship between the input features and the target output, which can be difficult due to the presence of noisy or irrelevant features, imbalanced class distributions, and non-linear relationships between features and target.

The proposed method addresses these challenges by combining a genetic algorithm with a decision tree classifier, referred to as GA-DT. The GA is used to search for a subset of the most discriminative features from the failure descriptions, which are then used as input to the decision tree classifier. By doing so, the GA helps to overcome the issue of noisy or irrelevant features, as it only selects the most informative features for the classifier to use. Additionally, decision trees are known to be able to handle imbalanced class distributions and non-linear relationships, making them a suitable choice for this task.

The effectiveness of the proposed GA-DT model is demonstrated through experiments, which show improved performance compared to using only a decision tree classifier or only a genetic algorithm. This highlights the contribution of the proposed method, which combines the strengths of both GA and decision tree classifiers to improve the accuracy of the predictive model.

Then, the main goal of this study is to build an advanced predictive model capable of predicting failure outcomes significantly using failure analysis description. Another objective is to investigate the potential of using a supervised variable selection technique using a genetic algorithm to identify more informative and useful text features from the text dataset that contains a very large number of words, and to show whether the features selected by the proposed method can significantly improve the performance of predictive models between failure conclusion features and failure analysis description features. We propose a methodology based on the association of a genetic algorithm with a supervised model such as the decision tree or support vector machine evaluated by the F1 score as a fitness function for the identification of the discriminating variables applied the failure analysis textual data. This function allows calculating the accuracy of predictive models applied on unbalanced dataset. The suggested algorithms are called GA-SVM and GA-DT.

This article is structured as follows: In the second part, we present what Feature Selection is and its associated algorithms. Then, we detail the operating principle of population-based metaheuristic algorithms. We focus more particularly on Genetic Algorithm, and their detailed operation that allows the selection of relevant features. In this part of this work, we present machine-learning algorithms used to calculate the fitness value for the metaheuristic algorithms. We go deep in the description of supervised methods such as Support Vector Machine (SVM), and Decision Tree (DT). In the third part, we present the results obtained by applying our metaheuristic-machine learning algorithms combination on the failure conclusion features and failure analysis description features. We show that the results observed allow us to pick the most valid model, which is the GA-DT, confirmed with the different metrics as BLEU score and cosine similarity at a division of 70% training set and 30% testing set. Finally, and after discussing the results, we close with a general conclusion on the interest of the combination of feature selection algorithms with machine learning methods, its capability and performance in dimension reduction, and on the possibilities of implementing other tools belonging to metaheuristic algorithms to improve the accuracy rates.

Mathematical methods

Supervised learning modelling.

Multiclass error-correcting output codes (ECOC) model using support vector machine (SVM). The Error-Correcting Output Codes (ECOC) framework is a basic yet effective method for dealing with the multi-class categorization problem based on the embedding of binary classifiers, where the classifier consists of multiple binary learners such as support vector machines (SVMs). The ECOC model classifiers allow to store training data, parameter values, prior probabilities, and coding matrices[21]. These classifiers aims to perform tasks such as predicting labels or posterior probabilities for new data. The multi-class ECOC model using SVM methods consist of three major components that are encoding, binary classifier learning, and decoding steps. In the coding procedure, a coding matrix is usually first determined for several classes, where each row of the coding matrix represents a specific class. Then, a group of independent binary classifiers is formed based on a different partition of the original data according to each column of the coding matrix. Finally, a new data is predicted as a specific class through the decoding procedure based on the outputs of the learned binary classifiers and the coding matrix.

Let $X = \{x_j\}_{j=1}^{n}$ be a training set of n samples of observed variables, where a d-dimensional vector represents each sample, and let C be an unobserved random variable denoting the class membership of x_j, where $C \in \{c_1, \ldots, c_k, \ldots, c_K\}$ with K denoting the number of class. In k^{th} class SVM problem, class c_k is separated from the remaining classes. All k binary SVM classifiers are combined together to make a final multi-class classifier. Here the remaining means that all the data points from classes other than c_k are combined to form one class c_l. The optimal hyperplane that separates data points from the class c_k and the combined class c_l is found by using the standard SVM approach. We denote the optimal separating hyperplane discriminating the class c_i and the combined class c_k as[22]:

$$g_k(x_j) = w_k \times \phi(x_j) + b_k \quad k \in \{1, \ldots, K\}. \tag{1}$$

where $w_k \in \mathbb{R}^S$ is the weight vector, b is the bias and the mapping function ϕ projects the training data into a suitable feature space $\mathbb{R}^S$ to allow for nonlinear decision surfaces. The parameters of the decision function $g_k(x_j)$ are determined by the following minimization[23]:

$$\min J(w_k, \xi) = \frac{1}{2}\|w_k\|^2 + C\sum_{j=1}^{n} \xi_j. \tag{2}$$

subject to

$$y_j\left(w_k^T \phi(x_j) + b_k\right) \geq 1 - \xi_j \quad \xi_j \geq 0; j = 1 \ldots n. \tag{3}$$

with scalar $y_j \in \{-1, +1\}$ denoting its class label, $C \in \mathbb{R}^+$ is a regularization constant and ξ_j denote a slack variable can be introduced to relax the separability constraints in Eq. (2).

The decision rule $f_k(x_j)$ that assigns the vector x_j to the class c_k given by:

$$f_k(x_j) = \text{sign}(g_k(x_j)). \tag{4}$$

The main difficulty in this approach is that the outputs of the classifiers $f_k(x_j)$ are binary values. The usual way to handle this problem is to ignore the sign-operator in Eq. (4). After finding all the optimal hyperplanes given by $g_k(x_j)$ for $k \in \{1, \dots, K\}$, we say x_j is in the class which has the largest value of the decision function and is given by[24]:

$$\hat{c_k}(x_j) = \underset{k}{\text{argmax}}\big(g_k(x_j)\big). \tag{5}$$

In this approach the index of the largest component of the discriminant functions $g_k(x_j)$ for $k \in \{1, \dots, K\}$ is assigned to the data point x_j. The error rate $\mathcal{R}^{SVM}$ of the SVM classifier, which is defined as:

$$\mathcal{R}^{SVM}(x_j) = \frac{1}{n}\sum_{j=1}^{n} 1_{c_k \neq \hat{c_k}}. \tag{6}$$

with x_j that belongs to the class c_k estimated by the method classifier in the class $\hat{c_k}$ and $1_{c_k \neq \hat{c_k}}(x_j)$ is the indicator function defined as:

$$1_{c_k \neq \hat{c_k}}(x_j) = \begin{cases} 1, & \text{if class } c_k \neq \hat{c_k} \\ 0, & \text{if class } c_k = \hat{c_k} \end{cases} \tag{7}$$

Decision tree classifier. A decision tree classifier is a non-parametric classifier that does not require any a priori statistical assumptions to be made regarding the underlying distribution of data. The basic structure of the decision tree, however, consists of one root node, a number of internal nodes and finally a set of terminal nodes. A node is a subset of the predictors that is used to determine a split. A non-terminal node or parent node is a node that is further split into two child nodes. Growing a tree consists of selecting the optimal splits to determine a non-terminal node, and the assignment of each terminal node to a class[25]. The data is recursively divided down the decision tree according to the defined classification framework.

Classes are simply assigned to a terminal node by observing which class is mostly commonly observed in that region of the tree. Thus, the challenge is to optimally choose the best variable and split that variable to maximize the purity or similarity among the responses. The impurity of a parent node τ, denoted $i(\tau)$, is zero when all observations are in the same class. A split s is determined by selecting the best predictor and split value that optimizes the highest reduction in purity[26]:

$$\Delta(s, \tau) = i(\tau) - \sum_{b=1}^{B} p(\tau_b/\tau)i(\tau_b). \tag{8}$$

where τ_b denotes child node b, $p(\tau_b/\tau)$ is the proportion of observations in τ that are assigned to τ_b, and B is the number of branches after splitting. Two common impurity functions are the entropy criterion[26]:

$$i(\tau) = -\sum_{k=1}^{K} p_k \log_2(p_k). \tag{9}$$

and the Gini index criterion

$$i(\tau) = -\sum_{k=1}^{K} p_k^2. \tag{10}$$

where p_k is the proportion of observations in class c_k with $k \in \{1, \dots, K\}$. Pruning is based upon successive steps of removing lower branches that lead to improved classification rates. Once the final tree is determined by $\Delta(s, \tau)$, it is natural to evaluate its predictive performance by comparing the observed class to the predicted class for observation x_j. In a terminal node m, representing a region R_m with n_m observations, let

$$\hat{p_{mk}} = \frac{1}{n_m}\sum_{j=1}^{n_m} 1_{C_k}(x_j). \tag{11}$$

denote the proportion of class c_k observations in terminal node m[27]. We classify the observations in node m to class

$$\hat{c_k}(x_j) = \underset{k}{\text{argmax}}(\hat{p_{mk}}). \tag{12}$$

The misclassification error rate is simply the proportion of observations in the node that are not members of the majority class in that node.

$$\mathcal{R}^{DTC}(x_j) = \frac{1}{n}\sum_{j=1}^{n}\left(1 - \max_k(\hat{p_{mk}}(x_j))\right). \tag{13}$$

Supervised learning modelling. Genetic algorithms (GA) are a type of evolutionary optimization computation that became popular through the work of Holland[28]. These algorithms are based on the concept of natural selection of solutions by copying its main principles. Each solution may be considered as a population where each element is represented in the form of a chromosome, with selected textual feature positioned as genes[28]. The GA steps reproduce the various evolutionary operations such as crossover and mutation allowing to select for each generation the best chromosomes and to identify at the end an optimal chromosome with respect to an optimization criteria defined by a fitness function[29]. Figure 1 shows the steps of the informative feature selection procedure using a GA[30].

The GA can be applied on data matrix $X = \{x_j(y)\}_{j=1}^{n}$ with $x_j(y) \in \mathbb{R}^d$ and y is the set of textual features for failure description dataset. This procedure gives in each one of these cases an optimal chromosome $z_0 = [z_{01} \cdots z_{0l} \cdots z_{0L}] \in \mathbb{R}^L$ with z_{0l} textual feature form y and L the number of variables chosen to select. The optimal chromosome allows extracting a new sub-data matrix $\{x_j(z_0)\}_{j=1}^{n}$ of under-dimensioned data on which we can apply methods of data analysis. The GA steps are briefly described thereafter, being detailed in the articles[31] and[32].

1. Initialization: The initial parameters are : the chromosome size L (the number of genes corresponding to the number of features to be selected in each case) ; the population size N (the number of chromosomes per generation) ; the number of elites N_e (the chromosomes with the best fitness values in the current generation that are guaranteed to survive to the next generation) ; the fraction of crossover F_c (the number of chromosomes selected to perform crossover N_c such that $N_c = F_c \times (N - N_e)$). The stop parameters are: the maximal number of iterations T and the tolerance ϵ for the 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 y ($L < S$ is the size of the chromosomes):

$$P(0) = \left\{z_i = [z_{i1}, \ldots, z_{iq}, \ldots, z_{iL}] \in \mathbb{R}^L\right\}_{i=1}^{N}. \tag{14}$$

The initial population $P(0)$ of wavenumbers variables is chosen randomly from the set of uniformly distributed variables ranging over their maximum and minimum limits[31] :

$$z_i^0 \sim U(z_i^{min}, z_i^{max}). \tag{15}$$

where z_i^0 signifies the initial l^{th} variable of the i^{th} population; z_i^{min} and z_i^{max} are the minimum and maximum limits of the l^{th} decision variable; $U(z_i^{min}, z_i^{max})$ signifies a uniform random variable ranging over$[z_i^{min}, z_i^{max}]$. Then, computation is done over generations. For each generation (t), we obtain the population of chromosomes $\{z_{i(t)}\}_{i=1}^{N}$ the steps thereafter give another population of chromosomes $\{z_{i(t+1)}\}_{i=1}^{N}$.

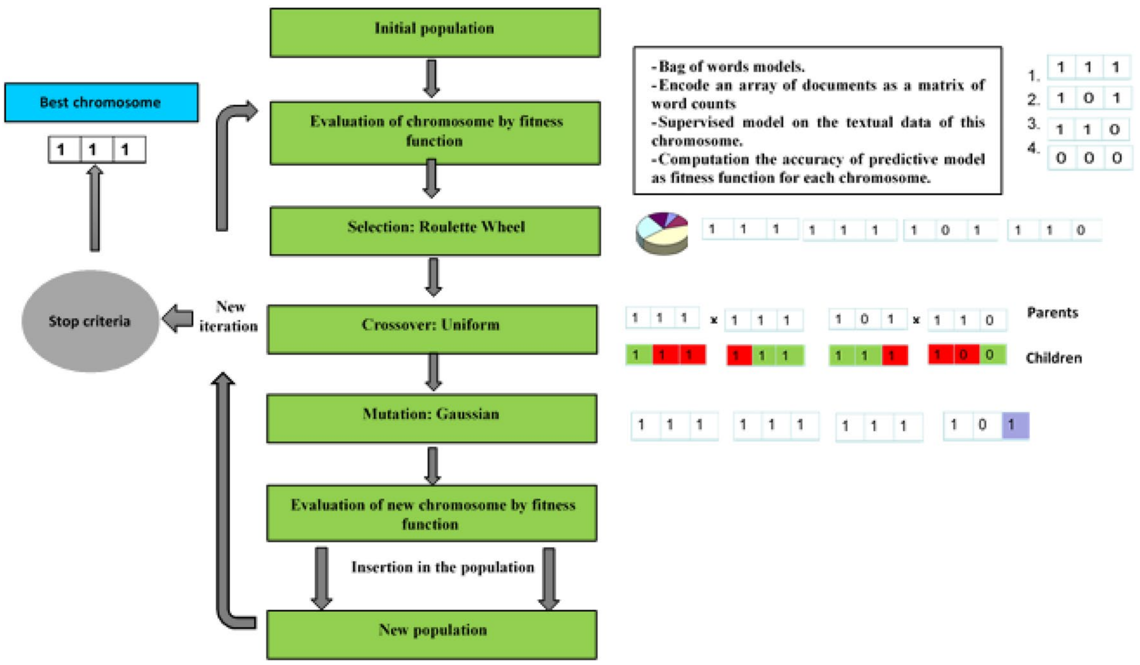
Initial population
Evaluation of chromosome by fitness function
Selection: Roulette Wheel
Crossover: Uniform
Mutation: Gaussian
Evaluation of new chromosome by fitness function
Insertion in the population
New population
Best chromosome
1 1 1
Stop criteria
New iteration
-Bag of words models.
-Encode an array of documents as a matrix of word counts
-Supervised model on the textual data of this chromosome.
-Computation the accuracy of predictive model as fitness function for each chromosome.
1. 1 1 1
2. 1 0 1
3. 1 1 0
4. 0 0 0
Parents
Children

Figure 1. Synoptic representation of the proposed GA methodology.

2. Evaluation: Each chromosome$z_{i(t)}$ is rated by a fitness function $F(.)$ that assigns a value $F_i = F(z_{i(t)})$. The smaller the F_i value is, the more corresponding chromosome will have chance to be selected. The role of a fitness function is to measure the quality of the chromosome in the population according to the given optimization objective[32]. Since we want to create a predictive model between the failure description dataset X and the failure conclusion dataset Y, we propose to use the supervised model for each chromosome such as the decision tree (DT) and the support vector machine (SVM) and then to calculate the F_1 score of each model built as a fitness function to assess the qualities of our predictive model obtained. The F_1 score of these supervised learning methods is one of the simplest methods that can be used as a classical fitness function to evaluate the accuracy of predictive model. The fitness function is defined as follows:
$$F_i^{\text{model}} = F^{\text{model}}(z_i) = \left(1 - F_1^{\text{model}}(z_i)\right). \tag{16}$$
with
$$F_1^{\text{model}}(z_i) = 2 \times \frac{P_r^{\text{model}}(z_i) \times R_c^{\text{model}}(z_i)}{P_r^{\text{model}}(z_i) + R_c^{\text{model}}(z_i)}. \tag{17}$$
where F_1^{model} is theF_1 score defined as the harmonic mean between precision and recall; P_r^{model} is positive predictive value (precision) and R_c^{model} is the sensitivity (Recall) of the predictive model such as SVM and DT. This function (F_1 score) is very useful when dealing with unbalanced class issues. These are problems when one class can dominate the data set. For each fitness function F_i, the values are ordered in ascending order and the best N_e chromosomes are selected based on this ordering. These surviving chromosomes will be copied unchanged in the next population.
3. Selection: Used for choosing parents from the population for crossing, this step may be implemented in different ways: rank, stochastic, roulette wheel, stochastic universal sampling selection, etc. We have chosen the stochastic universal sampling selection since this method is zero-biased, has no deviation between the expected reproduction rate and the algorithmic sampling frequency, and has a minimum spread[33]. The selection is performed probabilistically so that an individual's selection probability is proportional to the individual's fitness. First, we compute the probability p_i of selecting the chromosome z_i and the cumulative probability q_i:
$$p_i = \frac{F_i}{\sum_{i=1}^{N} F_i}. \tag{18}$$
$$q_i = \sum_{i=k}^{i} p_k. \tag{19}$$
Next, we generate a uniform random number $r \in [0, \frac{1}{N}]$. If $r < q_1$ then we select the first chromosome z_1, otherwise we select the chromosome z_i such that $q_{i-1} < r \leq q_i$. The ascending ordered F_i values allows selecting N_e chromosomes guaranteed to survive to the next generation and $N_p = (F_c + 1) \times N - 2N_e$ parent chromosomes for the crossover.
4. Crossover: This step attempts to extract genes from the selected chromosomes and recombines them into potentially superior children. We have chosen the uniform crossover since it gives good results in a majority of the cases. A gene is randomly selected either from the first or from the second parent[34]. The crossover operation gives $N_c = (F_c \times N) - N_e$ children. To explain the uniform crossover, the parent's chromosomes $p_1[z_{iq}]$, $p_2[z_{iq}]$ and the children chromosomes $o_1[z_{iq}], o_2[z_{iq}], q = 1 \ldots L$ 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[35]:
   - 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 Eqs. (20) and (21):
$$o_1[z_i] = p_1[z_i] \quad \text{for each} \quad \alpha_q > 0.5, o_1[z_{iq}] = p_2[z_{iq}]. \tag{20}$$
$$o_2[z_i] = p_2[z_i] \quad \text{for each} \quad \alpha_q > 0.5, o_2[z_{iq}] = p_1[z_{iq}]. \tag{21}$$
5. Mutation: is a genetic operator used the modification of the value of gene to maintain genetic diversity from one generation of a population to the next one. We have chosen the Gaussian operator since it produces the best results for most of the fitness functions[36]. This operator adds a unit Gaussian distributed random value to $N_p - 2N_c$ chromosomes. The new values of the genes are then rounded to the nearest integer. The standard deviation of this distribution is the parameter that the call "scale" which is equal to one in the first generation, but this parameter is controlled during the next generations by another parameter that is "shrink". The standard deviation at the tth generation, σ_t is the same at all coordinates of the parent chromosome, and is given by the recursive formula[37]:
$$\sigma_t = \sigma_{t-1} \times (1 - \text{shrink}\frac{t}{T}). \tag{22}$$

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.

6. Steps 1 to 5 are repeated until the maximal number of iterations T is reached or when GA has converged, i.e. the average relative change in the fitness function value is less than the tolerance ϵ. This procedure gives an optimal chromosome z_0, which depends on the fitness function and the initial values. With the proposed choice of the GA steps, we have found that the same optimal chromosome is found whatever the initial values of chromosomes were used.

Computing similarities between documents.

BLEU score. The BiLingual Evaluation Understudy (BLEU) scoring algorithm assesses the similarity between a predictive document and a collection of reference documents. To assess the quality of document translation and summarization models, we use the BLEU score. The n-gram counts, clipped n-gram counts, modified n-gram precision scores, and a brevity penalty are used to calculate the BLEU score[38].

If necessary, the clipped n-gram counts function Countclip truncates each n-gram gram's count so that it does not exceed the highest count found in any one reference for that n-gram. The clipped counts function is defined as follows:

$$\text{Count}_{\text{clip}}(\text{n-gram}) = \min_n \big(\text{Count(n-gram); maxRef(n-gram)}\big). \tag{23}$$

where Count(n-gram) represent the n-gram counts and maxRef(n-gram) is the highest n-gram count observed in a single reference document for that n-gram. The updated n-gram precision scores are calculated as follows:

$$p_n = \frac{\sum_{D \in \{\text{Predictive Document}\}} \sum_{\text{n-gram} \in D} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{D' \in \{\text{Predictive Document}\}} \sum_{\text{n-gram}' \in D'} \text{Count}_{\text{clip}}(\text{n-gram}')}. \tag{24}$$

where n is the length of n-gram and Predictive Document is the set of sentences in predictive documents, D and $D^{'}$ are predictive documents . Given an n-gram weight vector w, the BLEU score formulation is given by[38]:

$$\text{BLEU score} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log(\bar{p_n})\right). \tag{25}$$

where N is the greatest length of n-grams, $\bar{p_n}$ are the geometric means of the modified n-gram precisions, and BP is the brevity penalty defined as

$$1_{c_k \neq \hat{c_k}}(x_j) = \begin{cases} 1, & \text{if} \quad c > r \\ \exp(1 - \frac{r}{c}), & \text{if} \quad c < r \end{cases} \tag{26}$$

BLEU score returned as a scalar value in the range [0, 1]. A BLEU score close to zero indicates low similarity between the predictive document and the references. A BLEU score close to one indicates strong similarity. If the predictive document is identical to one of the reference documents, the score is one.

Cosine similarity. The similarity of two vectors in an inner product space is measured by cosine similarity. It determines whether two vectors are pointing in the same general direction by measuring the cosine of the angle between them. In text analysis, it is frequently used to determine document similarity[39]. Let us see how documents in our corpus are related to one another. Let t_1 and t_2 be two vectors that represent the topic associations of documents d_1 and d_2, respectively, where $t_1^{(k)}$ and $t_2^{(k)}$ are the number of terms in d_1 and d_2 that are connected with subject k respectively. The cosine similarity can then be used to calculate a measure of document similarity[39]:

$$c = \frac{\sum_k t_1^{(k)} \times t_2^{(k)}}{\|t_1\| \times \|t_2\|}. \tag{27}$$

where $\|t_j\|$ denotes the norm of vector t_j. Cosine similarity score indicate a scalar value in the range [0, 1]. A cosine similarity close to zero indicates poor similarity between the predictive document and the references. A cosine similarity close to one indicates strong similarity.

Proposed methodology for evaluation the predictive model of failure analysis.

In this section, we present the methodology proposed for the selection of variables by the genetic algorithm combined with the decision tree (GA-DT) or the support vector machine (GA-SVM) model applied to textual data. Figure 2 shows the steps of the failure analysis modeling methodology by extracting the best textual features using supervised variable selection techniques and representing the predictive models between failure description X and conclusion of failure Y for this analyzed data. This proposed methodology consists of three main phases. First, we perform the pipeline preprocessing of the failure analysis description X and conclusion of failure Y. This is a most important and time consuming part of textual data because failure to clean and prepare the data could compromise the predictive model. The Phase 2 shows the application of the Word2Vec vectorization method on preprocessed textual data to obtain numerical data.

The Phase 3 shows the application of GA variable selection method combined with decision tree or support vector machine supervised learning on vectorized preprocessed data. To quantify the accuracy of the selected predictive model on discriminate textual features, we compute the different metrics like as BLEU score and

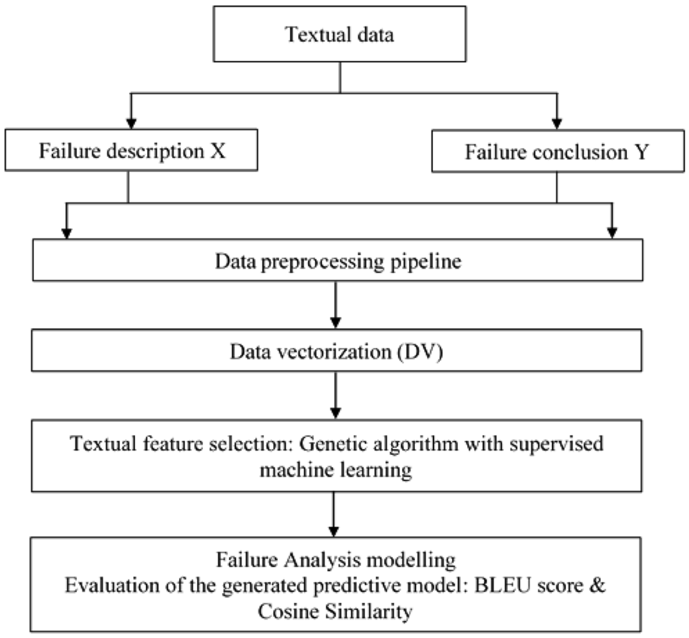
Textual data
Failure description X
Failure conclusion Y
Data preprocessing pipeline
Data vectorization (DV)
Textual feature selection: Genetic algorithm with supervised machine learning
Failure Analysis modelling
Evaluation of the generated predictive model: BLEU score & Cosine Similarity

Figure 2. Synoptic representation of the proposed GA methodology.

Cosine similarity. Finally we compare the predicted textual conclusion and the original textual conclusion to confirm the similarities between them.

Application and results

All data treatments were performed using the MATLAB-R2022b environment, and scripts are available upon request.

Data description and structure.

Data description and analysis is an important phase that precedes modeling. An accurate representation of the data is necessary to define the parameters of a model. We have a textual dataset on failure analysis of Microelectronics production. The original dataset provided by STMicroelectronics dated between 2019 and 2021 consists of two parts: the first is the description of failure analysis X (source of failure request, properties of samples and details of failure) and the second is the dataset of its conclusion Y (analysis conclusion, success rate and cycle time). Tables 1 and 2 contains a list of different features of X and Y with a short description. This data has been transformed from a vertical stacking of analysis to horizontal stacking. This means that its description (objective, context, etc.) as well as its conclusion of failure represent an observation. The transformation reduces the data size to 12,300 observations and we keep 19 preprocessed features out of dates. After getting clean processed data using the preprocessing pipeline introduced in[40], we vectorize using on Word2Vec. Genism's Word2Vec settings are kept except the vocabulary size is set to 1000 and the minimum word is three[41].

In formalizing our approach, we use the following notations: let $X = \{x_{ij}\}_{i=1,j=1}^{n,m}$ represents the input space of a given dataset where n is the number of samples and m is the number of features; $Y = \{y_{ij}\}_{i=1,j=1}^{n,p}$ represents the output space of conclusion failure dataset where p is the number of features.

Preprocessing pipeline.

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 are all examples of preprocessing text. We will start with the stages of the preparation pipeline:

Feature (Y)	Description
Pt failure/Elt by sample	This is the point on the sample where an expert observes the failure during fault analysis
Macro failure mode by sample	Macro failure mode of a sample is the type of failure observed on sample before a failure analysis is requested
Elementary failure mode	Elementary failure observed during analysis
Tech cause/defect by sample	Technical cause of the failure
Analysis conclusion	Conclusion of the failure analysis

Table 1. Textual features and its description present in the data set of the conclusion of failure analysis Y.

Feature (X)	Description
Subject	A unique subject particular to the fault expert desire to analyses
Context	Context of the failure analysis
Objectives/work description	Objective of the fault analysis and a description of how to proceed provided by expert
Source of failure (request)	Identification of source of the failure for the component of a given sample
Source of failure (detailed)	Details of the source of the failure record for the identified failure source
Requestor	Verify a product problem and determine corrective action through the evaluation of a small quantity
Requested activity	Tries to find the wrong things in material, design, production, installation, and service
Priority level	Level of priority of the analysis
High confidentiality	Confidentiality level of failure analysis
Confidentiality	The principle of confidentiality consists of giving access to data and information only to authorized persons with a defined and specific need to see or use such information
Reference	Is the identifier that describes the location and team handling the failure product. e.g., ADG CST FA Team-19-00281 contains information about location [ADG], failure analysis team [CST FA Team], year [19] and failure analysis number [00281]
Organization	Organizational failure analysis is described for in-depth identification of organizational deficiencies and failures that can lead to accidents
Organization Division	The description of current organizations and processes that control the causes of failures
Department	The failure analysis department described the section under the FA lab where the analysis is carried out
Cost Center	The Cost Center is a department or a distinct unit or division within the framework of a company. These cost centers indirectly contribute to the organization's profits
Site	This describes the location where the Failure Laboratory is situated e.g., Grenoble. The FA process is finished once there is enough information to make a conclusion about the location of the failure site
Lab	Determine the nature and the causes of a defect with an expertise of the product in a laboratory
Lab Team	Select the causes of a failure by the teams of people from the failure analysis laboratory
Project	The description given to the failure analysis project given by the FA team

Table 2. Textual features and its description present in the data set of the description of failure analysis X.

- Symbol and alphanumeric removal: This technique removes words from the text that do not add to the intelligence pattern or the analytic sample, such as symbols and occasionally alphanumeric words. They are just stop words and inflexions that are used to emphasize meaning, thus they have been removed[42].
- Tokenization and Thresholding: Tokenizing is to change or break the sentence into a token by using a separator[42]. Thresholding is a term used to remove words below certain length. In this paper, we set the threshold at two.
- Stemmatization and Lemmatization: this is the process of removing affixes (prefixes and suffixes) from textual features[43].
- Abbreviation: Abbreviations are common in FRACAS, hence the need to replace them with their original meaning. We have created a dictionary of abbreviations to alleviate this challenge.

Optimization of the parameters of the GA.

A critical phase of GA is the right choice of its parameters in order to ensure the convergence of the algorithm to the optimal solution. The parameters have been initialized as follows: the number of elites $N_e = 2$, the fraction of crossover $F_c = 0.8$, the maximum number of iterations $T = 100$, the population size $N = 100$ and the tolerance $\epsilon = 10^{-6}$. These values have been used for several implementation of GA since they give good results for similar data[44].

To identify the optimal values for L and N, the GA was evaluated for different sizes of chromosomes. When the algorithm has converged (tolerance ϵ) or when it has reached the maximum number of iterations (T), the values chromosome size L that give the maximize value of the fitness function are chosen as the optimal values (Eq. 28):

$$L = \max_{L=3\cdots 8} \{F(z_i)\}. \tag{28}$$

The best accuracy of GA-SVM and GA-DT was evaluated for different sizes of chromosomes, $L = 3, \ldots, 8$. Figures 3 and 4 show the fitness values of GA-DT and GA-SVM algorithms respectively. We found that $L = 3$ or 4 gives the highest fitness value for both methods. This indicates we need all four failure description features to build the best predictive model of analysis conclusion of failure.

Results and discussion of the proposed methodology for the prediction of the conclusion of failure.

The proposed methodology has been applied with two different fitness functions (SVM and DT). After selecting variables by the GA-SVM and GA-DT algorithms, we calculated the accuracy (%) to evaluate the performance of a predictive model, BLEU score and cosine similarity as metrics in order to quantify the results of the prediction of the conclusion of failure.

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted document to the total documents.

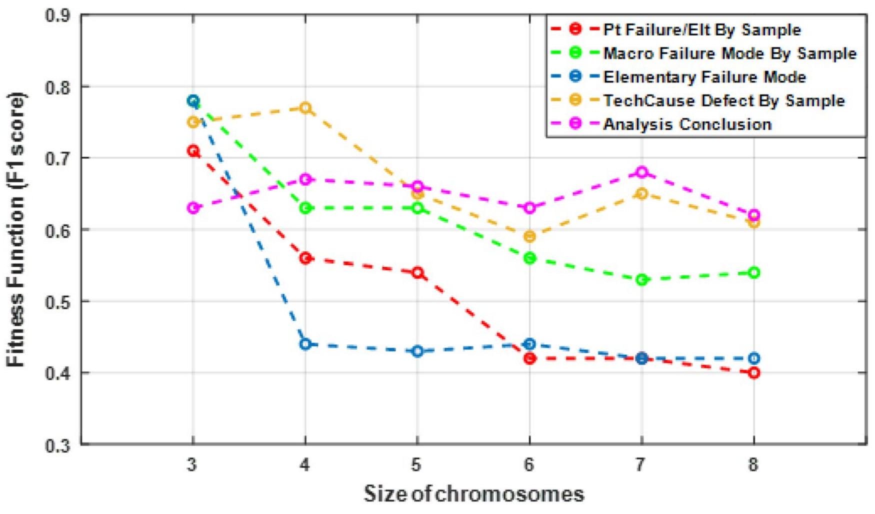

Figure 3. Values of GA-DT fitness functions for different sizes of L chromosomes. The optimal value is the highest F1 score.

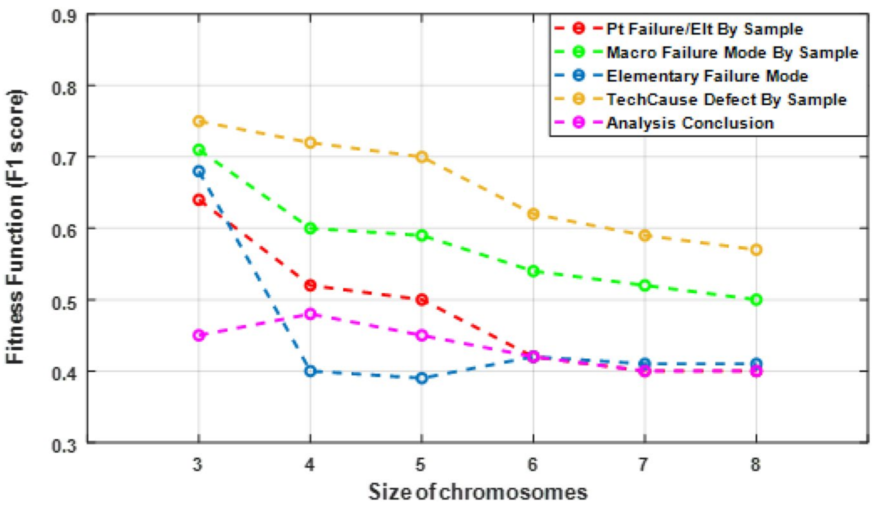

Figure 4. Values of GA-SVM fitness functions for different sizes of L chromosomes. The optimal value is the highest F1 score.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}. \tag{29}$$

Where TP is True Positives, TN is True Negatives, FP is False Positives and FN is False Negatives. FP and FN, these values occur when the actual documents contradicts with the predicted documents. These values (BLEU score, Cosine similarity and Accuracy), presented in Table 3, confirm that the GA-DT allows a better predictive model of the textual samples to predict the failure conclusion (features Y) compared to the other algorithm such as GA-SVM. We can see that the first four features of Y give good precision and good values of BLEU score and cosine similarity for GA-DT method except the last textual feature which is the conclusion of the analysis because each sample recorded on this variable is a large textual paragraph. About this latter feature we can say that the metrics calculated (accuracy = 25%; BLEU = 0.32; Cosine = 0.30) are very good compared to the other

Algorithm	Feature (Y)	Accuracy	BLUE score	Cosine similarity
GA-SVM	Pt failure/Elt by sample	74%	0.56	0.61
GA-DT	Pt failure/Elt by sample	**84%**	**0.72**	**0.85**
DT on all features	Pt failure/Elt by sample	70%	0.54	0.70
SVM on all features	Pt failure/Elt by sample	67%	0.48	0.52
GA-SVM	Macro failure mode by sample	75%	0.70	0.72
GA-DT	Macro failure mode by sample	**84%**	**0.95**	**0.97**
DT on all features	Macro failure mode by sample	72%	0.65	0.70
SVM on all features	Macro failure mode by sample	70%	0.62	0.65
GA-SVM	Elementary failure mode	72%	0.66	0.70
GA-DT	Elementary failure mode	**78%**	**0.95**	**0.97**
DT on all features	Elementary failure mode	69%	0.64	0.66
SVM on all features	Elementary failure mode	68%	0.62	0.64
GA-SVM	Tech cause/defect by sample	70%	0.48	0.54
GA-DT	Tech cause/defect by sample	**77%**	**0.75**	**0.93**
DT on all features	Tech cause/defect by sample	68%	0.65	0.67
SVM on all features	Tech cause/defect by sample	65%	0.60	0.62
GA-SVM	Analysis conclusion	13%	0.09	0.08
GA-DT	Analysis conclusion	**25%**	**0.32**	**0.30**
DT on all features	Analysis conclusion	8%	0.04	0.06
SVM on all features	Analysis conclusion	6%	0.02	0.05

Table 3. Values of the accuracy, BLEU scores, cosine similarities for both GA-SVM and GA-DT algorithms. A higher value of these metrics signifies a better predictive model within the conclusion of failure. Significant values are given in bold.

studies on textual dataset. One can also find that the application of variable selection by the genetic algorithm improves the accuracy of the model. These results are showed in Table 3.

In Table 4 we present some examples of results obtained after the application of the genetic algorithm with decision tree (GA-DT). We display the three best predictions for each failure analysis conclusion text sample. Then, we calculate the BLEU score to quantify the similarity between these predicted samples and the original sample. One can find that the values of BLEU scores are very close to one. This indicates strong similarity between the predicted samples and the reference ones.

Conclusion

We have proposed a methodology based on the association of a genetic algorithm with some supervised classifier methods for identification of discriminant textual features for the study of the best predictive model of failure conclusion using the features of failure descriptions.

The implementation of a genetic algorithm with a decision tree classifier as the fitness function led to the identification of a few interesting features. The BLUE score and the cosine similarity are used to evaluate the similarity between a predictive documents and a set of reference documents. We obtained very interesting values that indicate a strong similarity between the predictive documents and the references. We have also found that the application of variable selection by the genetic algorithm improve the accuracy and the metrics of the model obtained by DT or SVM methods.

We have shown that the discriminating features selected by the proposed GA-DT method provide the best predictive model of the failure conclusion according to the description of the failure process compared to GA-SVM model or the direct application of the decision tree or the support vector machine applied to all the features of the description of the failure (i.e., without any preselection method). As a perspective, we are working towards addressing the following challenges: 1) Improving the performance of the model by applying a generative sequence-to-sequence language model for failure conclusion generation given failure description; 2) Propose a methodology based on Genetic algorithm (GA) with decision tree (DT) to select the most important input variables that best predicts the conclusion (root cause) of a Failure analysis (FA). These variables will then be used to train a Transformer model for failure conclusion generation such as GPT2 transformer model etc.

Sample	Predict Y	BLUE score
[‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘arsen’, ‘compar’, ‘sampl’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’]	(1) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘leakag’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.974
	(2) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assemb’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘local’, ‘crater’]	0.949
	(3) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘detect’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘electron’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’]	0.938
[‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘sampl’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘local’, ‘crater’]	(1) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assemb’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘local’, ‘crater’]	0.953
	(2) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘leakag’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.943
	(3) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘1-trichloroethan’, ‘curv’, ‘trace’, ‘short’, ‘beryllium’, ‘detect’, ‘arsen’, ‘compar’, ‘ scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘ attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘local’, ‘hotspot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’, ‘pend’, ‘requestor’, ‘decis’]	0.917
[‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘1-trichloroethan’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘crater’]	(1) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘minor’, ‘leakag’, ‘detect’, ‘arsen’, ‘compar’, ‘transistor’, ‘outlin’, ‘packag’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.912
	(2) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘leakag’, ‘detect’, ‘short’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘electron’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’]	0.909
	(3) [‘induct’, ‘reactanc’,‘ray’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘leakag’, ‘detect’, ‘1-trichloroethan’, ‘short’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘electron’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’]	0.908
[‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘arsen’, ‘compar’, ‘sampl’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘abnorm’, ‘visual’, ‘crater’, ‘defect’]	(1) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘transistor’, ‘outlin’, ‘packag’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘spot’, ‘intern’, ‘visual’, ‘reveal’, ‘burnt’, ‘mark’, ‘crater’, ‘defect’]	0.916
	(2) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘decap’, ‘recov’, ‘transistor’, ‘outlin’, ‘packag’, ‘perform’, ‘crater’]	0.862
	(3) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘minor’, ‘leakag’, ‘detect’, ‘arsen’, ‘compar’, ‘transistor’, ‘outlin’, ‘packag’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.853
[‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘leakag’, ‘beryllium’, ‘detect’, ‘sampl’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘local’, ‘crater’]	(1) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘leakag’, ‘beryllium’, ‘detect’, ‘short’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘die’, ‘observ’, ‘carbon’, ‘attach’, ‘transistor’, ‘consid’, ‘critic’, ‘pwsso36l’, ‘dual’, ‘chip’, ‘base’, ‘spec’, ‘intern’, ‘assembl’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.971
	(2) [‘induct’, ‘reactanc’, ‘ray’, ‘miss’, ‘wire’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘1-trichloroethan’, ‘nobelium’, ‘abnorm’, ‘beryllium’, ‘observ’, ‘curv’, ‘trace’, ‘short’, ‘detect’, ‘leakag’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘intern’, ‘assembl’, ‘defect’, ‘electron’, ‘reveal’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘crater’]	0.862
	(3) [‘induct’, ‘reactanc’, ‘ray’, ‘nobelium’, ‘abnorm’, ‘inspect’, ‘astatin’, ‘reject’, ‘sampl’, ‘curv’, ‘trace’, ‘short’, ‘leakag’, ‘beryllium’, ‘detect’, ‘1-trichloroethan’, ‘arsen’, ‘compar’, ‘scan’, ‘auger’, ‘microscopi’, ‘analysi’, ‘delamin’, ‘issu’, ‘die’, ‘attach’, ‘observ’, ‘intern’, ‘assemb’, ‘defect’, ‘liquid’, ‘crystal’, ‘hot’, ‘spot’, ‘nearbi’, ‘pin’, ‘failur’, ‘local’, ‘crater’]	0.823

Table 4. Some examples of textual samples of analysis conclusion and their three best predictions by the GA-DT algorithm.

Data availability

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: rammal_abbass@hotmail.com.

Received: 5 October 2022; Accepted: 28 February 2023
Published online: 27 March 2023

References

1. Farhat, H. Chapter 9—failure analysis. In Farhat, H., editor, *Operation, Maintenance, and Repair of Land-Based Gas Turbines* (Elsevier, 2021).
2. Farshad, M. Chapter 2-failure investigation of plastic pipes. In Farshad, M., editor, *Plastic Pipe Systems*, pp. 28–25 (Oxford, 2006).
3. Blokdyk, G. *Failure Reporting Analysis And Corrective Action System A Complete Guide* (American Society for Quality Control, West Wisconsin, 2020).
4. Adel, M. *et al.* Early damage detection of fatigue failure for rc deck slabs under wheel load moving test using image analysis with artificial intelligence. *Eng. Struct.* **246**, 1130–1150 (2021).
5. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. & Dean, J. Distributed representations of words and phrases and their compositionality. In *An overview and empirical comparison of natural language processing (NLP) models and an introduction to and empirical application of autoencoder models in marketing* (eds Burges, C. J. *et al.*) (Curran Associates Inc, USA, 2013).
6. Nota, G., Postiglione, A., Postiglione, A. & Carvello, R. Text mining techniques for the management of predictive maintenance. *Proc. Comput. Sci.* **200**, 778–792 (2022).
7. Li, S., You, M., Li, D. & Liu, J. Identifying coal mine safety production risk factors by employing text mining and bayesian network techniques. *Process Saf. Environ. Prot.* **162**, 1067–1081 (2022).
8. Liu, L., Kang, J., Yu, J., & Wang, Z. *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*, 30–31 October 2005.
9. Galvao, R. *et al.* A variable elimination method to improve the parsimony of mlr models using the successive projections algorithm. *Chemom. Intell. Lab. Syst.* **92**(1), 83–91 (2008).
10. Derksen, S. & Keselman, H. Backward forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *Br. J. Math. Stat. Psychol.* **45**(2), 265–282 (1992).
11. Centner, V. *et al.* Elimination of uninformative variables for multivariate calibration. *Anal. Chem.* **68**(21), 3851–3858 (1996).
12. Mehmood, T., Liland, K., Snipen, L. & Sæbog, S. A review of variable selection methods in partial least squares regression. *Chemom. Intell. Lab. Syst.* **118**, 62–69 (2012).
13. Guney, A., Bozdogan, H. & Arslan, O. Robust model selection in linear regression models using information complexity. *J. Comput. Appl. Math.* **398**, 1 (2021).
14. Liu, W., Wang, Z., Zeng, N., Alsaadi, F. & Liu, X. A pso based deep learning approach to classifying patients from emergency departments. *Int. J. Mach. Learn. Cyber.* **12**, 1939–1948 (2021).
15. Li, H. *et al.* A generalized framework of feature learning enhanced convolutional neural network for pathology-image-oriented cancer diagnosis. *Comput. Biol. Med.* **151**, 106265 (2022).
16. Sivanandam, S. & Deepa, S. *Introduction to Genetic Algorithms* (Springer, Berlin, Germany, 2008).
17. Janikow, C. Z. A knowledge-intensive genetic algorithm for supervised learning. *Mach. Learn.* **13**, 189–228 (1993).
18. Chauhan, A., Agarwal, A. & Sulthana, R. Genetic algorithm and ensemble learning aided text classification using support vector machines. *In. J. Adv. Comput. Sci. Appl.* **12**, 1 (2021).
19. Lei, S. *A feature selection method based on information gain and genetic algorithm, 2012*. In *Paper presented at the international conference on computer science and engineering*, 23–25 March 2012.
20. Uysal, A. & Gunal, S. Text classification using genetic algorithm oriented latent semantic features. *Expert Syst. Appl.* **41**, 5938–5947 (2014).
21. James, G. & Hastie, T. The error coding method and picts. *J. Comput. Graph. Stat.* **41**, 377–387 (1998).
22. Basu, A., Walters, C., & Shepherd, M. Support vector machines for text categorization, 2003. in Paper presented at the 36rd annual hawaii international conference, 23–25 March 2003.
23. Mayor, S., & Pant, P. Document classification using support vector machine. *Int. J. Eng. Sci. Technol.***4** (2012).
24. Rahman, S., Mutalib, S., Khanafi, N., & Ali, A. *Exploring feature selection and support vector machine in text categorization, 2013.* In *Paper presented at the 16rd international conference on computational science and engineering*, 3–5 December 2013.
25. Noormanshah, W., Nohuddin, P. & Zainol, Z. Document categorization using decision tree: Preliminary study. *Int. J. Eng. Technol.* **7**, 437–440 (2018).
26. Aggarwal, C. & Zhai, C. A survey of text classification algorithms. In *Mining Text Data* (ed. Aggarwal, C.) (Springer, Boston, MA, 2012).
27. Suresh, A. & Bharathi, C. Sentiment classification using decision tree based feature selection. *Int. J. Control Theory Appl.* **9**, 419–425 (2016).
28. Holland, J. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence* (MIT press, London, England, 1992).
29. Forrest, S. Genetic algorithms: Principles of natural selection applied to computations. *Science* **261**, 872–878 (1993).
30. Rammal, A., Perrin, E., Vrabie, V., Assaf, R. & Fenniri, H. Selection of discriminant mid-infrared wavenumbers by combining a naïve bayesian classifier and a genetic algorithm: Application to the evaluation of lignocellulosic biomass biodegradation. *Math. Biosci.* **289**, 153–161 (2017).
31. Mitchell, M. Genetic algorithms: An overview. *Complexity* **1**, 31–39 (1995).
32. Yangn, M., Yang, Y. & Su, T. An efficient fitness function in genetic algorithm classifier for landuse recognition on satellite images. *Sci. World J.* **1**, 1 (2014).
33. Ranjini, A. & Zoraida, B. Analysis of selection schemes for solving job shop scheduling problem using genetic algorithm. *Int. J. Res. Eng.* **2**, 775–779 (2013).
34. Picek, S. & Goluba, M. Comparison of a crossover operator in binary-coded genetic algorithms. *WSEAS Trans. Comput.* **9**, 1064–1073 (2010).
35. Goncalves, J., Mendes, M. & Resende, M. A hybrid genetic algorithm for the job shop scheduling problem. *Eur. J. Oper. Res.* **167**, 77–953 (2005).
36. Hinterding, R. Gaussian mutation and self-adaption for numeric genetic algorithms, 1995. Paper presented at the ieee international conference on evolutionary computation (1995).
37. Deep, K. & Thakury, M. A new mutation operator for real coded genetic algorithms. *Appl. Math. Comput.* **193**, 211–230 (2007).
38. Papineni, K., Toubakh, S., Ward, T., & Zhu, W. *Bleu: A method for automatic evaluation of machine translation, 2002*. in *Paper presented at the 17rd annual meeting on association for computational linguistics,* 07–12 July 2002.
39. Gunawan, D., Sembiring, C. & Budiman, M. The implementation of cosine similarity to calculate text relevance between two documents. *J. Phys. Conf. Ser.* **978**, 1 (2018).
40. Ezukwoke, K., Toubakh, H., Hoayek, A., Batton-Hubert, M., Boucher, X., & Gounet, P. *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*, 23–27 August 2021.
41. Kamal, M., Barakbah, A., & Mubtadai, N. *Temporal sentiment analysis for opinion mining of asean free trade area on social media, 2016*. In *Paper presented at the international conference on knowledge creation and intelligent computing* pp. 15-17 (2016).

42. Bharti, K. & Singh, P. Hybrid dimension reduction by integrating feature selection with feature extraction method for text clustering. *Expert Syst. Appl.* **42**, 3105–3114 (2015).
43. Nawangsari, R. P., Kusumaningrum, R. & Wibowo, A. Word2vec for indonesian sentiment analysis towards hotel reviews: An evaluation study. *Proc. Comput. Sci.* **157**, 360–366 (2019).
44. Kristiyanti, D., & Wahyudi, M. Feature selection based on genetic algorithm, particle swarm optimization and principal component analysis for opinion mining cosmetic product review, 2017. In *Paper presented at the 5rd international conference on cyber and IT service management*, 08–10 August 2017.

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.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to A.R.