applied sciences

MDPI

Article

Classification of Events in Selected Industrial Processes Using Weighted Key Words and K-Nearest Neighbors Algorithm

Mateusz Walczak, Aneta Poniszewska-Marańda * and Krzysztof Stepień

Institute of Information Technology, Lodz University of Technology, Al. Politechniki 8, 93-590 Lodz, Poland; mateusz.walczak@p.lodz.pl (M.W.); krzysztof.stepien@dokt.p.lodz.pl (K.S.)
* Correspondence: aneta.poniszewska-maranda@p.lodz.pl

Abstract: The problem of classifying events in the industry is related to a large amount of accumulated text data including, among others, communication between the company and the client, whose expectations regarding the quality of its service are constantly growing. The currently used solutions for handling incoming requests have numerous disadvantages; they imply additional costs for the company and often a high level of customer dissatisfaction. A partial solution to this problem may be the automation of event classification; for example, by means of an expert IT system. The presented work proposes the solution to the problem of classifying text events. For this purpose, textual descriptions of events were used, which were collected for many years by companies from many different industries. A large part of text events are various types of problems reported by company customers. As part of this work, a complex text-classification process was constructed by using the *K-Nearest Neighbors* algorithm. The demonstrated classification process uses two novel proposed mechanisms: the *dynamic extension of stop list* and *weighted keywords*. Both of the mechanisms aim to improve the classification performance by solving typical problems that occur when using a fixed stop list and a classical keyword extraction approach by using TF or TF-IDF methods. Finally, the *Text Events Categorizer* system that implements the proposed classification process was described.

Keywords: text classification; keyword extraction; text-feature extraction; K-NN algorithm; N-gram method

Citation: Walczak, M.; Poniszewska-Marańda, A.; Stepień, K. Classification of Events in Selected Industrial Processes Using Weighted Key Words and K-Nearest Neighbors Algorithm. *Appl. Sci.* **2023**, *13*, 10334. https://doi.org/10.3390/app131810334

Academic Editor: Pengjie Ren

Received: 8 August 2023
Revised: 6 September 2023
Accepted: 13 September 2023
Published: 15 September 2023

1. Introduction

Over the last decades, the computerization of almost all areas of social life has been progressing. Increasing the computing power of computers and speeding up the Internet significantly influenced the amount of data sent and archived by enterprises from many different industries. Faced with growing piles of data, humanity has been forced to ask itself questions about possible ways to use the collected information in the most useful way.

In response to such questions, more and more complex algorithms are being constructed to analyze large datasets. Only a few dozen years ago did the then technological state not allow for a full learning process to be carried out in a relatively short time for a human being. Therefore, algorithms such as neural networks have long lingered in the minds of mathematicians and computer scientists as a more theoretical than practical solution. At present, day by day, the computational capacity of modern computers is becoming more and more adequate for the needs of various types of intelligent algorithms.

Companies collect a wide variety of data. However, the subjects of our work are only textual data. Events occurring in the company are most often described in the language of people and thus use text not numbers, which are much easier to interpret and understand with a calculating machine, which is a computer.

Every modern company has a customer contact unit. All communication with the client, his requests, claims, ideas and comments, must be documented and archived by the company for possible later verification. In large corporations, the number of text events

accumulated in this way can give many employees a headache. How to use large text datasets to save time and money? There is no single correct answer to this question.

The aim of the presented work is to demonstrate that text classification can effectively address numerous challenges associated with the management of extensive sets of textual data collected by enterprises from various industries. Currently, existing approaches to text-event classification often rely on human resources to assign events to their respective categories. Regardless of whether it is the customer, an employee or an external company responsible for event classification, this human involvement can introduce various issues, including additional costs, communication challenges and human errors. These problems, if left unaddressed, can lead to customer dissatisfaction. Therefore, the most effective solution is to automate the event-classification process to minimize or eliminate human involvement entirely. Section 4 attempts to present a system capable of automatically classifying texts as an improvement to customer communication processes by categorizing the content sent by the customer.

The main objective of this work is to create an expert system that allows for the classification of text events from various industrial processes. The implementation of such an expert system for the automatic categorization of new events is a very complex issue. The construction of the system began with designing a process for classifying text events, and then its algorithmic and mathematical description was created. To achieve this goal, a universal system for the classification of text events was implemented, which for properly structured datasets from various industries is characterized by a satisfactory level of efficiency and quality of classification. The proposed software is a desktop system implemented with Java technology using a JavaFX library.

The proposed text-classification process consists of four main steps. The first of them is the preparation of each text by building its model, and the second stage is the determination of keywords for the examined set of texts. Then, all the developed features for each of the texts from the set should be extracted in order to be able to proceed to the proper classification. The described text classification method is based on the use of distance metrics and similarity measures for the purpose of comparing individual pairs of texts using the K-Nearest Neighbors algorithm.

The main contributions of this paper are as follows:

1. Designs a complex text-classification process using the K-Nearest Neighbors algorithm;
2. Proposes two novel mechanisms: the *dynamic extension of stop list* and *weighted keywords*, which address typical issues that arise when using conventional keywords and a fixed stop list;
3. Presents an expert system that allows for the automatic categorization of text events of various types, originating from various industrial processes.

This paper is structured as follows: Section 2 contains a polemic on the issues of event classification in industry and shows the text classifier as a potential solution to some of the problems related to it. Section 3 describes the entire text-classification process with a detailed explanation of the two proposed mechanisms: the dynamic extension of stop list and weighted keywords. Section 4 describes the system architecture, data layer and logical layer, while Section 5 deals with the analysis of text classification results, carried out by using the proposed system on selected datasets originating from various industries.

2. Problems of Event Classification in Industry

The vast majority of enterprises and organizations from various sectors of the economy keep records in which they describe all or at least the most important events occurring in the company's history. Health care units, such as hospitals or clinics, keep medical records describing the course of the disease and treatment of their patients, police archive offices keep files of criminals and IT companies collect documentation of their projects and systems.

A very large part of the collected events is a record of the company's communication with the client. Every thriving company is obliged to run a customer service office: to answer its requests and questions, solve problems and respond to possible ideas for cooperation.

Every act of communication with the customer must be archived. Depending on the form of communication, the documentation may include recordings of telephone conversations, the history of conversations in an email box and also notifications collected in the internal system intended for the service of supplicants. As already indicated in the introduction, the subjects of consideration in our work are only those events that are recorded in text form.

2.1. Customer Expectations

Research conducted in 2018 by the organization Statista Research Department [1] showed that as many as 62% of Americans admitted that they had contacted at least one customer service office in the last month since the survey. According to another study commissioned and conducted by *Microsoft* [2], 54% of consumers worldwide say their expectations for customer service are higher than they were a year ago.

The cited results of the first study illustrate the huge scale of the documentation produced, which lingers on the shelves and disks of many American and international companies. However, it is only when the results of both studies are put together that one can see a picture showing not only the unimaginable number of documented events collected day by day by global companies, but also the ever-increasing demands of customers. After all, customers have always wanted, want and will want communication with the company to be faster, better and as less difficult as possible.

2.2. Currently Used Solutions

The scale of this phenomenon, due to the collection of a huge amount of text data, is very large. It is also worth considering how companies handle individual events and what the process is from the submission of an application, complaint or question by the client to the reaction of the company and formulation of the answer.

The first and probably the most important step in handling an event is classifying it to the appropriate type or category, deciding what the report is about and who should handle it. What, then, are the most popular models of event classification functioning in modern corporations? The most popular models of solving the event-classification problem that are used in modern enterprises are presented in Figure 1.

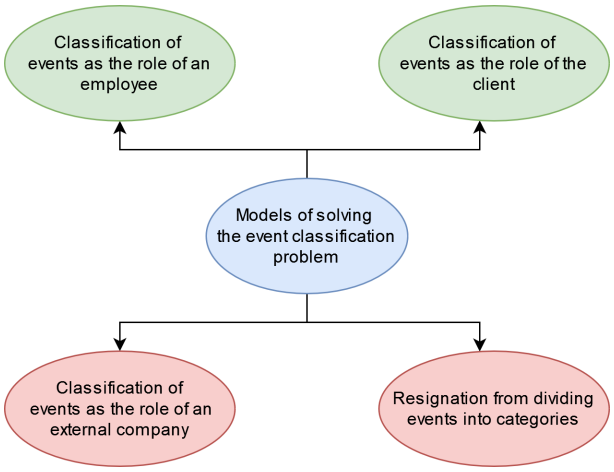

Figure 1. The most popular models of solving the event-classification problem.

In the model of the classification of events as an employee's role, the responsibility for classifying an event lies with the company, more specifically with its employees. In this model, the customer informs the head office of the company's customer service office about

his event by using a clearly defined communication path through the organization (phone call, email, etc.). Then, a customer service employee analyzes the request and, sometimes consulting with colleagues, decides which branch or department of the company should deal with the specific request. Then, things can go in two ways:

- The customer is informed by the company's employee about where he should go with his case. He receives specific instructions and contact details for the selected unit, e.g., a phone number or email address.
- A customer service employee transfers the case to specialists from the appropriate department who will deal with the application and then contact the supplicant themselves.

This model is by far the most popular solution that has dominated the market in various forms.

The model of the classification of events as the role of the client assumes that the responsibility for classifying the event is shifted to the client. While looking for the company's contact details, the client comes across information whereby the only way to report their case is to fill out a form on the website. One of the fields in the form, right after the event description, is the mandatory selection of the problem category from the drop-down list. To submit an application, the customer must choose one of the categories. After selecting it, for many solutions of this type, a new form element appears with another mandatory item, forcing the client to select a subcategory, i.e., make another classification.

Then, the form completed by the customer is sent to the appropriate department of the company that handles events from the category selected by the customer. If the event has been correctly classified by the applicant, the employees of this department analyze the report in order to contact him later. Otherwise, it is the responsibility of the company unit to which the specific incident occurred to redirect the case to the appropriate place.

This solution is often chosen by smaller enterprises that cannot afford to maintain additional employees forming the customer service office. In this model, the event path is theoretically shorter than in the first model; in practice, due to the customer's ignorance, the effect may be completely opposite.

In addition to the two most popular models of solving the problem of event classification, there are also others, such as:

- Classification of events as the role of an external company—involves hiring an external company that performs the function of a customer service office and is responsible for classifying events and their delegation to the appropriate place.
- Resignation from dividing events into categories—a solution designed for smaller companies, for which it is possible to form an expert unit that handles all or a significant majority of events.

2.3. Disadvantages of Currently Used Solutions

The presented-above currently used models of solutions have numerous disadvantages, such as follows:

- *Additional costs*. In the first described model, the responsibility for classifying the event rests with the company. Depending on the structure and size of the company, selected employees must devote part of their time to analyzing and categorizing reports (in the case of smaller companies) or the company may separate a unit or employ several employees whose sole or main task will be to classify new incidents into appropriate "drawers" and directing them to the right place (this solution is most often implemented in larger companies).
  In the first case, employees assigned to the role of a case analyzer lose the time they could spend on their main tasks. In the second case, the company has to fund extra jobs for the people responsible for this process. The conclusion is as follows: the employer will not avoid additional costs because both possibilities generate costs.

- *Customer dissatisfaction*. While the second model of the solution, which assumes that the customer is responsible for classifying his problem, request or inquiry, does not force the company to allocate funds to maintain specialists dealing with the categorization of requests, it has weaknesses of a different kind.
  In this model, the client is forced to specify the type of event he describes. It happens relatively often that the client does not understand what is hidden under the names of the categories, from which he has to choose the right one. As a result, reports go to the wrong sector of the company, whose employees are forced to analyze them and send them to the right place.
- *Communication problems*. The above-mentioned solution models cause many potential interpersonal communication problems. Both communication with an external company that performs the tasks of the customer service office and communication between employees of the company can cause a number of misunderstandings, which translate into longer time taken to handle the event.
- *Human Errors*. The company employee who deals with the classification of applications is only a human being, often guided by his own emotions, susceptible to fatigue and external factors. Even the best and most experienced employees make mistakes when categorizing events that result from human nature and are often unavoidable.

The currently used solutions have many shortcomings. At this point, the question arises: how to automate the classification process so that it does not imply the need for additional hands or nerves and customer dissatisfaction? The answer may be a system that will automatically classify events based on data archived by the company, containing a list of past events with the corresponding category.

The first solution that offers the possibility of the automatic classification of events has appeared on the market. An example of a system whose manufacturers boast of such operation is the Freshdesk [3] application. Unfortunately, the company Freshworks, which is the contractor of the system, does not provide information about the effectiveness of the built-in classification mechanism, nor about the methods and algorithms used for this purpose.

2.4. Related Works

In recent years, a lot of research has been conducted in the field of text classification. The review of related works is divided into two parts. The first part focuses on the use of the *K-Nearest Neighbors* algorithm for text classification. The second part is an analysis of solutions using keyword-based text features similar to those extracted in the proposed classification process.

2.4.1. *K-Nearest Neighbors* Algorithm for Text Classification

K-Nearest Neighbors (KNN) is a supervised learning algorithm. KNN is one of the most popular classifiers used for text classification. Despite its simple design, the KNN algorithm classifier is still used in many modern studies and projects.

Research conducted by the authors of [4] have shown that the K-Nearest Neighbors algorithm is effective for text datasets. In [5], the authors conduct a comparison of different classifiers for text classification and define numerous advantages of the KNN classifier, such as its effectiveness with a large amount of data and robustness with noisy data.

KNN was featured in a survey focused on supervised machine learning techniques for automatic text classification [6]. In this survey, it demonstrated a comparable classification performance to other state-of-the-art classifiers, including Support Vector Machine, AdaBoost and Naive Bayes. Additionally, the K-Nearest Neighbors algorithm underwent examination via a comparative analysis conducted by the authors of [7], where it exhibited a similar classification performance to Linear Regression and Random Forest.

Of course, like other classifiers, KNN is not free of defects. Among the most frequently cited disadvantages of the KNN algorithm are the high computational costs and the difficulty in finding the optimal value of k [4,5,8,9].

The purpose of this paper is not to compare different classifiers but to compare different approaches to feature extraction based on the keywords. Therefore, it was decided to use one of the most popular classifiers, which undoubtedly is the K-Nearest Neighbors algorithm.

2.4.2. Features and Keyword Extraction

Various works illustrate many approaches to constructing a feature vector representing text. This section focuses on the solutions using feature vectors consisting of features based on keywords, because such features constitute the majority of features extracted from texts in the proposed classification process that is described in Section 3.

The authors of [10] decided to use the popular Term Frequency–Inverse Document Frequency (TF-IDF) method to extract features from texts. The TF-IDF method is not without flaws. This method for large text datasets generates feature vectors with a huge number of dimensions. To reduce the number of dimensions of the obtained vectors, the authors used two techniques: Latent Semantic Analysis (LSA) and Linear Discriminant Analysis (LDA). The big advantage of the solution proposed in this paper is the easily controllable number of dimensions in the feature vectors, which is completely dependent on the fixed number of keywords. The presented solution does not require any additional techniques to reduce the number of dimensions of the feature vector.

In [11], the authors compared the TF-IDF and BM-25 methods. The results of the text classification obtained by both techniques are almost the same. The BM-25 method requires much more computation than TF-IDF and, like IDF, generates a very multidimensional feature vector that may contain features that are irrelevant during the classification process.

The authors of [12], who also used the popular TF-IDF method, presented an interesting approach to reducing the dimensions of the feature vector. This approach assumes the removal of keywords for which the calculated TF-IDF value is lower than the assumed threshold. Performing such operations carries the risk that too few keywords will reach the threshold and the final feature vector will have too few dimensions to effectively classify texts. The classification process proposed in this paper effectively eliminates this problem by using a hyperparameter specifying the number of selected keywords.

In [13], the authors proposed an improved TF-IDF algorithm. The authors noticed a disadvantage of the traditional TF-IDF algorithm, which does not take into account the distribution of information between feature items. For this reason, the authors modified the IDF formula to increase the weights of the feature items that frequently appear within a particular class while reducing the weight of the feature items that are evenly distributed across various classes. The same defect, repeated in many approaches to feature extraction by TF-IDF and related methods, was noticed by the authors of this paper. The solution to this problem is the proposed *weighted keywords* method based on the TF algorithm, which is less computationally complex than the TF-IDF algorithm. Therefore, the presented method generates a more balanced, less dimensional feature vector.

3. Classification of Texts Algorithm for Industrial Processes

This section consists of a description of the text-classification process that was implemented in the *Text Events Categorizer* system. The order of discussing the stages of the classification process is consistent with the actual order of performing individual activities and algorithms: (1) the preparation of texts; (2) determining the keywords; (3) the process of feature extraction; and (4) classification, using for example the K-Nearest Neighbors algorithm (Figure 2).

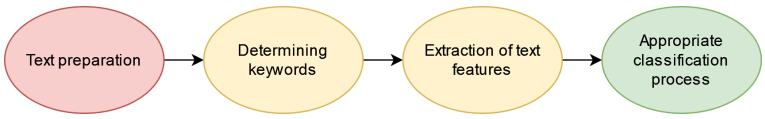

Figure 2. Sequence of stages in text-classification process.

3.1. Text Preparation

In order for a text written in a natural language in the form of a long string of characters to be classified, it must be properly prepared for this. For this purpose, a text model is built to represent it. The preparation of the proposed text model requires the following steps:

- Extracting individual sentences from the text based on punctuation marks "." (dots) found in the text;
- To divide the resulting sentences into single words, using white space (spaces) and other punctuation marks.

The text model obtained in this way is no longer one long string of characters but a vector of sentences, which are vectors of words—short strings of characters, convenient for further analysis.

3.1.1. Removal of Redundant Words

Not all words in the text are meaningful. Every language contains a set of words that are very repetitive and do not add any logical value to the sentences and texts in which they are found. A set of such words for a particular language is called a *stop list*.

The next step in preparing the text model is to remove all the words that are in the stop list for the language in which the text was written. The created Text Events Categorizer system is written in English and designed to classify texts in that language. For this reason, it uses a stop list dedicated to the English language [14].

3.1.2. Stemization of the Text

The final stage on the way to obtaining a fully prepared text model is stemming words. Stemization is the process of reducing words to their root, so these words that are semantically very similar, differ in inflection or come from the same word family are represented by the same string of characters, i.e., the root.

A text model prepared in this way, containing only potentially meaningful words that are the cores of their semantic families, enables the more-effective selection of keywords.

3.2. Determining Keywords

To determine a set of keywords for a given set of texts, the Term Frequency (TF) method was used, consisting of counting the number of occurrences of a given word in all texts.

To determine keywords using this method, we need to perform calculations on a set of all our texts from one category. The result of the calculation is a set of pairs—the word and the number of its occurrences in all documents. This set of pairs should be sorted by the number of occurrences in descending order and n first words selected. The selected n words become keywords.

The above operation is repeated l times, where l is the number of text categories in the analyzed set. The result is l sets of keywords, each set representing a different category. The received keyword vectors have the following symbols:

$$K_1, K_2, \ldots, K_{l-1}, K_l. \tag{1}$$

This set of l keyword vectors has two obvious disadvantages:

- It may happen that one of the words appears in the vectors for each category. This situation can lead to errors during later classification because a word marked as a keyword does not really carry any information.
- Each keyword is equally important, no matter how many vectors it is repeated in. This situation is potentially problematic, because it is very important that a given word is a keyword not only in its category.

The following subsections contain the description of the proposed algorithms and solutions aimed at eliminating and preventing the above-mentioned problems.

3.2.1. Dynamic Extension of Stop List

Let there be a keyword k satisfying the following conditions:

$$k \in K_1, k \in K_2, \ldots, k \in K_{l-1}, k \in K_l. \tag{2}$$

The above set of conditions means that the keyword k belongs to all the keyword vectors for each category. In such a situation, it does not carry any significance for the subsequent feature-extraction process, and thus also for the categorization itself.

Therefore, the word k is removed from all keyword vectors and dynamically appended to the stop list for the currently used dataset. As a result, the size of the keyword vectors (the number of words in the vector) is $n - 1$, where n is the initially assumed number of keywords. The next step is to fill in the missing space by adding the next keyword occupying the next position in the sorted set of pairs for each category.

The above algorithm is performed for each keyword that meets the condition (2). The whole process ends when there are no more words in the keyword vectors that meet this condition, and the size of each vector is the expected value n.

3.2.2. Weighted Keywords

Weighted keywords are an attempt to solve the second problem described in 3.2. Weighted keywords are a set of pairs: a keyword and weight (a floating-point value). They consist of the same set of keywords determined earlier, which is enriched with a weight calculated in accordance with the developed formula:

$$W_i = \left(1 - \frac{N_{W_i \in K_l}}{l-1}\right)^2, \tag{3}$$

where W_i—weight of ith keyword, l—number of categories and $N_{W_i \in K_l}$—number of keyword categories (other than our own) in which the ith keyword occurs.

An important conclusion from the analysis of the above formula is that keyword weights can only reach values in the range $\langle 0; 1 \rangle$.

In order to better understand the formula, an example will be presented: $l = 3$ and $n = 3$, and the designated keyword vectors have the following form:

$$K_1 = \{\text{“}autumn\text{”}, \text{“}tail\text{”}, \text{“}cat\text{”}\}, \tag{4}$$

$$K_2 = \{\text{“}spring\text{”}, \text{“}tail\text{”}, \text{“}dog\text{”}\}, \tag{5}$$

$$K_3 = \{\text{“}summer\text{”}, \text{“}tail\text{”}, \text{“}cat\text{”}\}. \tag{6}$$

The calculation of the weight values for the selected keywords from the above set is as follows. For the word “autumn”, the following value was obtained:

$$W_{autumn} = \left(1 - \frac{0}{2}\right)^2 = 1 \tag{7}$$

The word “autumn” appeared only in one “own” category, so it has the greatest possible weight. For the word “cat”:

$$W_{cat} = \left(1 - \frac{1}{2}\right)^2 = 0.25 \tag{8}$$

The word “cat” appeared in one additional category (two in total). For the word “tail”:

$$W_{tail} = \left(1 - \frac{2}{2}\right)^2 = 0 \tag{9}$$

The word "tail" appeared in all categories; therefore, it should be considered that it has no meaning, and its weight is equal to 0. If the *dynamic extension of stop list* algorithm was used before determining the keyword weights, the word "tail" should be replaced with a different one in each of the vectors.

The discussed example very well illustrates the effectiveness and sense of using both solutions discussed in the right order. After determining the keywords, we need to perform the process of removing repeated keywords in each vector of the *dynamic extension of stop list* in order to later calculate the weights for all the keywords obtained in this way. This sequence of actions aimed at determining keywords along with their weights was implemented in the Text Events Categorizer system.

3.3. Extraction of Text Features

After the text models are prepared and the keyword vectors are determined, it is time to extract the characteristics of the text. The text feature is the result of a function that, depending on its type, returns a number (a floating-point value) or a string of characters (text, string). The first type of feature will be called numeric features and the second will be called string features.

3.3.1. Numerical Text Features

The proposed numerical features are based on the occurrence of keywords. The functions of numerical features whose extraction was implemented in the Text Events Categorizer system are as follows:

The adopted designations:

- T_i—set of words that is the field of study of a given feature.
- K—set of keywords consisting of l keyword vectors $K_1, K_2, \ldots, K_{l-1}, K_l$.
- $N_{K_i \in T_i}$—number of occurrences of the elements of the set K_i in the set T_i. Regarding the meaning of the weighted keywords, it will be the sum of the products of the number of occurrences of individual elements of the K_i set in the T_i set and the corresponding weights.
- $C_i(T_i, K)$—feature function value.

Understanding individual features depends on whether "normal" keywords or *weighted keywords* are used to extract the features; therefore, each feature has a basic nomenclature appropriate for the use of "normal" keywords and additional ones in parentheses, intended for the use of *weighted keywords*.

C_1—number of occurrences (the sum of the product of the weights and occurrences) of keywords from a given category in the entire text.

The feature describing the number of occurrences (the sum of the product of the weights and occurrences) of keywords from the *i*th category occurring throughout the text is given by the formula

$$C_1(T_1, K_i) = N_{K_i \in T_1}, \tag{10}$$

where T_1—the set of all words in the text.

In order to explain the complicated naming and interpretation of the above formula, which depend on the type of keywords, an example of calculating the value of the C_1 feature is presented as follows:

Let the set of keywords K_i have the following form:

$$K_i = \{\text{“}virus\text{”}, \text{“}disease\text{”}, \text{“}plague\text{”}, \text{“}annihilation\text{”}\}, \tag{11}$$

and the set T_1 being the field of study (the set of all the words in the text) is as follows:

$$\begin{aligned} T_1 = \{&\text{“}virus\text{”}, \text{“}destroys\text{”}, \text{“}everything\text{”}, \text{“}road\text{”}, \\ &\text{“}plague\text{”}, \text{“}virus\text{”}, \text{“}cause\text{”}, \text{“}disease\text{”}\}, \end{aligned} \tag{12}$$

At the beginning, the extraction variant using "normal" keywords is described. The occurrence of individual elements of K_i set in the T_1 set is analyzed as below:

- "virus"—occurs two times;
- "disease"—occurs once;
- "plague"—occurs once;
- "annihilation"—never occurs.

In this variant, the value of the C_1 feature will simply be the sum of the keyword occurrences. After adding all the occurrences, we obtain

$$C_1(T_1, K_i) = N_{K_i \in T_1} = 2 + 1 + 1 + 0 = 4. \quad (13)$$

The next calculations were carried out by using the second, more complicated method of extracting numerical features by using *weighted keywords*. Let the pairs of keywords from the K_i set and the weights calculated for them look like this:

$$K_w = \{(\text{“}virus\text{”}, 0.25), (\text{“}disease\text{”}, 1), (\text{“}plague\text{”}, 0.25), (\text{“}annihilation\text{”}, 1)\}, \quad (14)$$

In this case, in accordance with the previously presented description, the sum of products of the number of occurrences of individual elements of the K_w set in the T_1 set and the corresponding weights should be calculated as

$$C_1(T_1, K_w) = N_{K_w \in T_1} = 2 \cdot 0.25 + 1 \cdot 1 + 1 \cdot 0.25 + 0 \cdot 1 = 0.5 + 1 + 0.25 + 0 = 1.75. \quad (15)$$

C_2—the number of occurrences (the sum of the product of the weights and occurrences) of keywords from a given category in the first three sentences of the text.

The feature describing the number of occurrences (the sum of the product of the weights and occurrences) of the keywords from *i*-th category appearing in the first three sentences of the text is given by a similar formula as the feature C_1:

$$C_2(T_2, K_i) = N_{K_i \in T_2}, \quad (16)$$

where T_2—set of words appearing in the first three sentences of the text.

C_3—the ratio of the number of occurrences (the sum of the product of the weights and occurrences) of keywords from a given category in the entire text to the total number of words in the text.

The feature describing the ratio of the number of occurrences (the sum of the product of the weights and occurrences) of keywords from the *i*th category occurring in the entire text to the total number of words occurring in the text (except for those on the stop list) is given by the formula

$$C_3(T_3, K_i) = \frac{N_{K_i \in T_3}}{|T_3|}, \quad (17)$$

where T_3—the set of all the words in the text and $|T_3|$—the number of elements (words) of the set of all the words in the text.

3.3.2. Normalization of Numerical Text Features

The extracted numerical features reach very different floating-point values. In order to enable their effective interpretation in the proper classification process, all features should be normalized, bringing them to a specific, clearly defined space (range).

One of the most popular normalization methods used when building one's own system is to project all the feature values into the range $\langle 0; 1 \rangle$. To perform this, divide the obtained values of a given feature for each of the training texts by the maximum value of this feature, achieved for one of these texts, according to the formula

$$N(C_i) = \frac{C_i}{C_i^{max}}, \tag{18}$$

where $N(C_i)$—the value of the feature normalization function (normalized feature), C_i—the feature value before normalization and C_i^{max}—the maximum value reached by the feature.

3.3.3. Chain Features of the Text

Only one string feature is implemented in the *Text Events Categorizer* system. This feature will not use the keywords on which all the proposed numerical text features are based.

In addition to the notations introduced in the section on the numerical features of the text, an additional notation was adopted:

- W_T^n—n-th most frequent item in the set T (for $n = 1$ it will be the most frequent item, for $n = 2$ it will be the second most common item, etc.).

C_4—the five most common words in the text.

The feature that returns a string is the set of the five most common words in the entire text, separated by spaces, formally described as follows:

$$C_4(T_4) = W_T^1 + W_T^2 + W_T^3 + W_T^4 + W_T^5, \tag{19}$$

where T_4—the set of all words in the text, and the "+" operation means the concatenation of two strings of characters and adding a space (spacing, white space) between them.

3.3.4. Extracted Feature Vector as a Set of Numbers and Strings

The extracted feature vector consists of the values obtained by all four features described in the previous subsections (C_1–C_4). The first three features, which are numerical features, are extracted for each of the l sets of keywords, where l is the number of categories by which we classify the texts. The calculated value of each feature from C_1 to C_3 is thus described by one of the different floating-point values. In the case of the string feature C_4, the situation is much less complicated. The value of the extracted feature C_4 is one string of characters.

From the above considerations, it is clear that the end result of the entire process of extracting text characteristics is a vector consisting of $3l+1$ elements, including:

- Those from $3l$ of floating-point values resulting from the extraction of numerical features (from C_1 to C_3);
- Those from the one string, which is the value of the extracted string feature C_4.

The vectors of the characteristic features of the texts obtained in this way make up the next stage of classification. The last part of this section discusses how the extracted feature vector is interpreted and used in text classification by the *K-Nearest Neighbors* algorithm.

3.4. Appropriate Classification Process

The classification of texts is a process that involves comparing them with each other many times. However, in order for such a comparison to be possible, the texts must be properly prepared and modeled, and a full process of the extraction of their characteristics must be carried out. As a result of these steps, described in the previous subsections, we obtain vectors consisting of numbers and strings that represent each of the texts.

But how do we compare vectors consisting of different types of data? The answer is to perform a separate analysis for both types of data. The methods used to compare the calculated numerical and string features are described in the following subsections.

3.4.1. Distance Metrics

Selected distance metrics were used to compare the numerical elements of the text-feature vectors. Distance metrics are functions that take the coordinate vectors of two points as arguments. In the case of text classification, the points suspended in the $3l$ dimensional

classification space (l is the number of categories by which we classify texts) are texts, or more precisely, their vector representations [15,16].

Two distance metrics were implemented in the Text Events Categorizer system:

- *Euclidean Metric*—to calculate the distance $d_e(x, y)$ between two points x, y, calculate the square root of the sum of squares of the differences in coordinate values with the same indices, according to the formula

$$d_e(x, y) = \sqrt{(y_1 - x_1)^2 + \cdots + (y_n - x_n)^2} \tag{20}$$

- *Street metric (Manhattan, urban)*—to calculate the distance $d_m(x, y)$ between two points x, y, calculate the sum of the absolute values of the differences in the coordinates of the points x and y according to the formula

$$d_m(x, y) = \sum_{k=1}^{n} |x_k - y_k| \tag{21}$$

At this point, it should be emphasized again that for distance calculations using metrics, only floating-point values are taken into account—the first $3l$ of vector elements. Strings, i.e., $(3l + 1)$th elements of the extracted feature vectors will be compared by using another method dedicated to strings, which is presented in the next part of this work.

3.4.2. Measures of Text Similarity

The last $(3l+1)$ elements of the text vectors are strings (text strings). To compare two strings, it was decided to use the measure of text similarity—the *n-grams* method:

The *n-grams* method determines the similarity of text strings s_1, s_2 based on the number of common n-element substrings. It is described by the formula

$$sim_n(s_1, s_2) = \frac{1}{N - n + 1} \sum_{i=1}^{N-n+1} h(i), \tag{22}$$

where N is the number of elements of the longer string s_1, s_2:

$$N = max\{N(s_1), N(s_2)\}, \tag{23}$$

and $h(i) = 1$ if the n-element subsequence starting at the i-th position in s_1 occurs at least once in s_2; otherwise, $h(i) = 0$.

There are the following variants of the *n-grams* [17] method:

- *1-grams*, or *unigrams*;
- *2-grams*, or *bigrams*;
- *3-grams*, or *trigrams*.

In the proposed solution, two of the above methods were implemented: the *bigrams* method and the *trigrams* method. The analysis of the above formulas shows that the values obtained by using the *n-grams* methods are in the range $[0; 1]$, where 0 means no similarity at all and 1 means full similarity.

3.4.3. Final Result of Text Comparison

There is a very significant difference in meaning between the values returned by the similarity measures and distance metrics.

The distance and similarity functions, in terms of understanding the values they return, are actually inverse functions. A distance value of zero means full similarity, and zero similarity means maximum distance. So, to obtain the final distance between the two texts t_1 and t_2 while completing their comparison, the following should be performed:

$$D(t_1, t_2) = d(t_1^n, t_2^n) + (1 - sim_n(t_1^s, t_2^s)), \tag{24}$$

where $D(t_1, t_2)$ is the final distance between the texts, t_1^n and t_2^n are the vectors of the extracted numerical features of the texts, t_1^s and t_2^s are the strings constituting the last extracted feature, $d(t_1^n, t_2^n)$ is the distance between the numerical features of the texts and $sim_n(t_1^s, t_2^s)$ is the calculated similarity between the string features of the texts.

The distance D obtained in this way expresses the degree of differentiation of the two texts. The higher the value of D, the more the compared texts differ from each other. However, when the distance D is close to 0, the texts can be considered very similar or almost identical.

3.4.4. K-NN Algorithm

K-Nearest Neighbors is a very simple classifier. The k-NN algorithm is sometimes said to be lazy. This is due to the fact that it does not create an internal representation of the training data and starts searching for a solution only when analyzing a specific pattern from the [18,19] test set.

The algorithm stores a set of all the training patterns (in this case, the patterns are text models), against which the distance of the test pattern (the text currently being classified) is calculated, which was defined in the previous subsection—Formula (24) .

The algorithm then selects k training patterns, called neighbors, to which the currently classified pattern has the shortest distance (in other words, these are the training patterns to which the tested test pattern is most similar). The final result—the category to which the analyzed pattern will be assigned—is the most frequent category among the K-Nearest Neighbors. In a situation where two or more categories have the same number of representatives, the winner is the category whereby one of the training texts was closest to the classified text [19–21].

In the Text Events Categorizer system, the value of k depends on the examined set of texts, and specifically on its size. The value of k is calculated by using the formula proposed in [22]:

$$k = \lfloor \sqrt{N} \rfloor, \tag{25}$$

where N is the size of the set of the training texts.

The category determined in this way, by using the *K-Nearest Neighbors* algorithm, is the result of the entire classification process described in this section. At the same time, it is the application's answer to the question: *"Where does the text fit best?"*, which in relation to the previous section of this work can be paraphrased as: *"Who should take care of this event? Where should it go?"*.

The next section of this work concerns the technical aspects of the Text Events Categorizer system. The results of the classification for exemplary collections of texts are discussed in Section 5.

4. Technical Aspects of Text Events Categorizer Approach

The *Text Events Categorizer* system is used to classify the text events of various types, originating from various industrial processes. The system implements the entire classification process, which was described in the previous section. The solution provides the user with a number of possibilities related to the creation, management and use of text datasets. Figure 3 presents the use case diagram for the system.

The functioning of the system is based on many complex operations on large text datasets, which in the Text Events Categorizer are called *datasets*. The system should be able to create a dataset by defining the following properties: (1) names; (2) password to access and manage the dataset; and (3) a set of texts by attaching a file in *.xlsx* format, containing a set of text data with categories in a clearly defined form: the file should consist of two columns, the first of which contains the category of a given text and the second full of its content.

The *Teach dataset* function initiates the preparation of the texts for classification; i.e., for each of the texts belonging to a given dataset, it performs all the activities described

in Section 3, except for the proper classification process, i.e., (1) text preparation, (2) the determination of keywords and (3) the extraction of text features. After this process, which is called *learning* in the system, the dataset goes from *unlearned* to *learned*. The next two functions *perform test classification* and *classify new text* are possible only for datasets in the *trained* state.

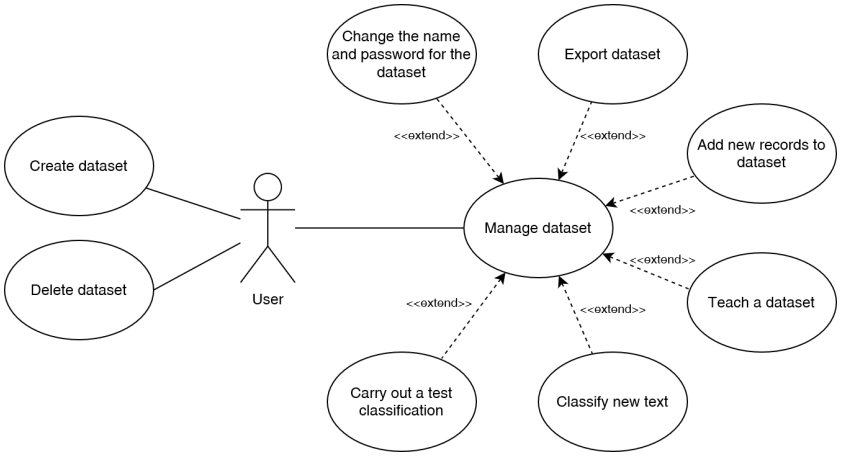

Figure 3. Use case diagram for *Text Events Categorizer* system.

It is worth mentioning here that if the user uses *add new records to the dataset* option, after having been prepared for classification (training), it will return to the *unlearned* state and the learning process will have to be repeated.

Perform test classification—for the learned dataset, the system should provide the ability to perform a test classification of all texts. Such a process assumes the proper classification for each of the texts from a given dataset. For this purpose, for each of them, the system should perform the following series of steps: (1) exclude the text from the training set (remove it from the classification space), (2) carry out the proper classification process and (3) check the correctness of the classification.

Classify new text—the most important function for the *Text Events Categorizer* system, which assumes that for a dataset in the *trained* state, the system fully classifies the text entered by the user, and it goes through all the stages of the classification process described in Section 3. The user then evaluates the correctness of the displayed category by choosing one of three answers to the question asked by the system: *Is this category correct?*: (1) *yes*; (2) no—in this case, the program asks the user to enter the correct category for the entered text; and (3) *I do not know*. Depending on the user's response, the system responds accordingly. If the answer is *yes* or *no*, i.e., learning about the correct category for the text, the text is appended to the set of texts, which remains in the *learned* state. If the user selects *I do not know*, the text will not be saved as a new record.

The architecture of the Text Events Categorizer system is presented in Figure 4.

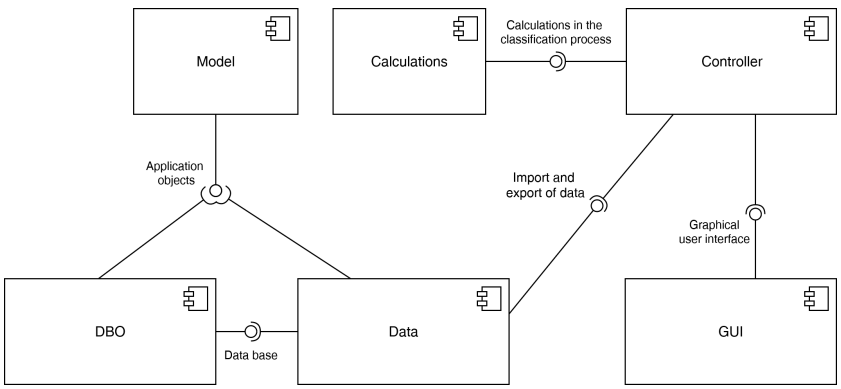

Figure 4. Component diagram for *Text Events Categorizer* system.

The dataset in the system's database is represented by the following attributes: (1) `id`—the dataset identification number; (2) `beingManagedNow`—the flag specifying whether there is a user who currently manages a given dataset; (3) `learned`—the flag specifying the current state of the dataset; (4) `name`—the dataset name; (5) `password`—the dataset password; (6) `records`—the number of texts (records) in a given dataset; and (7) `size`—the memory size in bytes occupied by the dataset.

The single text representation in the database includes (1) `id`—the text identification number; (2) `category`—the category object to which a given text belongs, consisting of its identification number and name; (3) `features`—the array containing the extracted numeric features of a text when it is in the *trained* state (when the text is in the *unlearned* state, the `features` array is empty); (4) `textFeatures`—the array containing the extracted string features of the text (similarly, the `textFeatures` attribute also remains as an empty array as long as the text is in the *unlearned* state); and (5) `sentences`—the array of individual sentences of the text, represented by a table of the words occurring in them.

The classes responsible for the classification process are presented in Figure 5.

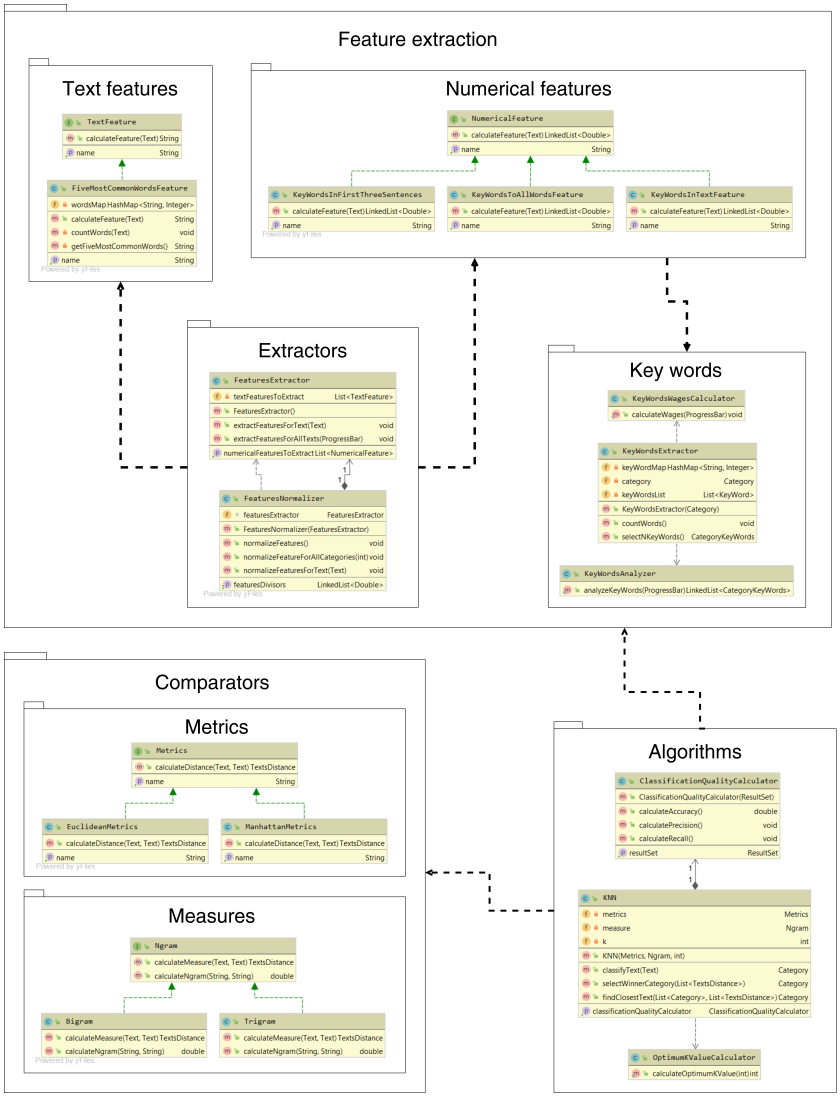

Figure 5. Class diagram for classification process in *Text Events Categorizer* system.

One of the functions of the *Text Events Categorizer* system is the *train dataset* used to extract all the features from all the texts in the dataset. Performing a complete dataset training process allows one to classify text events from that dataset. The interaction overview diagram for the *train dataset* function is shown in Figure 6.

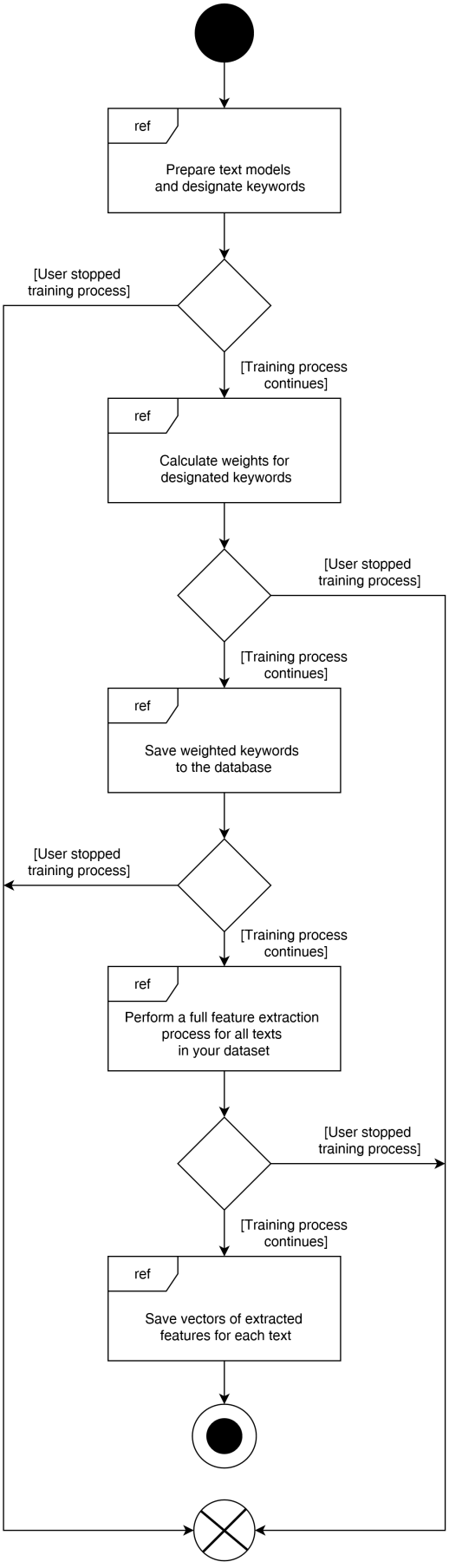

Figure 6. Interaction overview diagram for *train dataset* function in *Text Events Categorizer*.

5. Analysis and Verification of Text Events Categorizer Approach—Classification Results

This section discusses the results of the classification performed on three selected datasets by using the *Text Events Categorizer* system. Firstly, the measures of the classification quality are defined, then the selected datasets are used to test if the system is working and finally the results of the experiments are presented and analyzed.

5.1. Classification Quality Measures and Selected Datasets

In order to assess the quality of the classification, the three most popular quality measures were used:

- Measure of effectiveness (*accuracy*):

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{26}$$

- Measure of precision (*precision*):

$$P = \frac{TP}{TP + FP} \tag{27}$$

- Sensitivity measure (*recall*):

$$R = \frac{TP}{TP + FN} \tag{28}$$

The following notations were adopted in the above formulas:

- *TP* (*True Positive*) is the number of texts correctly classified as a given category;
- *TN* (*True Negative*) is the number of texts correctly classified as other than the examined category;
- *FP* (*False Positive*) is the number of texts incorrectly classified as the examined category;
- *FN* (*False Negative*) is the number of texts incorrectly classified as other than the examined category.

In order to conduct the classification tests by using the *Text Events Categorizer* system, it was decided to select three sets of text data, which were given the following designations along with names referring to the companies they come from:

- set Z_1—*Experian Informatic Solutions*;
- set Z_2—*Bank Of America*;
- set Z_3—*Reuters Articles*.

The first two datasets *Experian Informatic Solutions* and *Bank Of America* come from the *Consumer Complaint Database* [23] and are closely related to the issues of work discussed in Section 3. They contain reports from customers of the two above-mentioned American companies, which were sent to customer service offices. The sets Z_1 and Z_2 are therefore sets of text events, perfectly fitting into the problem described in Section 2.

The last *Reuters Articles* dataset consists of news articles from the *Reuters* news agency [24]. The set Z_3 is a completely different set in relation to the other two sets. Press articles are characterized by a completely different language and structure than reports received by customer service offices. The Z_3 set was chosen precisely because of its difference, which will allow us to verify the quality of classification for text data of a completely different type, coming from a market sector very distant from the activities of the companies from which the data constituting the Z_1 and Z_2 sets come from.

Set Z_1—*Experian Informatic Solutions*

The Z_1 dataset contains 4274 texts divided into three categories. Due to the long names of individual labels (category names), it was decided to introduce the following designations to represent each of them:

- Category K_1—*credit-reporting company's investigation*;

- Category K_2—*incorrect information on credit report*;
- Category K_3—*problem with a credit-reporting company's investigation into an existing problem*.

The number of texts for each of the three labels is listed in Table 1.

Table 1. Structure of *Experian Informatic Solutions* dataset.

Category	Number of Texts
K_1	981
K_2	2903
K_3	390
In total	4274

Set Z_2—*Bank Of America*

The Z_2 dataset consists of 4318 texts divided into three categories. The structure of the *Bank Of America* dataset is presented in Table 2.

Table 2. Structure of *Bank Of America* dataset.

Category	Number of Texts
Mortgage	1870
Credit card	953
Bank account or service	1495
In total	4318

Set Z_3—*Reuters Articles*

The largest set of texts is the Z_3 set, which includes 6290 press articles divided into four groups (Table 3).

Table 3. Structure of *Reuters Articles* dataset.

Category	Number of texts
Earn	3735
Acq (acquirement)	2125
Money supply	97
Trade	333
In total	6290

5.2. Course of Experiments

The experiments began by creating in Text Events Categorizer system for each of the text datasets described in the previous subsection, the corresponding datasets with the same names. The purpose of the experiments was twofold:

- Checking the quality of the Text Events Categorizer system classification for the selected datasets;
- Verifying that the proposed novel mechanisms of the *dynamic extension of stop list* and *weighted keywords* effectively solved the problems that occur with the use of conventional keywords and a fixed stop list. To this end, the impact of using the proposed solutions on the quality of the classification was tested.

The aim of this research was therefore not only to determine the quality of the classification for three sets of data but also to determine what impact the use of the proposed mechanisms in the feature-extraction process has on the quality of the classification (primarily on its effectiveness).

For this purpose, two series of experiments were planned, consisting of carrying out a test classification for each of the three sets of data for which the process of feature extraction was carried out:

1. *Series 1*—using conventional keywords and a fixed stop list (not using the proposed mechanisms);
2. *Series 2*—using the proposed mechanisms of the *dynamic extension of stop list* and *weighted keywords*.

During the experiments in *Series 1*, the Text Events Categorizer system was modified so that it does not use the mechanisms mentioned above. An additional modification needed to perform all the planned tests was the system's calculation of the remaining classification quality measures, precision and sensitivity, and writing their values on the console screen. In all the experiments, the street metric was used to calculate the distance between the texts.

5.3. Obtained Results

The presentation of the results began with the most important and most intuitive measure of quality—effectiveness—and then moved on to the measure of precision and then to the values achieved by the measure of sensitivity.

5.3.1. Classification Efficiency

Table 4 and Figure 7 demonstrate the efficiency of classification for each of the experiments performed. The data show that the lowest classification efficiency values were achieved in both series of experiments for the Z_1 dataset. The efficiency for the Z_2 set was higher than for the Z_1 set but much lower than the classification efficiency obtained for the Z_3 set, which significantly exceeded 90% for both sets of experiments.

Table 4. Classification efficiency results for performed experiments.

Dataset	Series 1 (%)	Series 2 (%)
Experian Informatic Solutions	71.59	72.79
Bank Of America	78.86	84.37
Reuters Articles	93.23	94.63

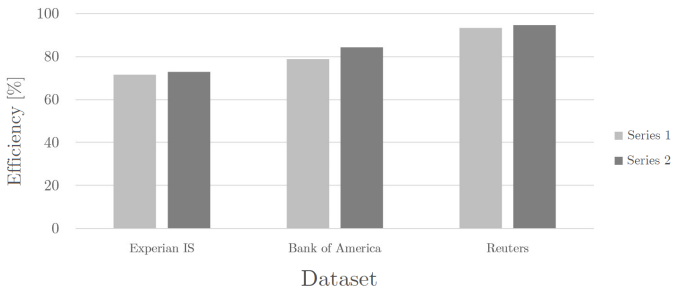

Figure 7. Classification efficiency results for performed experiments.

5.3.2. Classification Accuracy

Table 5 and Figure 8 present the results of the classification accuracy obtained for each category from each dataset for both series of experiments. As in the case of the efficiency results, the best results were obtained for the Z_3 dataset. The accuracy values obtained for the individual categories of the Z_1 and Z_2 sets are at a similar level, and in most cases they range between 70% and 90%. The exception is the experiment using *Series 1* for the category K_3 from set Z_1, for which the classification accuracy was calculated to be zero.

Table 5. Classification accuracy results for performed experiments.

Category	Series 1 (%)	Series 2 (%)
Accuracy results for *Experian Informatic Solutions* dataset		
K_1	74.46	76.13
K_2	71.48	72.48
K_3	0.00	77.78
Accuracy results for *Bank Of America* dataset		
Mortgage	81.70	90.82
Credit card	74.62	72.91
Bank account or service	76.89	84.10
Accuracy results for *Reuters Articles* dataset		
Earn	98.33	98.69
Acq	85.43	88.25
Money supply	95.29	94.51
Trade	96.50	96.90

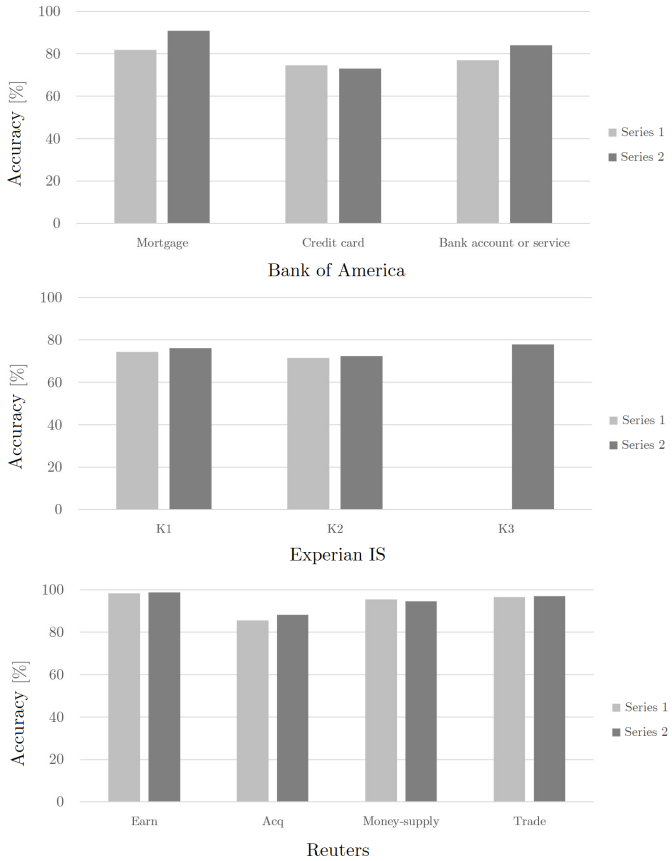

Figure 8. Classification accuracy results for performed experiments for three datasets.

5.3.3. Classification Sensitivity

This subsection presents the calculated classification sensitivity values for each of the labels in the datasets studied (Table 6 and Figure 9) for both series of experiments performed. The calculated classification sensitivity values are definitely the highest for the Z_3 dataset. Worse results, although still higher than 50% for each category, were obtained for the Z_2 dataset. Definitely the lowest classification sensitivity results, not exceeding 30%, were obtained for categories K_1 and K_3 from the Z_1 dataset.

Table 6. Classification sensitivity results for performed experiments.

Category	Series 1 (%)	Series 2 (%)
Sensitivity results for *Experian Informatic Solutions* dataset		
K_1	21.10	25.68
K_2	98.27	98.00
K_3	0.00	3.59
Sensitivity results for *Bank Of America* dataset		
Mortgage	91.93	92.57
Credit card	56.76	79.64
Bank account or service	76.58	77.12
Sensitivity results for *Reuters Articles* dataset		
Earn	91.33	92.96
Acq	97.36	97.93
Money supply	83.51	88.66
Trade	90.99	93.99

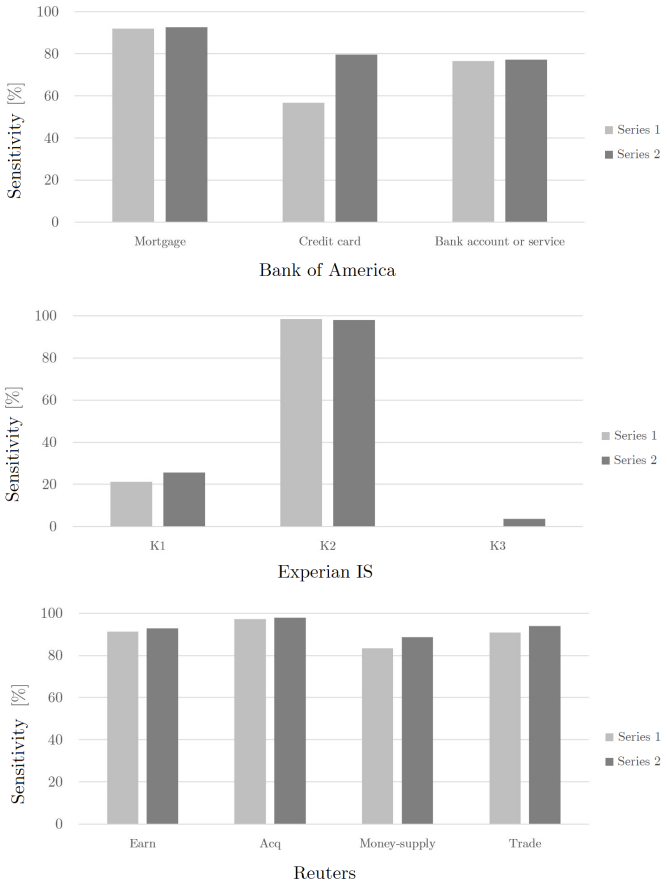

Figure 9. Classification sensitivity results for performed experiments for three datasets.

5.4. Discussion

This subsection discusses the results of the performed experiments, presented in two parts. The first one focuses on the results of the classification efficiency, and the second one focuses on the values achieved by the other two quality measures.

5.4.1. Classification Efficiency

Classification efficiency is by far the most important and also the most intuitive measure of classification quality. It informs us about the percentage of correctly classified texts.

The classification efficiency for all the experiments was above 70%, which is definitely a satisfactory result (Table 4 and Figure 7). The most difficult set of texts for classification turned out to be Z_1, for which the worst classification efficiency was achieved, both in *Series 1* and *Series 2* (values slightly exceeding 70%). The middle place in the hierarchy of classification efficiency with a much better result (about 80% depending on the series) is occupied by the Z_2 set. The best efficiency values, significantly exceeding 90% in both test series, were obtained for the Z_3 set.

Structure of the Set and the Efficiency of Classification

Based on the analysis of the obtained results, it can be seen how the structure of the set affects the effectiveness of classification. For this purpose, it is necessary to look at the names of the categories into which the texts from individual collections were classified.

All three labels for Z_1 are very similar, each containing the expression *credit report*. After reading these three names, we may come to the conclusion that it is practically impossible for a person without proper banking knowledge and knowledge of *Experian Informatic Solutions* procedures and methods to correctly classify the texts in this collection on their own. Such a slight difference in meaning between the names of the categories translates into a very high syntactic similarity between the texts from different labels in this collection, thus making it very difficult to classify. Nevertheless, the *Text Events Categorizer* system coped quite well with the classification task set before it, achieving a high efficiency of over 70% (the analysis of the calculated precision and sensitivity values in the following subsections allows for a better interpretation of the obtained efficiency).

In the case of the Z_2 set, the names of the categories are much more diverse. Despite the fact that they still all concern the banking sector, each of them represents a different part of it. Such a structure of the set implies less syntactic similarity between the texts representing different labels, which is the reason for the higher classification efficiency than in the case of Z_1.

The Z_3 set is completely different from its predecessors. The text data in this collection are press releases, divided into thematic categories. In this case, the texts from individual categories are very different from each other, both in meaning and syntactically. Therefore, *Reuters Articles* are by far the easiest to classify and achieved the best performance results.

Performance in Series 1 vs. Performance in Series 2

The Figure 8 clearly shows that each of the experiments in *Series 2* yielded a higher classification efficiency value than the corresponding experiment in *Series 1*. The difference between the calculated values ranges from 1.2% for the Z_1 set to less than 6% for the Z_2 set. For each set, the difference in the classification efficiency for the *Series 1* and *Series 2* experiments is therefore noticeable.

After considering the research results listed above for both series, a preliminary thesis should be put forward that the use of the proposed novel mechanisms in the extraction process, the *dynamic extension of stop list* and *weighted keywords*, has a positive impact on the efficiency of the classification for each of the analyzed datasets. However, the effectiveness is only one of the components used to assess the quality of the classification.

5.4.2. Accuracy and Sensitivity of Classification

The effectiveness of the classification is often not sufficient to objectively assess the quality of the classification carried out. To obtain a fuller picture of how effective the individual experiments were, we need to use other measures of classification quality—accuracy and sensitivity.

High Efficiency and Low Classification Sensitivity

This part focuses on the values of the accuracy and sensitivity measures for the Z_1 dataset (Tables 5 and 6 and Figures 8 and 9), and we compare them with a relatively high and satisfactory classification efficiency: 71.59% in *Series 1* and 72.79% in *Series 2*.

When we first look at the classification accuracy and sensitivity diagrams for the *Experian Informatic Solutions* dataset, we can easily see that one light gray bar is missing in both diagrams. It turns out that both the accuracy and sensitivity for the K_3 category for the *Series 1* classification have a value of zero. Such values of accuracy and sensitivity measures mean that no text was classified in this category (neither correctly nor incorrectly), which shows that a high classification efficiency does not necessarily imply high quality. For the discussed experiment, the high efficiency of classification is mainly related to the large number of correct classifications of texts into the K_2 category, which is by far the largest group of texts in this set.

At this point, however, it should be noted that both the results of the sensitivity measure and measure of accuracy for the analogous experiment for the Z_1 set in *Series 2* is not zero, and in the discussed diagrams there is a set of dark gray bars unlike their brighter neighbors. It turns out that in the second iteration of the research, the *Text Events Categorizer* system managed to correctly classify a small part of the texts in the K_3 category. Such results again highlight the proposed solutions used in the *Series 2* experiments as having a positive impact not only on efficiency but also on the overall quality of the classification.

Comparison of Accuracy and Sensitivity in Both Series of Experiments

However, to confirm the beneficial effect of using the *dynamic extension of stop list* and *weighted keywords* on the quality of the classification, it is necessary to compare the values achieved by the accuracy and sensitivity measures for each pair of experiments, conducted for all three examined sets. The necessary data can be found in Tables 5 and 6 and in Figures 8 and 9. After carefully analyzing this data, it should be noted that

- The accuracy metric values are greater for the *Series 2* experiments in most cases. The differences range from almost 10% for the *Mortgage* category in the Z_2 set in favor of the *Series 2* study to less than 2% for the *Credit Card* category and also *set* Z_2 in favor of the *Series 1* experiment. Only in the case of two categories was a higher measure of accuracy obtained for the *Series 1* experiments—in addition to the already-mentioned *Credit Card* category for the Z_2 set, this was also the case for the *money supply* for the Z_3 set.
- The values of the sensitivity measure are higher for the studies from *Series 1* in all cases, except for the K_2 category for the Z_1 set. The record difference in favor of the experiment from the second iteration was obtained for the *Credit Card* label from the Z_2 set—less than 23%.

Taking into account the discussed results and summarizing the discussion of the presented research findings, a final conclusion can be drawn, confirming the noticeable and positive impact of using the proposed innovations in the feature-extraction process on the quality of the classification for all the selected datasets. However, it is important to note that, based on the relatively small number of detailed studies, it would be premature to conclude that the *dynamic extension of stop list* or *weighted keywords* will significantly improve the ranking for all the existing datasets. Nevertheless, the utilization of the novelties proposed in this paper is certainly worth considering in the context of designing similar systems or processes for text classification.

6. Conclusions

This paper presented the classification of texts as an issue that can solve many problems related to the use of very large sets of text data collected by enterprises from various industries. To achieve this purpose, we made an attempt to implement a universal system for the classification of text events, which for properly structured datasets from various industries will be characterized by a satisfactory level of efficiency and quality of classifica-

tion. The obtained classification quality results are definitely satisfactory. The proposed system implementing the designed text-classification process, taking into account two fully original solutions, is an effective attempt to at least partially solve the problem posed in Section 2.

The results of this research demonstrated that the quality of the classification for text events from selected industries, as offered by the *Text Events Categorizer* system, is indeed satisfactory. However, the most important conclusion drawn from the analysis of the obtained results is the positive impact of using the proposed novel mechanisms on the efficiency and quality of the classification. The demonstrated innovative solutions, namely the *dynamic extension of stop list* and *weighted keywords*, effectively mitigate the disadvantages of using a fixed stop list and classical keyword extraction approach, such as the TF method.

The future of text classification looks bright, but at the same time, it is very difficult to predict. It should be mentioned here that in addition to the *K-Nearest Neighbors* algorithm used and described in this paper, there are other methods of text classification based on, e.g., a *Naive Bayesian classifier*, *decision trees* or *neural networks*. Which of the listed solutions will become the dominant and most effective method for text classification, or maybe an even more interesting and effective classification method will emerge? The question posed in this way must wait for an answer, although it will not be easy.

Author Contributions: Conceptualization, M.W. and A.P.-M.; methodology, M.W. and A.P.-M.; validation, A.P.-M. and K.S.; formal analysis, M.W., A.P.-M. and K.S.; investigation, M.W.; resources, M.W.; data curation, M.W.; writing—original draft preparation, M.W., A.P.-M. and K.S.; writing—review and editing, A.P.-M. and K.S.; visualization, A.P.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Share of Customers in the United States Who Have Contacted Customer Service for Any Reason in the Past Month from 2015 to 2018. Available online: https://www.statista.com/statistics/815526/customers-who-have-contacted-customer-service-in-the-past-month-us/ (accessed on 7 March 2023).
2. 2017 State of Global Customer Service Report. Available online: http://info.microsoft.com/rs/157-GQE-382/images/EN-CNTNT-Report-DynService-2017-global-state-customer-service-en-au.pdf (accessed on 7 April 2023).
3. Freshdesk. Available online: https://freshdesk.com/pl/ (accessed on 8 March 2023).
4. Kowsari, K.; Jafari Meimandi, K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text classification algorithms: A survey. *Information* **2019**, *10*, 150. [CrossRef]
5. Hassan, S.U.; Ahamed, J.; Ahmad, K. Analytics of machine learning-based algorithms for text classification. *Sustain. Oper. Comput.* **2022**, *3*, 238–248.
6. Kadhim, A.I. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **2019**, *52*, 273–292. [CrossRef]
7. Shah, K.; Patel, H.; Sanghvi, D.; Shah, M. A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augment. Hum. Res.* **2020**, *5*, 1–16. [CrossRef]
8. Taunk, K.; De, S.; Verma, S.; Swetapadma, A. A Brief Review of Nearest Neighbor Algorithm for Learning and Classification. In Proceedings of the 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 15–17 May 2019; pp. 1255–1260. [CrossRef]
9. Deng, X.; Li, Y.; Weng, J.; Zhang, J. Feature selection for text classification: A review. *Multimed. Tools Appl.* **2019**, *78*, 3797–3816. [CrossRef]
10. Dzisevic, R.; Sesok, D. Text Classification using Different Feature Extraction Approaches. In Proceedings of the Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, 25 April 2019; pp. 1–4. [CrossRef]
11. Kadhim, A.I. Term Weighting for Feature Extraction on Twitter: A Comparison Between BM25 and TF-IDF. In Proceedings of the International Conference on Advanced Science and Engineering (ICOASE), Duhok, Iraq, 2–4 April 2019; pp. 124–128. [CrossRef]

12. Liu, Q.; Wang, J.; Zhang, D.; Yang, Y.; Wang, N. Text Features Extraction based on TF-IDF Associating Semantic. In Proceedings of the IEEE 4th International Conference on Computer and Communications (ICCC), Chengdu, China, 7–10 December 2018; pp. 2338–2343. [CrossRef]
13. Guo, A.; Yang, T. Research and improvement of feature words weight based on TFIDF algorithm. In Proceedings of the IEEE Information Technology, Networking, Electronic and Automation Control Conference, Chongqing, China, 20–22 May 2016; pp. 415–419. [CrossRef]
14. English Stop Words. Available online: https://countwordsfree.com/stopwords (accessed on 14 May 2023).
15. Wajeed, M.A.; Adilakshmi, T. Fuzziness in Text Classification Using Different Similarity Metrics. In *Advances in Computer Science and Information Technology. Computer Science and Information Technology. CCSIT 2012*; Meghanathan, N., Chaki, N., Nagamalai, D., Eds.; Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering; Springer: Berlin/Heidelberg, Germany, 2012; Volume 86, pp. 249–260.
16. Joulin, A.; Grave, E.; Bojanowski, P.; Mikolov, T. Bag of tricks for efficient text classification. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain, 3–7 April 2017; Volume 2, pp. 427–431.
17. Kondrak, G. N-gram similarity and distance. In Proceedings of the 12th International Symposium on String Processing and Information Retrieval, Buenos Aires, Argentina, 2–4 November 2005; pp. 115–126.
18. Seidl, T. Nearest Neighbor Classification. In *Encyclopedia of Database Systems*; Liu, L., Ozsu, M.T., Eds.; Springer: Boston, MA, USA, 2009.
19. Zhang, S.; Li, X.; Zong, M.; Zhu, X.; Cheng, D. Learning k for KNN Classification. *ACM Trans. Intell. Syst. Technol.* **2017**, *8*, 43. [CrossRef]
20. Batal, I.; Hauskrecht, M. Boosting KNN Text Classification Accuracy by Using Supervised Term Weighting Schemes. In Proceedings of the CIKM '09: 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, 2–6 November 2009; pp. 2041–2044.
21. Jiang, S.; Pang, G.; Wu, M.; Kuang, L. An improved K-nearest-neighbor algorithm for text categorization. *Expert Syst. Appl.* **2012**, *39*, 1503–1509. [CrossRef]
22. Band, A. How to Find the Optimal Value of K in KNN? Towards Data Scienc. Available online: https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb (accessed on 29 May 2023).
23. Consumer Complaint Database. Available online: https://www.kaggle.com/selener/consumer-complaint-database (accessed on 27 March 2023).
24. Reuters-21578 Text Categorization Collection Data Set. Available online: https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection (accessed on 27 May 2023).