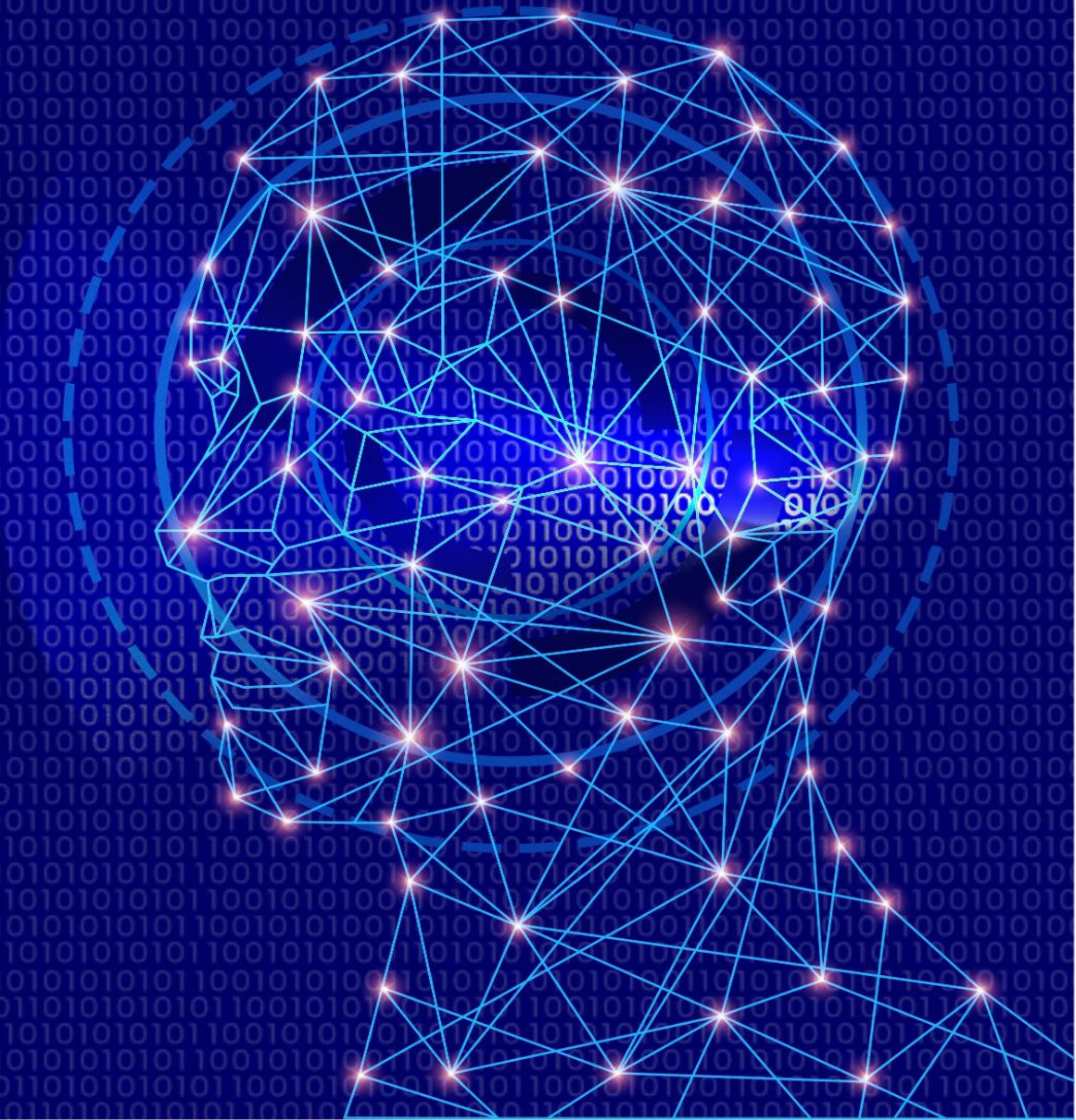
International Journal of

ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH

AAIR

Volume 3, Issue 1, 2023
ISSN: 2757-7422



International Journal of ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH

AAIR

Advances in Artificial Intelligence Research

https://dergipark.org.tr/aair

Owner

Osman ÖZKARACA - Muğla Sıtkı Koçman University

Editor-in-Chief

Ali KEÇEBAŞ - Muğla Sıtkı Koçman University

Osman ÖZKARACA - Muğla Sıtkı Koçman University

Editors

Date of Publication	February 2023
Language	English
Frequency	Published twice in a year
Graphic designer	Özden Işıktaş
Correspondence Address	Muğla Sıtkı Koçman University, Faculty of Technology, Information Systems Engineering, 48000 Kötekli/MUĞLA
Phone	0 506 4887818 – 0252 211 5526
Correspondence Mail	osmanozkaraca@mu.edu.tr

Volume : 3 2023 Issue : 1

International Journal of ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH
AAIR

Table of Contents

Pages | Research Articles

Volume : 3 2023 Issue : 1

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 1), pp. 1-12, 2023
doi: 10.54569/aair.1142568
Published online: Feb 15, 2023

Semantic Similarity Comparison Between Production Line Failures for Predictive Maintenance

Hilal Tekgöz [1,*], Sevinç İlhan Omurca[1], Kadir Yunus Koç [2], Umut Topçu [3], Osman Çelik [2]

[1] Kocaeli University, Department of Computer Engineering;
[2] Ibss Consulting, Department of Research and Development;
[3] Vestel Beyaz Eşya San. ve Tic. A.Ş., Department of Research and Development

Abstract

With the introduction of Industry 4.0 into our lives and the creation of smart factories, predictive maintenance has become even more important. Predictive maintenance systems are often used in the manufacturing industry. On the other hand, text analysis and Natural Language Processing (NLP) techniques are gaining a lot of attention by both research and industry due to their ability to combine natural languages and industrial solutions. There is a great increase in the number of studies on NLP in the literature. Even though there are studies in the field of NLP in predictive maintenance systems, no studies were found on Turkish NLP for predictive maintenance. This study focuses on the similarity analysis of failure texts that can be used in the predictive maintenance system we developed for VESTEL, one of the leading consumer electronics manufacturers in Turkey. In the manufacturing industry, operators record descriptions of failure that occur on production lines as short texts. However, these descriptions are not often used in predictive maintenance work. In this study, semantic text similarities between fault definitions in the production line were compared using traditional word representations, modern word representations and Transformer models. Levenshtein, Jaccard, Pearson, and Cosine scales were used as similarity measures and the effectiveness of these measures were compared. Experimental data including failure texts were obtained from a consumer electronics manufacturer in Turkey. When the experimental results are examined, it is seen that the Jaccard similarity metric is not successful in grouping semantic similarities according to the other three similarity measures. In addition, Multilingual Universal Sentence Encoder (MUSE), Language-agnostic BERT Sentence Embedding (LAbSE), Bag of Words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF) outperform FastText and Language-Agnostic Sentence Representations (LASER) models in semantic discovery of error identification in embedding methods. Briefly to conclude, Pearson and Cosine are more effective at finding similar failure texts; MUSE, LAbSE, BoW and TF-IDF methods are more successful at representing the failure text.

Keywords: *Predictive maintenance; natural language processing; sentence similarity; word representation methods.*

1. Introduction

With the huge increase in text data generated over time, Natural Language Processing (NLP) has attracted strong attention in the field of Artificial Intelligence (AI). Today, measuring similarity between sentences, paragraphs or documents accurately enables us to develop successful implementations in a wide range of NLP tasks such as information retrieval, keyword extraction, text summarization, text clustering, question answering or predictive maintenance (PdM). In the early literature, text representation was mostly based on Bag of Words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF) techniques. As a result of these representations, two texts were considered similar as long as they contained the same terms [1]. However even though BoW and TF-IDF provide how often the words appear in a given document and how distinctive they are, they are unable to capture the word meaning. To overcome these drawbacks of classical document representations, representations based on modern word embeddings that can capture semantic features and linguistic relationships between words using deep neural networks have gained importance [2]. Document representations based on classical approaches are sparse and ignore the semantic and syntactic similarity between words [3]. In contrast, word embedding methods such as Word2Vec [4], Glove [5], FastText [6] map variable length text to dense vectors and overcome the problem of lack of syntactic and semantic information in representations. However, all these models ignore the context of the words so they are defined as context unaware representations.

*Corresponding author
E-mail address: hilaltekgoz@outlook.com.tr

Received: 19/Jul/2022; Received in revised form: 20/Sep/2022; Accepted: 02/Nov/2022.

Against this, Bidirectional Encoder Representations from Transformers (BERT) [7] provides an effective way to contextualized representation of documents due to the fact that it generates vector representations for a word depending on its context. The similarity of the document vectors can be calculated using a distance measure such as Cosine similarity which has been widely used among NLP researchers to date [8].

Although many text mining jobs and NLP studies are gaining importance in industrial applications, there are few NLP studies in the field of PdM. Most of the systems in PdM aim to predict the potential failures before they occured. And several machine learning and deep learning methods are used for these kinds of solutions in the related literature. Essentially, in PdM systems, the explanations of the failures occurring in the production lines are recorded as text information by the operators. Analyzing the failure texts can provide a deep insight for PdM. Further, they can be considered as a valuable source of getting information about human feedback. Consequently, failure descriptions give more opportunity to explore valuable insights such as estimating unplanned downtimes which cannot be achieved from quantitative data. As known, an unplanned downtime occurs when there is an unexpected shutdown or failure of equipment or process. An unplanned downtime causes delays in regular production or maintenance schedules which can be extremely costly and inconvenient for a business operation.

In our study, we have the basic motivation to estimate the possible downtimes in minutes through the explanation text of a failure that occurs in the washing machine production line of VESTEL which is a manufacturing company for consumer electronics in Turkey. For this purpose, it is aimed to calculate the similarities between the explanation text of a newly occurring failure and the explanation texts of the existing past failures on the production lines.

Accurate measurement of similarities between texts strongly relies on more accurate document representations. Thus, in this study, we explored various models such as BoW, TF-IDF, MUSE, LAbSE, FastText and LASER for document representation. On the other hand, to obtain a better document similarity scheme, the efficiency of Pearson Similarity Metric, Cosine Similarity Metric, Jaccard Similarity Metric, and Levenshtein Similarity Metric were also explored. As a result of experiments with the Turkish production failure texts obtained from VESTEL, it was observed that the similarity metrics cosine and pearson gave more successful results, while MUSE, LAbSE models were more effective in document representations.

To the best of our knowledge, our study is the first in the literature to use Turkish NLP solutions in predictive maintenance.

The organization of this article is as follows: Section 2 will review previous studies in related literature. Section 3 will describe the techniques used. Section 4 will cover the applied models and dataset. Section 5 will discuss the conclusion and the future work.

2. Related Studies

Any unplanned interruption of industry to devices or machinery equipment within the organization can disrupt the core operation of the organization. An unplanned downtime not only causes costly delays and urgent repairs, but can also lead to poor customer experience and satisfaction in the long run. In the literature, PdM systems and posture prediction studies are encountered.

In Akhbardeh et al.'s study, it was aimed to cluster similar fault records using the MaintNet dataset produced for PdM and the DBSCAN clustering method. In the study, Pearson used cosine similarity metrics to examine similarity between clusters [9]. In another study on PDM, the multivariate time series classification method was used. The dataset used here was obtained from 23 sensors. They created an architecture called Conv-MHSA. It was emphasized that this architecture is much more computational efficiency than RNN methods [10]. Wellsandt et al. have announced that digital smart assistant systems for PdM will play an active role with developments in NLP [11].

Semantic similarity analysis is one of the key features of the NLP approaches [11]. Semantic similarity is usually calculated with datasets that include word pairs and a human-assigned similarity score. The versatility of natural language makes it difficult to define rule-based methods for determining semantic similarity measures [12]. Various word representation methods are used when calculating semantic similarities. For example, the Word2Vec method has recently gained attention due to new developments in the use of neural networks to learn low-dimensional, dense vector representation models known as word embedding [13]. Word representation methods provide vector representations for words where the relationship between two vectors reflects the linguistic relationship between two words and aims to capture semantic and syntactic similarities between words [13]. Traditional methods in semantic similarity analysis use the number of words combined to measure the similarity of sentences. However, traditional methods do not include deep similarities in semantic similarity evaluation and problems are encountered. The application of deep learning models in natural language processing tasks greatly improves the accuracy of sentence recognition tasks [14]. Pre-trained high-dimensional word vectors can also learn the context of the sentence. However, it is observed that the complexity of the algorithm increases as the vector gets larger [14]. In their study, Saipech and Seresangtakul converted the answers students entered into the system into vectors using TF-IDF. They used cosine similarity to measure the similarity of the answers. When the results obtained are compared with the scores prepared by expert teachers, it is seen that they give similar results [15]. Ruchindramalee et al. used Word2Vec and cosine similarity techniques to find similarities between social media and news sources. In the study, 70% accuracy was obtained in news sharing on social media compared to reliable online news sources [16]. In a study by Xiaolin et al, Word2Vec and Pearson developed a Chinese word-based approach using the similarity technique. As a result of the study, they obtained a high Pearson correlation coefficient [17]. Pre-trained transformer networks can be used to discover semantic text similarities. BERT, which stands for two-way encoder representations, is used in a variety of tasks such as question answering, semantic similarity, and language extraction. A new BERT model can be created by fine-tuning BERT models with just one additional output layer.

3. Materials and Methods

3.1. Similarity Metrics

The Pearson correlation coefficient [18] is a measure of the linear dependence between two random variables (real-valued vectors). Historically, it is the first official measure of correlation. The Pearson correlation coefficient of two variables, x and y, is formally defined as the covariance obtained by dividing the product of the standard deviations of the two variables and is shown in Equation 1.

$$r_{xy} = \frac{\sum(x-\underline{x})\sum(y-\underline{y})}{\sqrt{(x-\underline{x})^2}\sqrt{(y-\underline{y})^2}} \tag{1}$$

If the r_{xy} value defined in Equation 1 is 0, then x and y are said to be uncorrelated. However, the closer the r_{xy} value is to 1, the more it is said that there is a relationship between x and y [19]. According to the general guideline suggested by Evans (1996), the correlation results are classified as in the following [20]:

• very strong: 0.80 – 1.00

• strong: 0.60 – 0.79

• medium: 0.40 – 0.59

• weak: 0.20 – 0.39

• very weak: 0.00 – 0.19

Cosine similarity is created by measuring the angle between two vectors. If the cosine value is 0, it reports that the text snippets are not similar. A high cosine value indicates that the pieces of text are quite similar to each other. This gives a measurement of orientation, not magnitude; can be viewed as a comparison between documents in a normalized space [21]. Equation 1 shows the mathematical formula of cosine similarity.

$$cos\ (\theta) = \frac{A.B}{\|A\|\,\|B\|} \tag{2}$$

Where, parameters A and B represent different document vectors. It is also known as the Jaccard similarity coefficient [22], which is a statistical method used to find the similarity and diversity between two finite sets. It is defined as in Equation 3. In the equation, A and B denote the vectors of failure texts.

$$J(A,B) = \frac{\|A \cap B\|}{\|A \cup B\|} \tag{3}$$

The Levenshtein similarity is a simple edit distance. The distance between two strings is defined as the minimal number of basic operations (insert, delete, or replace) needed to convert one string to another. The ratio between the distance and length of longer strings is considered as the similarity between these strings.

3.2. Document Representation Methods

In traditional document representation methods, a document is considered as a bag of words. BoW was first proposed in text retrieval problems to create the vector representations of text documents [24]. The basic assumption is that the words selected from the documents are expressed using frequency sets in an unordered manner. Thus, categorical data is converted into numerical form [25].

TF-IDF is a commonly used method to convert words to vectors when analyzing documents. The TF-IDF determines the relative frequency of words in a selected document in inverse proportion over the entire document. Here each word is treated as a feature. In determining the value, it uses the frequency of the term in the TF document and the reverse document frequency of the term IDF [26]. In addition, using less features on texts with TF-IDF, higher performance is achieved compared to the word bag method [27].

The Word2Vec method is one of the word representation methods used to represent documents. It understands and vectorizes the meanings of words in a document based on the close distances of words with similar meanings in a given context [28]. There are two main learning algorithms. These are continuous bag-of-words and continuous skip-gram algorithms. With the continuous word bag method, it predicts the current word according to the context. It predicts the surrounding words based on the current word with Skip-gram. Unlike the standard word bag model, continuous wordbags use a distributed representation of context [29].

FastText is an open-source modern word representation method developed by the Facebook research team [6]. FastText is an effective word representation method for learning word representations and text classification. Its purpose is not to learn word representations, but to consider the internal structure of the word. It has been observed that this method gives more effective results in morphologically rich languages. FastText works by sliding a window above the introductory sentence or by using the continuous word bag method. It can be viewed as a series of updates in a neural network containing two layers of weights and three layers of neurons [30]. It is based on the skip-gram sub-method of Word2Vec method. Unlike Word2Vec, words are handled by dividing them into n-grams. Words divided into n-grams are represented vectorally as the sum of n-grams. Other methods that do not deal with n-grams ignore the morphology of the word because they convert each word into a vector. There is a limitation especially for languages with large vocabulary and very rare words. The FastText method overcomes this limitation thanks to the n-gram method.

3.3. Transformers

Transformers have been developed as a solution to the forgetting problem of long inputs. The Encoder-Decoder model developed to avoid long input problems was insufficient. It was created to reduce both long input problems and training time. Transformer is a 6-layer encoder-decoder model that creates a target sequence from the source sequence via the decoder [31]. The encoder and decoder consist of self-attention and a feed-forward layer. In the decoder, it provides a match to the markers with the encoder with an attention layer in between. It is a word representation method based on a masked language model and pre-trained using bidirectional transformers. In word vector generation methods such as Word2vec, GloVe and other neural network models, word vectors are mostly context independent. And to represent the ambiguous nature of words, it becomes harder to learn more relevant information. BERT models can successfully learn semantic features and create vectors according to the semantic differences of words.

In Transformer models, within the scope of this study, LaBSE, MUSE, LASER models were used to define Turkish sentence embeddings. It is possible to retrain models with fine tuning. To briefly explain these methods, LaBSE is a BERT based model trained on 17 billion monolingual sentences and 6 billion bilingual sentence pairs. MUSE is a sentence encoding model simultaneously trained on multiple tasks and 16 languages, including Turkish. LASER is a language model based on the BiLSTM encoder trained on 93 languages, including Turkish.

4. Experiments

4.1. Dataset Description and Preprocessing Steps

In this study, a real dataset obtained from a manufacturing company about consumer electronics in Turkey is used to explore the semantic similarities between the mechanical and electrical failure explanations in the production lines. There are 146 failure records belonging to a special production line in the dataset. The average document length of the failure texts is 30. The attributes of the dataset and some data samples are reflected in **Table 1**. In the failure description column, the abbreviation 'MK' means that the failure in the production line is a mechanical failure. Numerical values in the failure description column, are the identifying numbers of the equipment used in the production lines.

The failure texts in the dataset were constituted with the failure descriptions belonging to a specific Production line entered by the operators when a downtime in the production had occurred. While creating these records it has been considered that the production failures can only be mechanical or electrical.

Preprocessing is a very crucial task in terms of managing and analyzing the text data before it is fed into the model as an input. In our experiments, no stem reduction was performed on the words in the sentence collection. Only letters in the capital were converted to lowercase letters and numerical/special characters were removed. Additionally, first we only evaluated failure descriptions excluding station information. We then extended the experiments by adding textual station information to the failure descriptions.

Table 1. *Data Samples of The "Mechanical-Electrical" Failures Occurring on a Specific ProductionLine*

Shift Information	Start of Downtime	End of Downtime	Downtime in Minutes	Station Information	Failure Description
08:00-20:00	08:00:00	08:25:00	25	3.ist.2.pres	1,2,Tank Minimum Yağ
08:00-20:00	09:50:00	10:10:00	20	4.ist.3.pres	Sıcaklık Alarmı Veriyor . 10001.MK
08:00-20:00	16:25:00	16:35:00	10	2.ist.1.pres.	Sıcaklık Alarmı Veriyor
24:00-08:00	23:00:00	01:00:00	120	15.ist.tds tox	1503 Sök Tak Yapmada
08:00-16:00	10:50:00	11:05:00	15	9.ist.boy bük.kal.	HSV901 Yukarı Pozisyonuna
08:00-16:00	09:50:00	10:05:00	15	15.ist AmorAyağıBük	15.İST 15093 Nolu Sensör
08:00-16:00	11:15:00	12:15:00	60	1.ist.kesik sac bes.	SM101 Kaprin Kırık Hareket

4.2. Algorithm and Results

Recently, semantic discovery of similarities between texts has come to the fore as an important field of natural language processing for meaningful information search and matching. In this study, BoW, FastText, TF-IDF, Word2Vec methods were used to represent the failure descriptions. Other than these, Language-agnostic BERT Sentence Encoder, Multilingual Universal Sentence Encoder, LanguageAgnostic SEntence Representations models were used to represent sentence embeddings. In order to extract semantic similarities between failure text descriptions more accurately, the effectiveness of Cosine, Jaccard, Levenshtein, Pearson similarity scales were investigated. In **Figure 1**, the semantic similarity steps are shown in the diagram. This work was done with a machine with NVIDIA GTX 1660 Ti GPU and 16 GB Memory.

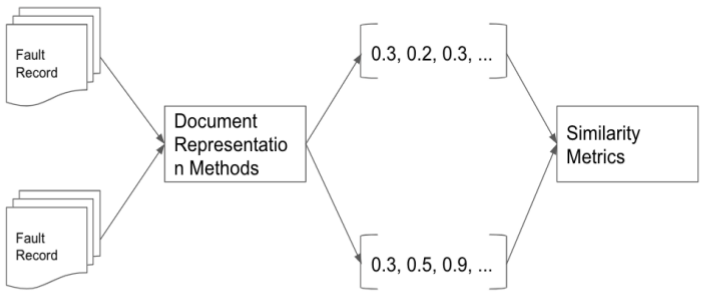


Figure 1. Steps to extract semantic similarities between failures

To investigate the effectiveness of the methods an example failure description text "Sıcaklık alarmı veriyor mk" ("Gives temperature alarm mk") has been considered as a query text. The similar failure descriptions can be extracted according to a user defined threshold value. In our experiments, the threshold value is determined as 70 to define whether the documents are semantically similar. The experimental results in **Tables 2, 3, 4, 5,** respectively, were obtained by BoW, TF-IDF, Word2Vec, FastText methods. The ID column in the tables indicates a unique identifier assigned to failure records. The Downtime in Minutes column shows how long the fault records continue.

Table 2. *Results obtained by BoW and subtracting numerical values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	71,43	25
	97	minimum yağ alarmi veriyor mk	71,43	15
	132	sicaklik alarmi veriyor	96,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	75,00	10
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	75,00	10

Looking at the results in **Table 2**, similar failure descriptions obtained according to Pearson and Cosine scales are ‘sicaklik alarmi veriyor mk’ (gives temperature alarm-mechanical failure) and ‘sicaklik alarmi veriyor’(gives temperature alarm) as the best match results. However, looking at the downtime minutes, it is seen that there is a difference of 10 minutes between them. This shows that if the failure in the production line is caused by the temperature alarm, it will be in the range of 10-20 minutes and the downtimes may differ from each other by 10 minutes. According to Jaccard, no similar failure record was found. According to Levenshtein similarity, ‘sicaklik alarmi veriyor’ (gives temperature alarm) and ‘minimum yağ alarmi veriyor’, (gives minimum oil alarm) which are semantically different, were found to be similar records.

Table 3. *Results obtained by TF-IDF and subtracting numerical values*

Similarity Metric	**Id**	**Failure Description**	**Similarity Score**	**Downtime in Minutes**
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	71,4	25
	97	minimum yağ alarmi veriyor mk	71,43	15
	132	sicaklik alarmi veriyor	96,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	84,4	10
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	84,0	10

When the results in **Table 3** are examined, it is observed that the same failure descriptions are captured with a similarity rate of 0.84 in the Pearson and Cosine similarity scales. When the results reflected in **Table 2** and **Table 3** are examined together, it is observed that the same document similarities are captured with the TF-IDF and BoW representations.

Table 4. *Results obtained obtained by Word2Vec and subtracting numerical values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	71,43	25
	97	minimum yağ alarmi veriyor mk	71,43	15
	132	sicaklik alarmi veriyor	96,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
Pearson	3	sicaklik alarmi veriyor mk	BASE	20

Looking at the results of **Table 4**, according to Pearson, Cosine and Jaccard similarity scales, no record matching the failure description of ‘sicaklik alarmi veriyor mk’ (gives temperature alarm-mechanical failure) was found. It was also found that the failure description ‘sicaklik alarmi veriyor’ matched with a similarity ratio of 0.96 on the Levenshtein scale. It is understood from the similarity results of Jaccard, Cosine and Pearson that the abbreviations in the fault descriptions (as in the ‘mk-mechanical failure’ abbreviation here) are not very effective in the similarity calculation when the documents are represented in word2vec.

Table 5. *Results obtained by FastText and subtracting numerical values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	71,4	25
	97	minimum yağ alarmi veriyor mk	71,43	15
	132	sicaklik alarmi veriyor	96,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20

	28	minimum yağ alarmi veriyor	76,41	20
	30	minimum yağ alarmi veriyor mk	77,43	25
	39	tank minimum yağ alarmi veriyor	72,96	25
	43	pompa minimum yağ alarmi veriyor	73,51	30
	56	tank minimum yağ alarmi veriyor mk	72,03	20
	58	tank minimum yağ alarmi veriyor mk	72,96	15
	59	minimum yağ alarmi veriyor mk	72,96	15
	97	minimum yağ alarmi veriyor mk	77,43	15
	122	minimum yağ alarmi veriyor pompa açilmiyor	75,3	25
	124	pres minimum yağ alarmi veriyor pompa açilmiyor	72,33	20
	132	sicaklik alarmi veriyor	99,82	10
	146	tank minimum yağ alarmi veriyor mk	72,96	25
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	28	minimum yağ alarmi veriyor	76,47	20
	30	minimum yağ alarmi veriyor mk	77,50	25
	39	tank minimum yağ alarmi veriyor	73,01	25
	43	pompa minimum yağ alarmi veriyor	73,00	30
	56	tank minimum yağ alarmi veriyor mk	72,55	20
	58	tank minimum yağ alarmi veriyor mk	73,96	15
	59	minimum yağ alarmi veriyor mk	72,96	15
	97	minimum yağ alarmi veriyor mk	77,43	15
	122	minimum yağ alarmi veriyor pompa açilmiyor	75,3	25
	124	pres minimum yağ alarmi veriyor pompa açilmiyor	72,33	20
	132	sicaklik alarmi veriyor	84,0	10

According to the results in **Table 5**, more failure descriptions were discovered as similar descriptions by using FastText. However, it was observed that the temperature alarm and the oil alarm cannot be distinguished semantically.

In our experiments, we also investigate the efficiency of transformer models by considering the sentence "Sıcaklık alarmı veriyor mk" as a query sentence. The obtained results for LaBSe, MUSE and LASER are reflected in **Tables 6, 7** and **8**, respectively.

Looking at the results in **Table 6** with Jaccard and Cosine similarity metrics, the two sentences "Sıcaklık alarmı veriyor mk" (gives temperature alarm-mechanical failure) and "Sıcaklık alarmı veriyor" (gives temperature alarm) were discovered to be similar. The LaBSe model succeeded in capturing the sentences that are semantically the most similar.

Table 6. *Results obtained by LaBSe and subtracting numerical values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	73,77	25
	97	minimum yağ alarmi veriyor mk	73,77	15
	132	sicaklik alarmi veriyor	96,67	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	96,67	10
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	96,67	10

Table 7. *Results obtained by MUSE and subtracting numerical values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	92,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	90,88	10
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	91,00	10

In the results of **Table 7**, it is observed that the Pearson, Cosine, and Levenshtein scales give a higher similarity rate for the failures such as ‘sicaklik alarmi veriyor mk’ and ‘sicaklik alarmi veriyor’. The same results are obtained for LaBSe and MUSE models by using Pearson and Cosine metrics.

Table 8. *Results obtained by LASER and subtracting numeric values*

Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
Levenshtein	3	sicaklik alarmi veriyor mk	BASE	20
	132	sicaklik alarmi veriyor	92,00	10
Jaccard	3	sicaklik alarmi veriyor mk	BASE	20
Cosine	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	86,43	25
	58	tank minimum yağ alarmi veriyor mk	82,96	15
	59	minimum yağ alarmi veriyor mk	82,21	15
	97	minimum yağ alarmi veriyor mk	86,43	15
	132	sicaklik alarmi veriyor	86,88	10
	146	tank minimum yağ alarmi veriyor mk	86,96	25
Pearson	3	sicaklik alarmi veriyor mk	BASE	20
	30	minimum yağ alarmi veriyor mk	86,00	25
	58	tank minimum yağ alarmi veriyor mk	82,46	15
	59	minimum yağ alarmi veriyor mk	82,11	15
	97	minimum yağ alarmi veriyor mk	86,83	15
	146	sicaklik alarmi veriyor	82,00	25

When the results in **Table 8** are examined, it is realized that the LASER model was not successful in capturing semantic similarities compared to the LaBSe and MUSE models. However, considering the Levenshtein scale, similar failure descriptions seem to be matched. As a result of the LASER model, it was concluded that the downtimes of the faults discovered similarly could be more distant from each other.

The following experimental results were obtained by combining station information and failure descriptions. In the remaining experiments, only Pearson and Cosine similarities, which gave more significant results in previous experiments, were taken into consideration as similarity metrics. Similar to the results obtained above, it was observed that the BoW and TF-IDF models were more successful than the Word2Vec model. As in **Table 5**, the new results obtained with the FastText model are not reflected in **Table 9**, as they fail to capture the semantic relationships between sentences.

Table 9. *BoW, TF-IDF and Word2Vec results obtained by including station information*

Model	Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
BoW	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	79,86	10
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	79.54	10
TF-IDF	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	79.86	10
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	79.54	10
Word2Vec	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20

In **Table 10**, results obtained with LaBSe, MUSE and LASER are summarized. As in the experiments reflected in **Table 9**; failure descriptions were also combined with station information to achieve these results.

Table 10. *LaBSe, MUSE and LASER results obtained by including station information*

Model	Similarity Metric	Id	Failure Description	Similarity Score	Downtime in Minutes
LaBSe	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	75.33	10
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	75.00	10
MUSE	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	75.33	10
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		132	sıcaklık alarmi veriyor 2.ist.1.pres	75.00	10
LASER	Cosine	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		30	minimum yağ alarmi veriyor mk	86.43	25
		58	tank minimum yağ alarmi veriyor mk	82.96	15
		59	minimum yağ alarmi veriyor mk	82.21	15
		97	minimum yağ alarmi veriyor mk	86.43	15
		132	sicaklik alarmi veriyor	86.88	10
		146	tank minimum yağ alarmi veriyor mk	86.96	25
	Pearson	3	sıcaklık alarmi veriyor 10001 mk4.ist3.pres	BASE	20
		30	minimum yağ alarmi veriyor mk	86.00	25
		58	tank minimum yağ alarmi veriyor mk	82.46	15
		59	minimum yağ alarmi veriyor mk	82.11	15
		97	minimum yağ alarmi veriyor mk	86.83	15
		146	tank minimum yağ alarmi veriyor mk	82.00	25

When the **Table 10** is examined, it has been observed that, LaBSe and MUSE models were more successful than the LASER model in finding the most semantically similar record with query sentence "sıcaklık alarmi veriyor 10001 mk 4.ist3.pres" among 146 failure records.

As a result of extensive experimental studies conducted for different query sentences, it has been observed that LaBSe and MUSE models are superior to LASER, BoW, TF-IDF, Word2vec and FastText models in discovering semantic similarities between failure explanations of production lines. In general, models and methods are good at finding semantic similarities, but sometimes similar fault records have different downtimes. Here, the fault downtime can be specified by taking the average duration of similar matching records.

5. Results and Future Studies

In this study, the semantic text similarities between the failure descriptions in the production lines have been explored for later use in PdM work. A real-life dataset obtained from a manufacturing company about consumer electronics in Turkey has been used for the experiments. The textual explanations of the failures in the production lines were created by the operators at the time of the production downtimes. In this study, while past failure explanations similar to a real time occurred failure explanation are discovered, it is aimed to estimate the downtime of the equipment based on text similarity analysis in future studies.

For this aim, BoW, TF-IDF, MUSE, LAbSE, FastText and LASER models were used to represent the failure descriptions. Besides, Pearson, Cosine, Jaccard, and Levenshtein Similarity Metrics were used to calculate text similarity between failure descriptions. When we analyze the experimental results, it has been observed that Pearson and Cosine are more effective at finding similar failure explanations. Additionally, it is concluded that, MUSE, LAbSE and traditional word representation methods such as BOW and TF-IDF outperform FastText and LASER models in the semantic discovery of failure descriptions. At the same time, different downtimes were found in similar fault records in the estimation of the downtime by taking advantage of the similarities of the fault records. Here, after the similarities are found, the downtime minutes should be determined by taking their averages.

Declaration of interest

It was presented as a summary at the ICAIAME 2022 conference.

Acknowledgements

This work has been supported by TUBİTAK in Turkey under project number 3215073.

References

[1] Chandrasekaran D, and Vijay M. "Evolution of semantic similarity—a survey." ACM Computing Surveys (CSUR) 54.2, 1-37, 2021.

[2] Wang Y, et al. "A comparison of word embeddings for the biomedical natural language processing." Journal of biomedical informatics 87,12-20, 2018.

[3] Liu J, Tianqi L, and Cong Y. “Newsembed: Modeling news through pre-trained document representations”, arXiv preprint arXiv:2106.00590, 2021.

[4] Mikolov T, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781, 2013.

[5] Pennington J, Richard S, and Christopher D.M. “Glove: Global vectors for word representation”. Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014.

[6] Bojanowski P, et al. “Enriching word vectors with subword information”, Transactions of the association for computational linguistics 5, 135-146, 2017.

[7] Devlin J, et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”, arXiv preprint arXiv:1810.04805, 2018.

[8] Mohammad S.M, and Graeme H. “Distributional measures of semantic distance: A survey”, arXiv preprint arXiv:1203.1858, 2012.

[9] Akhbardeh F, Travis D, and Marcos Z. “NLP tools for predictive maintenance records in MaintNet”. Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations. 2020.

[10] Yang H, Aidan L, and Travis D. “Predictive maintenance for general aviation using convolutional transformers”. Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 36. No. 11. 2022.

[11] Wellsandt S. et al. “Hybrid-augmented intelligence in predictive maintenance with digital intelligent assistants”. Annual Reviews in Control, 2022.

[12] Maulud D.H., et al. “State of art for semantic analysis of natural language processing”, Qubahan Academic Journal 1.2, 21-28, 2021.

[13] Netisopakul P, et al. “Improving the state-of-the-art in Thai semantic similarity using distributional semantics and ontological information”, Plos one 16.2, 2021.

[14] Zhang P, et al. “Semantic similarity computing model based on multi model fine-grained nonlinear fusion”, IEEE Access 9, 8433-8443, 2021.

[15] Saipech P, and Pusadee S. “Automatic Thai subjective examination using cosine similarity”. 2018 5th international

conference on advanced informatics: concept theory and applications (ICAICTA). IEEE, 2018.

[16] Chandrathlake R, et al. “A semantic similarity measure-based news posts validation on social media”. 2018 3rd International Conference on Information Technology Research (ICITR). IEEE, 2018.

[17] Jin X, Shuwu Z, and Jie L. “Word semantic similarity calculation based on word2vec”. 2018 International Conference on Control, Automation and Information Sciences (ICCAIS). IEEE, 2018. Pearson, K. Notes on Regression and Inheritance in the Case of Two Parents Proceedings of the Royal Society of London, 58, 240-242, 1895.

[18] Zhou H, et al. “A new sampling method in particle filter based on Pearson correlation coefficient”. Neurocomputing 216 , 208-215, 2016.

[19] Evans, J.D. “Straightforward statistics for the behavioral sciences”. Thomson Brooks/Cole Publishing Co, 1996.

[20] Jayakodi K., Bandara M., and Meedeniya D. “An automatic classifier for exam questions with WordNet and Cosine similarity”. 2016 Moratuwa engineering research conference (MERCon). IEEE, 2016.

[21] Jaccard P. “Nouvelles recherches sur la distribution florale”. Bull. Soc. Vaud. Sci. Nat., 44, 223-270, 1908.

[22] Levenshtein, V.I. “Binary codes capable of correcting deletions, insertions, and reversals”. Soviet physics doklady. Vol. 10. No. 8. 1966.

[23] Bosch A, Xavier M, and Robert M. “Which is the best way to organize/classify images by content?”. Image and vision computing 25.6, 778-791, 2007.

[24] Aksu, M Ç, and Karaman E. “FastText ve Kelime Çantası Kelime Temsil Yöntemlerinin Turistik Mekanlar İçin Yapılan Türkçe İncelemeler Kullanılarak Karşılaştırılması”. Avrupa Bilim ve Teknoloji Dergisi 20, 311-320, 2020.

[25] Trstenjak B, Sasa M, and Dzenana D. “KNN with TF-IDF based framework for text categorization”. Procedia Engineering 69, 1356-1364, 2014.

[26] Uçar, M.K, Bozkurt M.R, and Bilgin C. “Signal Processing and Communications Applications Conference”. IEEE, 2017.

[27] Jang, B.C, Inhwan K, and Jong W.K. “Word2vec convolutional neural networks for classification of news articles and tweets”. PloS one 14.8, 2019.

[28] Lilleberg J, Yun Z, and Yanqing Z. “Support vector machines and word2vec for text classification with semantic features”. 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC). IEEE, 2015.

[29] Santos I., Nadia N, and Luiza de Macedo M. “Sentiment analysis using convolutional neural network with fastText embeddings”. 2017 IEEE Latin American conference on computational intelligence (LA-CCI). IEEE, 2017.

[30] Tekgöz H, Çelenli H.İ, and Omurca S.İ. “Semantic Similarity Comparison of Word Representation Methods in the Field of Health”. 2021 6th International Conference on Computer Science and Engineering (UBMK). IEEE, 2021.

[31] Singh S, and Ausif M. “The NLP cookbook: modern recipes for transformer based deep learning architectures”. IEEE Access 9, 68675-68702, 2021.

[32] Feng F, et al. “Language-agnostic bert sentence embedding”. arXiv preprint arXiv:2007.01852, 2020.

[33] Yang Y, et al. "Multilingual universal sentence encoder for semantic retrieval." arXiv preprint arXiv:1907.04307, 2019.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 1), pp. 12-18, 2023
doi: 10.54569/aair.1216400
Published online: Jan 8, 2023

A Block-Building Based GRASP Method for Solving Container Loading Problem

Merve Özdemir[1*], Tuncay Yiğit [1]

[1] Süleyman Demirel University, Faculty of Engineering, Department of Computer Engineering, Isparta, Turkey

Abstract

The importance of container transportation is constantly increasing. For this reason, lower cost transportation is of great importance for companies in transportation by air, land, rail and sea in domestic and international markets. One way of reducing the costs is to utilize the container volume effectively. In this study, a block-building based GRASP method is proposed for solving the container loading problem. The proposed method achieved a loading factor over 91% on the BR dataset. The results are compared with other GRASP methods and other heuristic or meta-heuristic algorithms in the literature. The results show improvements in comparison to the other methods.

Keywords: *Container loading problem, optimization, heuristic, GRASP.*

1. Introduction

There is a significant increase in the need for logistics because of the growth in trade volumes locally and globally thanks to the rapid development in the field of transportation. Land, air, sea and rail transportations gain value and the investments in these areas are increasing. With the globalization of trade in the world, the volume of trade is also growing. The growing trade volume has brought competition to the fore in domestic and international businesses. Countries have started to invest more in the logistic sectors to perform better in the competitive environment.

The container loading problem (CLP) is one of the most important problems in terms of providing better and faster transportation. In today's competitive conditions in domestic and international markets, every organization aims to deliver its product at the lowest cost. One way of reducing the costs is to use the container volume at the maximum.

The CLP is a packaging problem where a large container needs to be filled with smaller boxes. As the number of box types and the number of boxes increase, the variety of settlements increases gradually. These types of problems are defined as NP-Hard problems which require exponential time to solve. Therefore, heuristics and meta-heuristics are important approaches in solving NP-Hard problems [1].

Heuristic algorithms imitate natural phenomena to accomplish any purpose. There is no guarantee that the best solution will be found. They aim to find solutions close to the best by deciding the effective ones among various alternatives. They usually reach almost the best solution efficiently. Heuristic algorithms are needed when the exact solutions of optimization problems have an undefined structure, and they can be used as a part of the process of finding the exact solution [2].

General purpose heuristic methods can be examined in six groups: Biological-based, physics-based, swarm-based, social-based, music-based and chemistry-based. The genetic algorithm, differential evolution algorithm, ant colony algorithm, cuckoo search algorithm, artificial neural networks, bee colony algorithm and artificial immune systems are biological-based methods. The imperialist competitive algorithm, parliamentary optimization algorithm and tabu search algorithm are social-based algorithms. The artificial chemical reaction algorithm is chemical-based; the harmony search algorithm is music-based; the simulated annealing algorithm, big bang big crunch algorithm, gravitational search algorithm, central force optimization, intelligent water drop algorithm and electromagnetism algorithm are physics-based heuristic methods. The particle swarm optimization is a swarm-based algorithm [3].

In this paper, a GRASP method is proposed to solve the container loading problem. The details of the proposed method are explained and other heuristic algorithms in the literature are compared with the proposed method and the results are presented in the following sections.

2. Literature Review

In literature, we can find several heuristic and meta-heuristic methods proposed to solve the container loading problem.

Parreno *et al.* proposed a GRASP method for solving the container loading problem. This method is based on a constructive block heuristic and tested on Bischoff and Ratcliff (BR) data set. Compared to parallel

*Corresponding author
E-mail address: merve.aydemir90@gmail.com

Received: 8/Dec/2022; Received in revised form: 12/Jan/2023; Accepted: 14/Jan/2023.

algorithms, it performs better on average but not for every problem of the dataset [4].

Moura and Oliveira presented a GRModGRASP algorithm based on the wall-building approach for solving container loading problem. Results were tested on BR and LN (Loh and Nee) datasets [5].

Gehring and Bortfeldt proposed a parallel genetic algorithm for the CLP with a single container. The parallel genetic algorithm follows a migration pattern. The quality of the parallel genetic algorithm has been demonstrated by the tests performed [6].

Dereli and Daş proposed two algorithms for the CLP using the ant colony optimization approach. The performances of these algorithms, whose parameters are determined by factorial design, have been tested for dataset given in the literature. The KKS-2 algorithm, one of the two proposed algorithms, gave better results than the KKS-1 algorithm [7].

Koyuncuoğlu examined customer priority and assignment rules in combination in his study. They proposed a mixed integer mathematical model for the simultaneous assignment of one or two forklifts to containers. The results of the model were analyzed using an approach based on genetic algorithm, tabu search, nearest neighbor and Lin-Kernighan heuristics [8].

Ceschia and Schaerf used local search methodologies as a solution technique in their study. Local search works on arrays of boxes to load, and the actual load is obtained by calling a special procedure the loader at each iteration. The loader places the boxes in the container using a deterministic heuristic that generates a suitable load according to the constraints [9].

Can and Sahingoz used simulated annealing meta-heuristics in their study. Their method was compared with Bortfeldt's test scenarios of the CLP [10].

Sheng *et al*. proposed a method using simulated annealing and tree-graph search procedure to solve the container loading problem. They examined the results by testing the obtained results with the BR datasets [11].

With the algorithm proposed by Zhou and Liu, it regards the loading arrangement as the position of individuals in the swarm, and as individuals interact with each other and with loading constraints, most of them will meet with the good ones and eventually stop at the best position. The algorithm can solve the three-dimensional container loading problem with or without pallets [12].

3. Material and Methods

3.1. Container loading problem

The container loading problem is solved in a three-dimensional space consisting of the X, Y, Z axis as shown in **Figure 1**. The X axis is parallel to the right and left directions, the Y axis is parallel to the back and front directions, and the Z axis is parallel to the up and down directions. The point where the three points intersect (0, 0, 0) is the origin point.

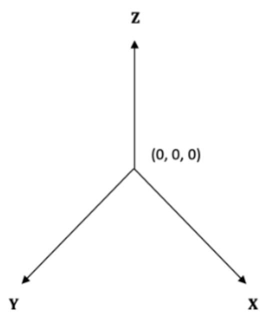


Figure 1. *X Y Z axis*

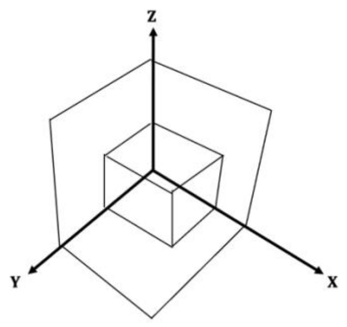


Figure 2. *Box placement in container*

As in **Figure 2**, there are *n* boxes of different sizes with width w_i, length l_i, and height h_i that are intended to be placed inside a container of width W, length L, and height H. Various constraints are defined for the container loading problem. They can be grouped into two categories: Mandatory and non-mandatory constraints [13]. We list the main constraints for the container loading problem as follows.

- All boxes must be completely inside the container.
- There must be no overlap between boxes.
- Each box must be placed parallel to the boundary surface of the container [14].

We can name the basic constraints as overflow, intersection and support constraints. In the overflow constraint, it is checked whether the dimensions of the box to be placed overflow from the container.

$$0 \leq xi \leq W - wi \tag{1}$$

$$0 \leq yi \leq L - li \tag{2}$$

$$0 \leq zi \leq H - hi \tag{3}$$

In Eqs. (1)-(3), xi is the width of the box to be placed; yi is the length of the box to be placed; zi is the height of the box to be placed. If inequality in Eqs. (1)-(3) are not satisfied, the box overflows the container. In the support constraint, it is checked whether the box to be placed is supported by the container floor or other boxes.

$$(yj \leq yi \leq yj + lj) \wedge (xj \leq xi \leq xj + wj) \tag{4}$$

$$(yj \leq yi \leq yj + lj) \wedge (xj \leq xi + wi \leq xj + wj) \tag{5}$$

$$(yj \leq yi + li \leq yj + lj) \wedge (xj \leq xi \leq xj + wj) \tag{6}$$

$$(yj \leq yi + li \leq yj + lj) \wedge (xj \leq xi + wi \leq xj + wj) \tag{7}$$

$$(yj \leq (yi + li)/2 \leq yj + lj) \wedge (xj \leq (xi + wi)/2 \leq xj + wj) \tag{8}$$

In Eqs. (4)-(8), xj is the width of the placed box; yi is the length of the placed box. If the box to be placed is on the container floor, the box is supported, but if the box to be placed is not on the container floor, the box is supported if at least three of the five inequalities in Eqs. (4)-(8) are met. In intersection constraint, it is checked whether the box to be placed intersects with the other boxes that are placed.

$$(xj - xi \geq wi) \vee (xi - xj \geq wj) \vee (yj - yi \geq li) \vee (yi - yj \geq lj) \vee (zj - zi \geq hi) \vee (zi - zj \geq hj) \tag{9}$$

It is checked whether the inequality given in Eq. (9) is satisfied between the box to be placed and all the boxes that are placed. If at least one of the inequalities is not satisfied, an intersection has occurred between the boxes.

3.2. GRASP algorithm

The GRASP is a multi-start or iterative process that was developed by Feo and Resende for solving hard combinatorial problems [15]. Each GRASP iteration contains a construction phase where a feasible solution is constructed, and a local search phase which works iteratively by replacing the current solution with a better solution found in the neighborhood of the current solution. Each solution constructed in the constructive phase is used in the local search phase as a starting solution. The best overall solution is returned as the result.

In the construction phase, a complete solution is iteratively constructed starting from an empty solution, by adding one element at a time. At each construction step, the greedy function measures the benefit of adding each element in a candidate list and they are sorted by their benefits. At each iteration of the construction step, these measurements are updated with the change brought about by selecting the previous item. Therefore, the heuristic is adaptive. The randomized component of a GRASP is obtained by random selection of one of the best candidates on the list. However, this is not always the best candidate. The list that contains only the best candidates. This list is named as the restricted candidate list (RCL) [16].

The solution generated in the GRASP construction phase cannot be guaranteed to be a locally optimal solution. Therefore, a local search is needed to try to improve each solution from the construction phase. A local search algorithm iteratively replaces the current solution with a better solution in the neighborhood. It stops when local search fails to find a better solution in the neighborhood. If there is no better solution in terms of the objective function values in the neighborhood, this solution is considered as the local optimum [17].

3.3. BB-GRASP method

In this paper, a block-building based GRASP method is proposed for solving the container loading problem. Six key elements are used as the block-building approach defined in [18]. The details of these six key elements are as follows:

(K1) **cover representation** is used to represent empty space
(K2) **general blocks** are used to construct solutions

(K3) free space is selected with **smallest Manhattan distance**
(K4) the evaluation function $VCS_{\alpha}(b,r)$ is used to select blocks
(K5) blocks are placed at the **anchor corner**
(K6) as a search strategy, the proposed **BB-GRASP method** is used

K1, represents the empty space inside the container. When any box is placed at the corner of the container, the residual empty space is polyhedron. Three overlapping cuboids are used to represent the residual space. Each cuboid is made of a large rectangular box that is internally discrete from the placed block, and this is called cover representation.

K2 includes the construction of blocks. The general block consists of boxes of different types. It can also contain boxes that are located in different directions.

In K3, Manhattan distance is employed to select the next residual space to be placed. The corner with the smallest distance is corresponding to the anchor corner of cuboid, and the residual space with the smallest Manhattan distance is selected for placing the selected blocks.

In element K4, the evaluation function shown in Eq. (10) is used to decide which block to choose for placing as defined in [19].

$$VCS_{\alpha}(b,r) = V(b) \times CS(b)^{\alpha} \tag{10}$$

Given a residual space r, each b block is sorted using Eq. (10). $V(b)$ represents the total volume of the boxes in block b. $CS(b)$ considers that a b block of dimensions $(l \times w \times h)$ is located at coordinate (x, y, z) of the container. Next, the covered surface area of block b by adjacent blocks and container walls are calculated. Then, normalized cover surface of block, $CS(b)$ in the range [0, 1], is computed as the covered surface of the block over the total block surface. α is a parameter that gives weight to the CS function.

In element K5, the blocks are placed in the following order: Near the corners, the edges, the faces of the container.

In element K6, as a search strategy, the GRASP method is used. The GRASP method has two stages: (1) Construction stage, (2) Local search stage. A solution is created by placing the blocks in the container step by step during the GRASP construction phase. Which block will be selected is determined by an evaluation function. A randomization strategy has been added to obtain different solutions. At each block selection, $VCS_{\alpha}(b,r)$ is calculated as the evaluation function for each of the blocks that can be placed in the residual space. $VCS_{\alpha}(b,r)$ provides an assessment of the block's fit to the residual space. A block is randomly selected among the blocks in the RCL with the highest evaluation function value among the blocks that can be placed at each step. RCL is created with the parameter δ $(0 < \delta < 1)$ from within the blocks that can be placed. The $100\delta\%$ blocks with the highest $VCS_{\alpha}(b,r)$ value is included in the list. This iteration continues until there are no more blocks that can be placed. This procedure corresponds to the element K4 in the block-building method. In the local search phase, a neighboring solution is obtained similar to a solution created during the construction phase. At this phase, the last placed $k\%$ blocks in the solution are removed from the container. The GRASP construction phase is repeated for the remaining space, but this time RCL consists of the best m blocks in that step.

During t time, GRASP construction phase and local search phase are repeated to find new solutions. The objective function of a solution is the ratio of total volume of the placed boxes to the volume of the container. The highest volume ratio is the output of the algorithm.

4. Results and Discussion

The proposed method is implemented in C++ language. Experiments were performed on a computer with an Intel Core i5 processor - 2.3 GHz Quad Core and 8 GB 2133 MHz memory.

BR datasets are used to solve this problem [20], [21]. The BR datasets consist of 15 tests. Since there are 100 problems in each dataset, there are 1500 problems in total. **Table 1** shows the variety of box types that increases from BR1 to BR15.

Table 1. *Box Types in Datasets*

Datasets	Number of Box Types
BR-1	3
BR-2	5
BR-3	8
BR-4	10
BR-5	12

BR-6	15
BR-7	20
BR-8	30
BR-9	40
BR-10	50
BR-11	60
BR-12	70
BR-13	80
BR-14	90
BR-15	100

The parameters used in the proposed BB-GRASP method are presented in **Table 2**.

Table 2. *Parameters for BB-GRASP Method*

t	10 seconds
δ	0.05
k	50
m	2
Maximum number of solutions	1000

The results of the BB-GRASP method are displayed in Table 3.

Table 3. *Results for BB-GRASP Method*

Dataset	Minimum Volume Ratio (%)	Maximum Volume Ratio (%)	**Average Volume Ratio (%)**
BR1	84.07	97.01	**93.46**
BR2	89.14	95.82	**93.51**
BR3	91.39	95.55	**93.58**
BR4	91.50	95.52	**93.34**
BR5	91.40	94.65	**93.17**
BR6	92.01	94.28	**93.11**
BR7	91.19	93.73	**92.60**
BR8	91.10	93.41	**92.38**
BR9	91.00	93.44	**92.27**
BR10	90.85	93.30	**92.02**
BR11	90.66	92.98	**91.88**
BR12	90.96	92.56	**91.80**
BR13	90.98	92.88	**91.80**
BR14	90.65	92.52	**91.63**
BR15	90.64	92.84	**91.62**

Table 3 shows the minimum, maximum and average volume ratio results of the BB-GRASP method. In the results, the maximum volume ratio varies between 93.58% and 91.62% for the BR datasets. Although different results are seen for different box types in minimum and maximum values, when the average is looked at, it is seen that there is a decrease in the maximum volume ratio as the box type increases.

Table 4. *Comparison with other heuristics*

Datasets	**BB-GRASP**	GRASP [3]	GRModGRASP [4]	SOA [11]	TS_CLP [10]
BR1	**93.46**	93.27	89.07	92.67	90.62
BR2	**93.51**	93.38	90.43	93.19	91.51
BR3	**93.58**	93.39	90.86	93.44	92.43
BR4	**93.34**	93.16	90.42	93.21	92.35
BR5	**93.17**	92.89	89.57	92.94	92.45
BR6	**93.11**	92.62	89.71	92.70	92.37
BR7	**92.60**	91.86	88.05	92.31	92.13

BR8	**92.38**	91.02	86.13	91.92	91.95
BR9	**92.27**	90.46	85.08	91.50	91.64
BR10	**92.02**	89.87	84.21	91.23	91.42
BR11	**91.88**	89.36	83.98	90.85	91.14
BR12	**91.80**	89.03	83.64	90.59	90.98
BR13	**91.80**	88.56	83.54	90.17	90.60
BR14	**91.63**	88.46	83.25	89.70	90.27
BR15	**91.62**	88.36	83.21	89.20	89.84

In **Table 4**, the BB-GRASP method is compared with the reported results of the SOA [11] which uses swarm optimization problem, TS_CLP [10] which uses simulated annealing algorithm, GRASP [3] and GRModGRASP [4] which uses the Grasp method in the literature. It is seen that better results are obtained for each dataset. Compared to other results, an improvement of at least 0.19% for BR1, 0.13% for BR2, 0.14% for BR3, 0.13% for BR4, 0.23% for BR5, 0.41% for BR6, 0.29% for BR7, 0.43% for BR8, 0.63% for BR9, 0.6% for BR10, 0.74% for BR11, 0.82% for BR12, 1.2% for BR13, 1.36% for BR14 and 1.78% for BR15 is observed. Even if the maximum volume ratio decreases as the box types increase, it is seen that a better volume ratio is obtained compared to other heuristics.

5. Conclusion

We proposed a block-building based GRASP method for the container loading problem. The block-building method was used for the placement of boxes and the GRASP method was applied as the search strategy. The performance of the proposed method was evaluated on the BR datasets and the results were compared with the results of some heuristic algorithms applied in the literature. According to the results, minimum 0.13% and maximum 1.78% improvement were obtained. The proposed method can help to reduce the costs of transportation which is crucial for domestic and international markets. Reducing the cost will increase the satisfaction between customers and organizations to higher levels. As a future work, some constraints that can be used in the CLP can be included or other heuristic algorithms can be used as a search strategy to increase the maximum volume ratio.

References

[1] Sheng L, Hongxia Z, Xisong D, Changjian C. "A Heuristic Algorithm for Container Loading of Pallets with Infill Boxes ". European Journal of Operational Research 252 (2016) 728-736.

[2] Karaboğa D. Yapay Zeka Optimizasyon Algoritmaları. Nobel Yayın Dağıtım, 2011.

[3] Alataş B. Kaotik Haritalı Parçacık Sürü Optimizasyon Algoritmaları Geliştirme. Doktora Tezi, Fırat Üniversitesi, Elazığ, Türkiye, 2007.

[4] Parreno F, Alvarez-Valdes R, Tamarit JM, Oliveira JF. "A maximal-space algorithm for the container loading problem ". INFORMS Journal on Computing, 20(3) 2008 412-422.

[5] Moura A, Oliveira JF. "A GRASP approach to the container-loading problem ". IEEE Intelligent Systems, 20(4) 2005 50-57.

[6] Gehring H, Bortfeldt A. "A Parallel Genetic Algorithm for Solving the Container Loading Problem ". International Transactions in Operational Research 9, 2002 497-511.

[7] Dereli T, Daş GS. "Konteyner Yükleme Problemleri için Karınca Koloni Optimizasyonu Yaklaşımı ". Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi, 25(4) 2010 881-894.

[8] Koyuncuoğlu MU. Bir Konteyner Terminalinde İstif Vinçlerinin Meta Sezgisel Yöntemler Kullanarak Çizgelenmesi. Yüksek Lisans Tezi, Pamukkale Üniversitesi Fen Bilimleri Enstitüsü, Denizli, Türkiye, 2012.

[9] Ceschia S, Schaerf A. "Local search for a multi-drop multi-container loading problem ". Journal of Heuristics, 19(2) 2013 275-294.

[10] Can O, Sahingoz OK. "Solving container loading problem with simulated annealing algorithm ". 15th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary, 19-21 November 2014.

[11] Sheng L, Xiuqin S, Changjian C, Hongxia Z, Dayong S, Feiyue W. "Heuristic Algorithm for the Container Loading Problem with Multiple Constraints ". Computers & Industrial Engineering 108, 2017 149-164.

[12] Zhou Q, Liu X. "A Swarm Optimization Algorithm for Practical Container Loading Problem ". IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, China, 29 Oct – 1 Nov 2017.

[13] Gehring H, Bortfeldt A. "A Genetic Algorithm for Solving the Container Loading Problem ". International Transactions in Operational Research 4, 1997 401-418.

[14] Huang Y, Hwang FJ, Lu H. "An effective placement method for the single container loading problem ". Computers & Industrial Engineering 97, 2016 212-221.

[15] Feo T, Resende MGC. "A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem ". Operations Research Letters 8, 1989 67-71.

[16] Feo T, Resende MGC. "Greedy Randomized Adaptive Search Procedures ". Journal of Global Optimization 6, 1995 109-133.

[17] Festa P, Resende MGC. "GRASP: basic components and enhancements ". Telecommunication Systems, 46(3) 2011 253-271.

[18] Zhu W, Oon W, Lim A, Weng Y. "The six elements to block-building approaches for the single container loading problem ". Applied Intelligence 2012, 37 2012 431-445.

[19] Araya I, Guerrero K, Nunez E. "VCS: A new heuristic function for selecting boxes in the single container loading problem. ". Computers & Operations Research 82, 2017 27-35.

[20] Bischoff EE, Ratcliff MSW. "Issues in the development of Approaches to Container Loading ". Omega-International Journal of Management Science, 23(4) 1995 377-390.

[21] Davies AP, Bischoff EE. "Weight distribution considerations in container loading ". European Journal of Operational Research, 114(3) 1999 509-527.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 1), pp. 19-26, 2023
doi: 10.54569/aair.1143632
Published online: Jan 9, 2023

Detection of Mealybugs Disease Using Artificial Intelligence Methods

Bekir Aksoy [1,*], Nergiz Aydın [1], Sema Çayır [1], Osamah Khaled Musleh SALMAN[1]

[1] Isparta University of Applied Sciences, Faculty of Technology, Mechatronic Engineering, Isparta, TÜRKİYE

Abstract

Today, the need for agricultural lands has increased even more due to the increasing population density. For this reason, increasing the yield of crops in agricultural areas becomes a very important need. It is very important to minimize the pests that negatively affect plant productivity in agricultural areas. In the study, it was aimed to detect the mealybug disease, which negatively affects plant productivity in agricultural areas, by using artificial intelligence methods. 539 disease-bearing and disease-free plant images collected from open access websites were used. These images are classified by VGG-16, Resnet-34 and Squeezenet deep learning algorithms. The most successful among the three architectures was determined as the VGG-16 and ResNet-34 model with an accuracy rate of 97%.

Keywords: *Deep learning; classification; mealy lice disease.*

1. Introduction

In order to meet the most basic food and clothing needs of humanity, agricultural production must increase with the increasing world population today. For this reason, pest control methods have an important place in organic agriculture in terms of productivity increase, environment and human health [1]. Today, agriculture is one of the most important strategic sectors in the world [2] and it brings 15-20% return to the country's economy [3]. However, problems caused by plant diseases and pests in the agricultural sector are one of the important causes of productivity loss. For this reason, it is aimed to reduce the loss of productivity by collecting data on the factors affecting the loss of productivity in the agricultural sector and analyzing the data using artificial intelligence methods [4]. Artificial intelligence systems perform many functions such as obtaining information, perception, learning, thinking and decision-making by analyzing the mental functions related to intelligence in humans with computer models and applying them to different agricultural systems by making meaningful results [5].

One of the important pests in the agricultural sector is known as mealybugs

disease. Mealy louse is a polyphagous pest that is seen in all ecozones, especially in the nearctic and palearctic, which are frequently encountered in agriculture due to sudden temperature changes and sudden environmental changes [6]. The main hosts are citrus species and varieties, and they cause significant damage to figs, vines, pomegranates, pears, greenhouses and ornamental plants [7]. There are 92 pests, 34 diseases, 16 nematodes and 155 weed species in the citrus orchards of Turkey, which may adversely affect the cultivation. Citrus mealybug, with its Latin name Planoccus citri Risso (Hemiptera: Pseudoccidae), is one of the important pests that we have encountered in this group in recent years [8]. Citrus mealybug damages plants directly by sucking (by sucking the plant sap) and indirectly by causing fumagine by secreting a honey-like substance [9]. In the vine plant, it spreads all over the plant and causes significant damage to the leaves, shoots, clusters and stems. For this reason, the mealy louse sucks the sap of the plant, causing the vine to weaken, lose yield and eventually dry out [10].

Biological, chemical and traditional methods used in the fight against mealybugs have important deficiencies due to the fact that the area to be combated is not well defined, and the method to be used is randomly applied to the area to be applied. The lack of selectivity in the methods used causes different problems. One of the important problems is the destruction of useful creatures due to pesticides applied to unwanted points. On the other hand, the inability to predict the amount of chemical, biological or conventional application to be used causes unnecessary product consumption and time losses.

Late detection of diseases especially seen in short lived plants reduces the quality and quantity of the product. Detection of diseases or pests in plants with artificial neural networks and image processing techniques has begun to increase. Detection of diseases on plants has become easier by using CNN models. Thus, it has made important contributions to the rapid decision-making on the spraying methods to be applied to diseases and pests in plants [11]. The plants are introduced to the computer using image processing techniques and the classification process takes place. Plants and pests are detected quickly with preprocessing [12].

When the literature is examined, many studies have been carried out on the separation of pests in plants with artificial intelligence and image processing.

Togacar et al. In their study, they classified the flower images by applying the feature selection method to

*Corresponding author
E-mail address: bekiraksoy@isparta.edu.tr

Received: 13/July/2022; Received in revised form: 19/Oct/2022; Accepted: 12/Nov/2022.

the Convolutional Neural Networks (CNN) architecture and achieved 91.10% success [13]. Aksoy et al., by using deep learning methods named AlexNet, DenseNet-121, ResNet-34, VGG16-BN and Squeezenet1_0, classified the diseases of apple plants with an accuracy rate of 99.52% [14]. Soydan detected rice burn disease by using image processing and artificial neural networks in his study. In the study, the images obtained from the ground and from the air with the drone were compared with the real diseased field data, and 100% accuracy was obtained [15]. Altas et al. determined the scabies disease in the vine leaves by using the image processing toolbox module of MATLAB. When the expert observation data and image processing technique data were compared, they observed that the results were very close to each other [16]. Sardogan detected diseases in apple and tomato plant leaves using the MATLAB program. To train the model, an CNN based model was created and the results were classified by the Learning Vector Quantization algorithm. He obtained an accuracy rate of 86% using a single filter on tomato leaves and 98.5% using four different filters on apple leaves [17]. Yaman et al. In their study, they detected leaf diseases in plants using deep learning model and feature selection method. They combined the data obtained from DarkNet-53 and ResNet-101 models and obtained 99.58% accuracy with a hybrid method [18]. In his study, Ghoury detected diseases in grape fruit and grape leaves by using the previously trained SSD_MobileNet v1 and Faster-R-CNN Inception v2 deep CNN models with a transfer teaching approach. Tested all test data with Faster-R-CNN Inception v2, it achieved an accuracy of 95.57%, and an accuracy of 59.29% with SSD_MobileNet v1 [19]. Ozerdem et al. In their study, the Gray Level Co-occurrence Matrix (GLCM) feature extraction method, which classifies the rust disease in lily plants in black and white form, and Multi-Layer Perceptron (MLP), K-Nearest Neighbors (k-NN), Least Squares Support Vector Machine (LS-SVM) achieved an accuracy rate of 88.9% using machine learning algorithms [20]. In the Kızılboğa study, seven different diseases seen in apple and quince plants were classified using the evolved deep network model, transfer learning methods (VGG16, Inception and ResNet deep network architectures) and SVM methods. It performed 88% success with CNN, feature extraction with 89.75% success with different CNN based InceptionV3 architecture, and classification with 89.5% accuracy [21]. Turkoglu et al. In their study, they classified the freckle disease seen in apricot plants with AlexNet, VGG-16 and VGG-19 deep learning models and classified the feature extraction obtained by k-NN method. They obtained 94.8% accuracy with the VGG-16 model, which is one of the deep learning architectures [22]. In Hayıt study, classified the yellow rust disease in wheat plants using artificial neural networks and nine methods together with traditional methods. He used CNN based Yellow, Rust and Xception models in his classification by completely applying a deep learning model. With the Xception model, one of the CNN models, an accuracy rate of 91% was obtained [23]. Kilic et al. In their study, they detected the diseases that occur on the leves of citrus plants using CNN, AlexNet and VGG16 artificial neural networks and achieved a92% accuracy rate [11]. In Aslan's study, he classified the diseases in the peach plant using an ESA-based AlexNet deep learning model and achieved an accuracy rate of 99.3% [24].

In the first stage of the study, the fight against pests encountered in agriculture and the negative effects of these, and the effects of mealybugs disease on the plant were discussed. In the second stage, academic studies carried out for the detection of plant pests using artificial intelligence methods were examined. In the third stage, the data set to be used in artificial intelligence was created with the images of the mealybug pests obtained from the open access internet pages. The created image dataset is trained with SqueezeNet, VGG-16 and Resnet34 CNN based deep learning architectures. CNN based deep learning architectures trained in the fourth stage were evaluated with F-Score, sensitivity, originality and accuracy performance evaluation criteria.

2. Material and Methods

The study material section consists of the data set used in the study, CNN-based deep learning architectures and performance evaluation criteria. In the method part of the study, the information about the work flow diagram of the study is discussed in detail below

2.1. Material

2.1.1. Data Set

The data set used in the study was collected from different open access websites. The collected data set consists of two classes, images with and without mealybugs disease. The dataset includes 209 images of diseased plants infected by mealybugs and 330 images of healthy citrus, vine, pomegranate, pear and cactus plants. An example image of the data set used in the study is given in **Figure 1** and the numerical contents of the images used in the study are given in **Table 1**.

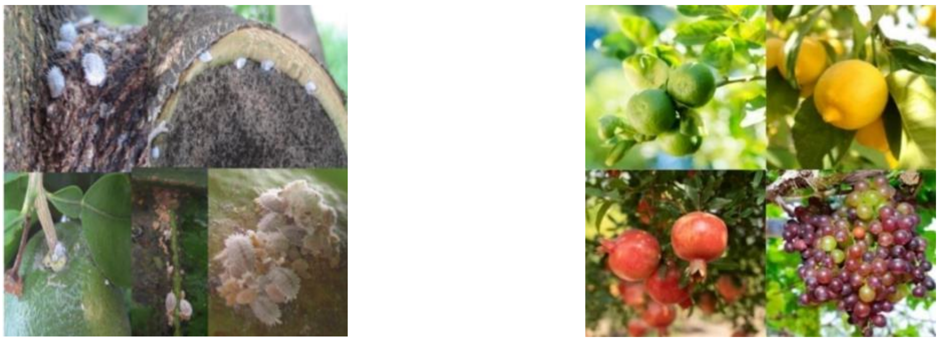

(a) (b)

Figure 1. *(a) View of mealybug spread on the plant [25]. (b) Healthy plant images in the dataset.*

Table 1. *Plant varieties and numbers in the data set*

Plants	Unhealty Plant Data Set	Healty Plant Data Set
GreenhousePlant	50	30
Citrus	67	111
Grape	33	99
Pomegranate	29	58
Pear	25	32

2.1.2 Esa-Based Models Used in The Study

Three different deep learning models were used in the training phase. These are VGG-16, ResNet-34 and SqueezNet.

2.1.2.1. VGG-16 Deep Learning Architecture

The VGG-16 architecture was presented by Simonyan and Zisserman in 2015 [26]. The VGG-16 architecture consists of 16 layers (13 convolutional, 3 fully connected layers) and is often used in object classification applications [27]. VGG-16 has a deeper (network depth = 16) architecture with more trainable parameters (about 138M) compared to AlexNet [28]. In **Figure 2**, the visual of the VGG-16 architecture is given. [29].

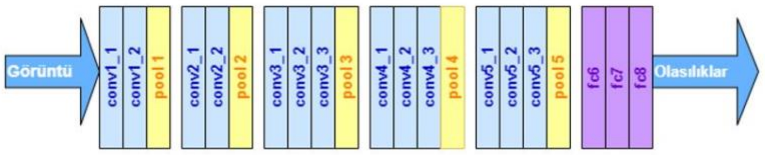


Figure 2. *Architecture of VGG16*

2.1.2.2. ResNet-34 Deep Learning Architecture

ResNet-34 is the type of CNN used in 2015 by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their “Deep Residual Learning for Image Recognition” study to facilitate training of networks that are significantly deeper. The biggest difference from other architectures is that the blocks feeding the next layers are also added to the model. The ResNet-34 architecture is shown in **Figure 3** [30].

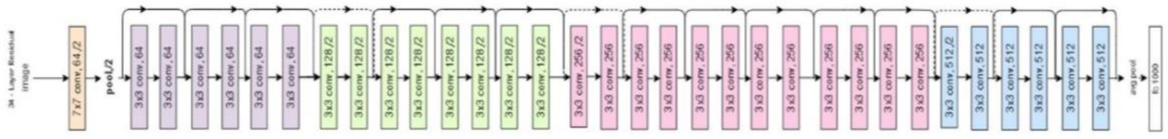


Figure 3. *Network architecture of the first 34 layers of Microsoft ResNet*

2.1.2.3. SqueezeNet Deep Learning Architecture

The SqueezeNet architecture was developed by Landola et al. It was presented by in 2016 [31]. SqueezeNet is an CNN network with 1.24M trainable parameters despite a mesh depth of 18 [31]. With this architecture, it is possible to create a neural network with fewer parameters and achieve AlexNet level accuracy with 50 times less parameters [32].

2.1.3 Performance Evaluation Criteria

There are many different performance evaluation criteria to evaluate the models that determine which class the observations obtained in classification problems belong to. The accuracy of the model is determined by comparing the accuracy value obtained in the classification method with the actual accuracy values [33-34]. The complexity matrix was used to evaluate the model in the study. The complexity matrix is a two-dimensional evaluation criterion, kxk as positivity and negativity, which shows the true accuracy value and the estimated accuracy value when classifying. The complexity matrix evaluation criterion is given in **Table 2**. [14, 34].

Table 2. *Complexity Matrix*

REAL VALUE	ESTIMATED VALUE		
		Positive	Negative
	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Sensitivity, originality, accuracy and F-score performance evaluation criteria were used to evaluate the performances of the deep learning models used in the study. Mathematical equations of performance evaluation criteria are given in equations 1-4 [12].

$$Sensitivity = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1}$$

$$Orıgınalıty = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} \tag{2}$$

$$Accuracy \ = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{3}$$

$$F - Skor \ = \frac{2 * \mathrm{TP}}{2 * \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{4}$$

2.2. Method

The work flow diagram is shown in **Figure 4**. First, a total of 539 images collected from open access websites were reduced to 224x224 size. The images brought to the size of 224x224 were randomly divided into 80% training and 20% testing. In the next step, the training dataset was trained with VGG-16, Resnet-34 and SqueezeNet deep learning architectures with 20 repetitive training. During the training, the learning rate was optimized for each model and the most appropriate learning rate was selected. By using the feature extraction obtained as a result of the training, the classification process was completed with the classical neural network.

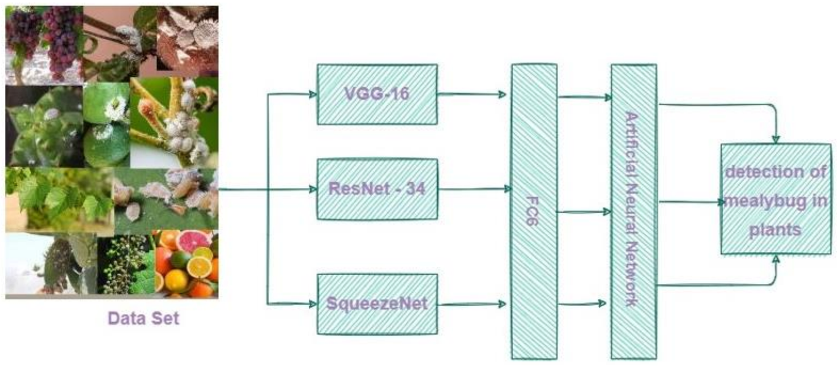
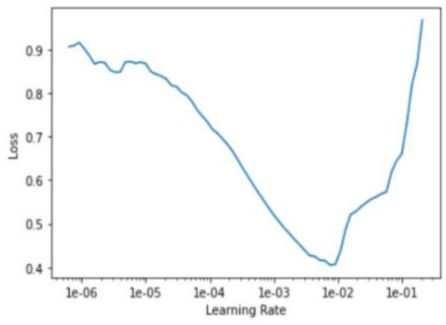


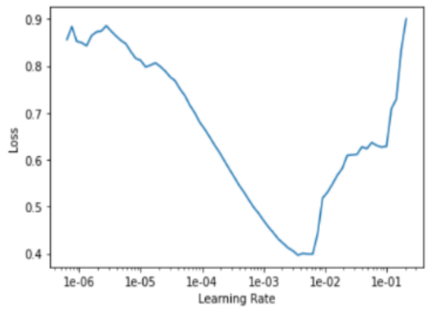
Figure 4. *Workflow Diagram*

3. Research findings

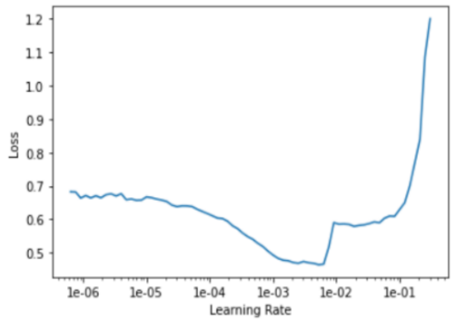
In the study, pests were detected by using ResNet-34, SqueezeNet and VGG-16 deep learning model. It has been trained in two classes as diseased plants and healthy plants infected by mealy lice. After the training process, the learning rate is optimized to increase the accuracy of the model. In the optimization process, the learning rate-loss graph was drawn and the most appropriate learning rate was selected in the decline region, and the model was retrained **Figure 5** shows the learning rate-loss graphs.



(a)VGG-16



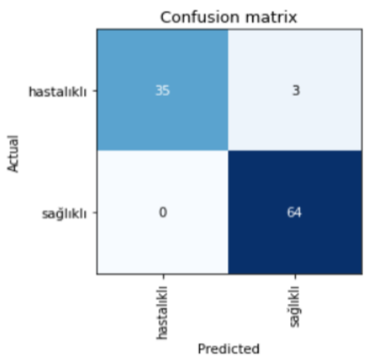
(b) ResNet-34



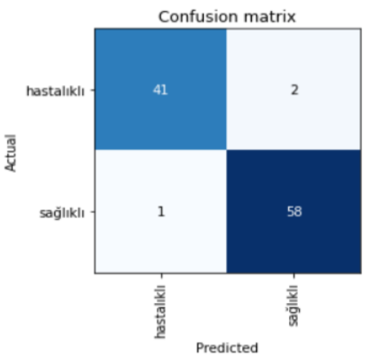
(c) SqueezeNet

Figure 5. *Learning rate-loss graphs (a-b-c).*

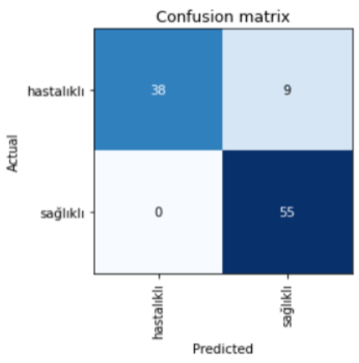
The models were retrained according to the learning rate selected in the learning rate-loss graph. The complexity matrix results obtained by evaluating the trained models with the test data are shown in **Figure 6**.



(a) VGG-16



(b) ResNet-34



(c) SqueezeNet

Figure 6. *Complexity matrix (a-b-c)*

When the confusion matrix was examined, it was determined that the VGG-16 model detected 35 correct and 3 incorrect images out of 38 diseased plant test images, and correctly predicted all 64 healthy plant test images. It is seen that the ResNet-34 model detected 41 correct and 2 incorrect images out of 43 diseased plant test images, and predicted 58 correct and 1 incorrectly predicted out of 59 healthy plant test images. It was determined that the SqueezeNet model detected 38 correct and 9 incorrect images out of 47 diseased plant test images, and predicted all 55 healthy plant test images correctly. The results of the three deep learning architectures used in the study according to sensitivity, originality, accuracy and F-score performance evaluation criteria are given in **Table 3**.

Table 3. *Performance Evaluation Criteria of Models*

Model	Sensitivity	Originality	Accuracy	F-score
VG G-16	0.921	1	0.970	0.958
ResNet-34	0.953	0.983	0.970	0.964
SqueezeNet	0.808	1	0.911	0.894

When **Table 3** is examined, it is seen that more than 90% accuracy was obtained from all three models. Among these three models, VGG-16 and ResNet-34 models were found to be the most successful models.

4. Discussions

The increasing world population has also triggered an increase in the needs in the field of agriculture. In order to meet the agricultural demands, it is necessary to make the right agricultural movements. Preventing productivity losses in agriculture is possible with the development of new and useful systems in parallel with today's technology, and it is important in terms of meeting agricultural needs. Productivity losses caused by mealybugs disease are also a danger in agriculture.

Preventing productivity losses by using artificial intelligence and image processing methods to mealybugs supports sustainability in agriculture. In this respect, Onur Ağin et al. they discussed the sustainability of agriculture in their study titled "The Role and Importance of Image Processing Techniques in Sustainable Agriculture" and pointed out that the use of image processing techniques in various fields of agriculture (spraying, irrigation, fertilization, etc.) is beneficial [35].

With the image processing techniques and deep learning methods used in the study, different plant species in agriculture were handled and classification and separation studies were carried out. Kadir Sabancı et al. In their study titled "Determination of Potato Classification Parameters with the Help of Image Processing and Artificial Neural Networks", they made an example study of the use of image processing in the agricultural field. The study was carried out with the help of image processing and artificial neural networks for the dimensional classification of potatoes and the differentiation of good quality potatoes [36]. Kaymak et al. On the other hand, in their study named “The Example of Use of Image Processing Technologies for Apple Orchards”, they aimed to detect and count the red apples on the trees in an apple orchard with image processing techniques. In order to realize this aim, a software has been developed in the computer environment. The software successfully finds the apples on the tree by using digitally transferred apple tree images and image processing techniques [37].

The successes obtained from the image processing studies have led to the design of machines in which image processing techniques are integrated in the future. For example, Melih Kuncan et al. In their project named “Image Processing Based Olive Sorting Machine”, they aimed to determine the olives according to their color and to perform the separation process. An electromechanical system has been developed for this process, and the image processing algorithms developed within the scope of the study have been tested on this system in real time and the results have been observed [38]. Kadir SABANCI et al. With the precision spraying robot developed in their project named “Image Processing-Based Precision Spraying Robot”, weeds in between rows in sugar beet agriculture were detected using image processing techniques and they performed a variable-level herbicide application model on the weed [39]. Bahadır ŞİN et al. In their project titled “Weed Detection Using Unmanned Aerial Vehicle (UAV) and Image Processing Techniques”, they used image processing methods to detect weeds [40].

Thus, artificial intelligence and image processing methods have been used in agriculture many times in different studies and have been integrated into projects. It is envisaged that the detection of mealybugs disease by image processing will be used in machinery and systems to be developed to prevent productivity losses in agriculture in the future.

5. Conclusion

Artificial intelligence applications have become a frequently used method due to the contributions it provides in applications in the agricultural sector today. It is very important to minimize the yield losses by detecting diseases and pests in plants by using artificial intelligence applications.

In the study, diseased and healthy plant images in the data set created from open access websites were trained with VGG-16, ResNet-34 and SqueezeNet deep learning models. The results were evaluated by plotting the success of the models, the learning rate-loss graph and the confusion matrix. Obtaining more than 90% accuracy in all three models shows that the data set is compatible with the deep learning models used in the study.

With the study carried out, artificial intelligence-based models were presented to prevent plant productivity losses caused by mealybugs disease. With the models presented, it is aimed to contribute to the academic literature for the diagnosis of similar diseases in plants. In future studies, it is aimed to increase the accuracy rates by expanding the data set and using different artificial intelligence models.

Acknowledgement:

Thanks to the open internet sites for the images used to create the dataset. Presented in abstract at the 4th International Conference on Artificial Intelligence and Applied Mathematics in Engineering (ICAIAME 2022)

References

[1] Tanrikulu, Y. (2019). Organik tarımda zararlılarla mücadele yöntemleri (Master'sthesis, Fen Bilimleri Enstitüsü).
[2] Uzundumlu, A. S. (2012). Tarım sektörünün ülke ekonomisindeki yeri ve önemi. AlinteriJournal of Agriculture Science, 22(1), 34-44.
[3] Direk, M. (2012). Tarım tarihi ve deontoloji. Eğitim Yayınevi.
[4] Uzun, Y., Bilban, M., & Arıkan, H. Tarım ve Kırsal Kalkınmada Yapay Zekâ Kullanımı.
[5] Bozüyük, T., Yağci, C., Gökçe, İ., & Akar G. (2005). Yapay Zekâ Teknolojilerinin Endüstrideki Uygulamaları.
[6] Bodenheimer, F. S. (1953). The Coccoidea of Turkey III. Revue de la Facultédes Sciences de l'Universitéd'Istanbul (Ser. B)., 18, 91-164.
[7] Williams, D. J., & Granara de Willink, M. C. (1992). Mealybugs of Central and South America (No. QL527. P83 W71).
[8] Uygun, N., & Satar, S. (2008). The current situation of citruspests and their control methods in Turkey. IOBC-WPRS Bulletin, 38, 2-9.
[9] Aydın İl Tarım ve Orman Müdürlüğühttps://aydin.tarimorman.gov.tr/Haber/271/Turuncgil-Bahcelerinde-Unlu-Bit-Zararlisi-Ile-Mucadele-Zamani-Geldi. [Erişim Tarihi: 22 May 2015]
[10] Williams, D. J. (2004). Mealybugs of southern Asia. Natural History Museum jointly with Southdene.
[11] Kiliç, E., Ecemiş, İ. N., & İlhan, H. O. (2021, October). Narenciye Ağaç Yaprak Hastalıklarının Evrişimli Sinir Ağları ile Sınıflandırılması. In 2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT) (pp. 452-456). IEEE.
[12] Altaş, Z., Özgüven, M. M., & Yanar, Y. Bitki hastalık ve zararlı düzeylerinin belirlenmesinde görüntü işleme tekniklerinin kullanımı: şeker pancarı yaprak leke hastalığı örneği.
[13] Toğaçar, M., Ergen, B., & Özyurt, F. (2020). Evrişimsel Sinir Ağı Modellerinde Özellik Seçim Yöntemlerini Kullanarak Çiçek Görüntülerinin Sınıflandırılması. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 32(1), 47-56.
[14] Aksoy, B., Halis, H. D., & Salman, O. K. M. (2020). Elma bitkisindeki hastalıkların yapay zekâ yöntemleri ile tespiti ve yapay zekâ yöntemlerinin performanslarının karşılaştırılması. International Journal of Engineering and Innovative Research, 2(3), 194-210.
[15] Soydan, O. (2020). Çeltik yanıklığı hastalığının görüntü işleme teknikleri kullanılarak tespit edilmesi (Master'sthesis, Fen Bilimleri Enstitüsü).
[16] Altaş, Z., Özgüven, M. M., & Dilmaç, M. Görüntü İşleme Teknikleri ile Bağ Yaprak Uyuzu Hasarının Belirlenmesi. Gaziosmanpaşa Bilimsel Araştırma Dergisi, 10(3), 77-87.
[17] Sardoğan Doğan, M. (2019). Bitkilerde görülen hastalıkların derin öğrenme yöntemleriyle tespiti ve sınıflandırılması (Master'sthesis, Fen Bilimleri Enstitüsü).
[18] Yaman, O., & Tuncer, T. Bitkilerdeki Yaprak Hastalığı Tespiti için Derin Özellik Çıkarma ve Makine Öğrenmesi Yöntemi. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 34(1), 123-132.
[19] Ghoury, S. Konvolüsyonel sinir ağlarını kullanarak üzüm ve üzüm yapraklarının hastalıklarının tespit edilmesi (Master's thesis, Fen Bilimleri Enstitüsü).
[20] Özerdem, M. S., & Acar E. (2011). Zambak yaprağı imgelerinde pas hastalıklarının GLCM tabanlı sınıflandırma yöntemleri ile tespiti. Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, 2(2), 95-105.
[21] Kızılboğa, A. Y. (2021). Yapay sinir ağları ve derin öğrenme teknikleri kullanılarak elma ve ayvada çeşitli hastalıkların tespit edilmesi (Master'sthesis, Fen Bilimleri Enstitüsü/Bilgisayar Mühendisliği Ana Bilim Dalı).
[22] Türkoğlu, M., & Hanbay, D. (2018, September). Apricot disease identification based on attributes obtained from deep learning algorithms. In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP) (pp. 1-4). IEEE.
[23] Hayıt, T. (2021). Buğdayda sarı pas (Pucciniastriiformis) hastalığının değerlendirilmesi için görüntü işleme teknikleri kullanılarak bir karar destek sisteminin geliştirilmesi.
[24] Aslan, M. (2021). Derin Öğrenme ile Şeftali Hastalıkların Tespiti. Avrupa Bilim ve Teknoloji Dergisi, (23), 540-546.
[25] Yayla, M. (2018). Sympherobius pygmaeus (Rambur) (Neuroptera: Hemerobıidae)un turunçgil bahçelerinde Planococcuscitri (Rissio)(Hemiptera: Pseudococcidae) nin mücadelesine yönelik kullanım olanaklarının araştırılması.
[26] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
[27] Demir, U., & Ünal, G. (2017, May). Inpainting by deep autoencoders using an advisor network. In 2017 25th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.
[28] Ünay, D. (2020). Meyve ve Sebzelerin Otomatik Sınıflandırılması için Farklı Derin Öğrenme Yöntemlerinin Karşılaştırılmalı Analizi. Turkish Studies-Information Technologies and Applied Sciences, 15(4), 533-544.
[29] Doğan, F., & Türkoğlu, İ. (2018). Derin öğrenme algoritmalarının yaprak sınıflandırma başarımlarının karşılaştırılması. Sakarya University Journal of Computer and Information Sciences, 1(1), 10-21.
[30] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deepresidual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[31] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.
[32] Özyurt, F., Sert, E. ve Avcı, D. (2020). Beyin tümörü tespiti için uzman bir sistem: Bulanık C-süper çözünürlük ve aşırı öğrenme makinesi ile evrişimli sinir ağı ile araçlar. Tıbbi hipotezler, 134, 109433.
[33] Deng X, Liu Q, Deng Y, Mahadevan S. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. Information Sciences. 2016; 340:250-261.

[34] Ruuska, S., Hämäläinen, W., Kajava, S., Mughal, M., Matilainen, P., & Mononen, J. (2018). Evaluation of the confusion matrix method in the validation of an automated system for measuring feeding behaviour of cattle. Behavioural processes, 148, 56-62.

[35] Ağın O. & Malaslı, M. Z. (2016). Görüntü işleme tekniklerinin sürdürülebilir tarımdaki yeri ve önemi: Literatür çalışması. Tarım Makinaları Bilimi Dergisi, 12(3), 199-206.

[36] Sabancı, K., Aydın, C., & Ünlerşen, M. F. (2012). Görüntü işleme ve yapay sinir ağları yardımıyla patates sınıflandırma parametrelerinin belirlenmesi. Journal of the Institute of Science and Technology, 2(2 Sp: A), 59-62.

[37] Kaymak, A. M., Örnek, M. N., & Kahramanlı, H. (2019). Görüntü işleme teknolojilerinin elma bahçelerine yönelik kullanim örneği. Uluborlu Mesleki Bilimler Dergisi, 2(1), 17-26.

[38] Kuncan, M., Ertunç, H. M., Küçükyıldız, G., Hızarcı, B., Ocak, H., & Öztürk, S. (2013). Görüntü işleme tabanlı zeytin ayıklama makinesi. Otomatik Kontrol Ulusal Toplantısı, 459-464.

[39] Sabancı, K., & Aydın, C. (2014). Görüntü işleme tabanlı hassas ilaçlama robotu. Journal of Agricultural Sciences, 20(4), 406-414.

[40] Şin B., & Kadıoğlu İ. (2019). İnsansız Hava Aracı (İHA) ve Görüntü İşleme Teknikleri Kullanılarak Yabancı Ot TespitininYapılması. *Turkish Journal of Weed Science*, *22*(2), 211-217

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 1), pp. 27-35, 2023
doi: 10.54569/aair.1164731
Published online: Jan 24, 2023

Detecting high-risk Area for Lumpy Skin Disease in Cattle Using Deep Learning Feature

Musa Genemo [1,*]

[1] Department of Computer Engineering, Gumushane University, Gumushane/Turkey

Abstract

Cattle's lumpy skin disease is a viral disease that transmits by blood-feeding insects like mosquitoes. The disease mostly affects animals that have not previously been exposed to the virus. Cattle lumpy skin disease impacts milk, beef, and national and international livestock trade. Traditional lumpy skin disease diagnosis is very difficult due to, the lack of materials, experts, and time-consuming. Due to this, it is crucial to use deep learning algorithms with the ability to classify the disease with high accuracy performance results. Therefore, Deep learning-based segmentation and classification are proposed for disease segmentation and classification by using deep features. For this, 10 layers of Convolutional Neural Networks have been chosen. The developed framework is initially trained on a collected Cattle's lumpy Skin Disease (CLSD) dataset. The features are extracted from input images; hence the color of the skin is very important to identify the affected area during disease representation we used a color histogram. This segmented area of affected skin color is used for feature extraction by a deep pre-trained CNN. Then the generated result is converted into a binary using a threshold. The Extreme learning machine (ELM) classifier is used for classification. The classification performance of the proposed methodology achieved an accuracy of 0.9012% on CLSD To prove the effectiveness of the proposed methods, we present a comparison with the state-of-the-art techniques.

Keywords**:** *Classification, deep learning, lumpy skin disease, optimization, segmentation*

1. Introduction

Cattle's lumpy skin disease is a viral disease that transmits by blood-feeding insects like mosquitoes [1]. The livestock sector globally is highly dynamic, contributes 40 % of the global value of agricultural output, and supports the livelihoods and food security of almost a billion people [2]. Keeping the safety of livestock sectors is safe as securing the life of those millions of dependents on livestock at specific and caring for the world at large.

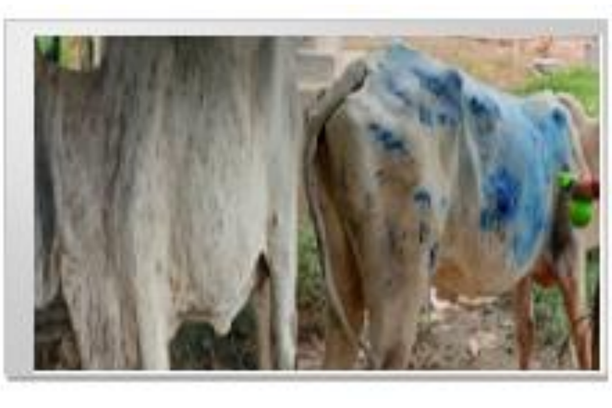


Figure 1. Lumpy skin disease

The lumpy skin disease is a more virulent cow disease that affects most cattle. The disease is contagious, and it has the potential to spread across borders, affecting neighboring countries. Due to lower output and restrictions on the international trade of live animals and animal products, the disease causes enormous economic losses. As a result, the disease's transmission and spread are tied to warm and humid climatic conditions associated with high biting arthropod population densities [2, 3].

Most cattle owners (farmers) follow a set of stages for lumpy skin disease treatment, beginning with traditional medicine (like watering some natural leaf, burning the area where they found the problem), and finally taking to Veterinary Doctor. This is time-consuming, and the diseases are diagnosed far too soon. Other

*Corresponding author: Musa Dima Genemo
E-mail address: musa.ju2002@gmail.com

Received: 20/Aug/2022; Received in revised form: 3/Nov/2022; Accepted: 12/Nov/2022.

issues, particularly in underdeveloped countries in Sub-Saharan Africa, include the inability to find a veterinarian.

Professional doctors are hard to come by, especially in the Continent of Africa. As a result, machine learning approaches play a crucial role in lumpy skin disease early treatment. Machine learning offers an alternative to challenges, is quicker, and is far more accurate in both combating and detecting disease. In contrast, machine Learning approaches are superior to manual detection and treatment. Convolutional neural networks (CNNs) with multiple convolutional layers are typically used in deep learning feature extraction [4,5].

In the first phase, a deep CNN method is used to locate the area of an infected location. In the second phase, the extracted feature must then be classified into the appropriate category, such as lumpy skin disease or non-lumpy skin disease. There are several difficulties during the classification process because of the following factors: (i) there isn't much of a color difference between various skin diseases and lumpy skin-affected regions, and they can even be mixed; in this scenario, confusion and incorrect conclusions may result. (ii) detection is challenging due to camera resolution quality. (iii) there is no database prepared for this purpose to the best of my knowledge. Therefore, it will be difficult to compare the state of the art. However, we use alternative solutions to overcome these issues.

According to the findings of the study, selecting the best features and using the results in feature fusion yields promising results [6-8]. As a result, we used multiple Fusion approaches to increase the number of features gathered from a Region of Interest (ROI) that is accessible from a variety of locations. Even though this phase increases the number of predictors, which increases computational time, we chose the best features using a meta-heuristic approach feature selection strategy. Based on the selection process, meta-heuristic procedures are more useful and have a reduced number of predictors [9]. In section two of this works several fusion and selection approaches have been discussed.

We employ the cattle's lumpy skin disease in this study, which is divided into two categories: lumpy skin disease and non-lumpy skin disease. Because the images in this dataset are not evenly distributed, training a CNN model with it is extremely difficult. Ear, back, pin, tail, thigh, toe, stomach, elbow, chest, brisket, neck, and many features such as hand, face, neck, foot, and so on are all included in this dataset. Each class has a varied number of images in it.

The key challenges in using these datasets include low contrast in the affected area, high irregularity, and lumpy in the joint area. This article introduces a mechanism for segmenting and classifying lumpy skin disease images into lumpy skin disease and non-lumpy skin disease. The foremost contributions are mentioned as under:

i. A dataset of different Cattel’s Lumpy Skin Disease (CLSD)is prepared.
ii. For skin image enhancement we presented local color-controlled histogram intensity values (LCcHIV), to boost the local contrast of a lumpy region.
iii. We offer a novel 10-layer CNN-based deep learning-based technique for segmenting lumpy regions.
iv. Finally, we applied the Extreme Learning Machine for classification.

The rest of the manuscript is organized as follows: The introductory section and the literature review section are included in Sections 1 and 2, respectively. Materials and methods are presented in Section 3, with a detailed mathematical explanation and visual results. Section 4 discusses the experimental setup and results. Finally, the conclusions are given in Section 5.

2. Literature Review

A lot of work has been done in the lumpy skin disease and human cancer detection and segmentation area but, to the best of my knowledge, there are not too many technical studies exist in the computer vision field for cattle’s lumps skin disease. I only found theoretical work related to this study. The theoretical works I found have been presented on qualitative assessment transmission of lumpy skin disease [9]. To find the transmission possibility of the disease they used probability to assess the risk. Lately, several computer vision and machine learning-based methods are introduced for the segmentation and classification of diseases in human health. An automated approach for lung cancer classification based on classical and transfer learning from a chest radiograph [10]. The introduced system consists of two major stages segmentation and classification. An approach for feature selection is adopted which chooses the optimum features for final recognition. The system achieved higher than 90% accuracy for all considered disease types. Human skin cancer is also discussed in some literature some of which are discussed in this section. In this work, they tried to classify dark spots/bubbles around found in the human body [11]. The high pass filter is used to highlight the edges; further, illumination is removed by a homomorphic filter [12]. Segmentation is a crucial step and provides significant information about cancer such as border, shape, asymmetry, and irregularity [13]. Morphological filtering with weight-based features selection approach is used for the detection of lesion boundaries [14]. After features extraction, classification is done to discriminate the affected region into benign/malignant. The KNN, decision tree [15], and SVM [16] are used for classification. Deep learning methods [17-19] are mostly utilized for cancer detection [20]. Esteva et al developed GoogLeNet and Inception V3 CNN models for skin cancer classification. AlexNet [21, 25] model is applied to the dataset to learn the pattern of cancer. The extracted features pattern in the form of the vector is passed to the multiclass SVM for discrimination among the healthy and infected regions. A deep full resolution convolution network (DFRCN) with a SoftMax layer [27] is used for classification.

3. Materials and Methods

This section discusses materials and implementation used for cattle lumpy skin disease.

3.1. Proposed Methodology

Our proposed method consists of four major steps that are: (i) Stretching of disease-infected area (ROI), in this step we perform the segmentation of the infected region. We apply feature extraction from segmented regions once the affected region has been identified. The existence of too many infected regions is the main obstacle to accurate segmentation of the infected area. As a result, contrast stretching is crucial for quality improvement since it eliminates the effects of noise.

Extraction of important features for accurate classification is the second significant problem. As a result, in this research, we focus primarily on the following factor that impacts the outcomes: a) low contrast between infected and healthy regions; b) similarity of texture patterns between infected and healthy regions; c) dissimilarity of images caused by lighting and illuminating effects; d) use of unrelated features; and compatibility of chosen classifiers. In terms of accuracy, sensitivity, precision, and computational time, this will help us enhance disease detection and recognition effectively. (ii) Deep feature extraction, Efficiency of automation is overly dependent on feature sets. While weak and redundant features degrade system performance, strong and distinctive, features may improve the model performance. The so-called ABCD rule [22, 24, 28, 29] is the framework for feature extraction in our work. The ABCD represents the lumpy's asymmetry, border structure, color variation, and skin structure, and defines the basis for a diagnosis Veterinarian.

We employ two different kinds of features at this stage: global features and local features. The entire structure is represented by a single feature vector for the global feature. We examine a lumpy's size, symmetry, and color descriptors in global features. During the training phase of the local feature, images are sampled into small patches, and each patch is given a feature vector to represent it. We can characterize the many regions of the lumpy more precisely by breaking it up into smaller pieces. In a patch, less than 50% of the lumpy pixels are eliminated. (iii) feature fusion, in this step, several feature vectors are retrieved and combined into one feature vector. Then, the classification model is fed the obtained result. (iv) classification. Finally, we applied Extreme Learning Machine (ELM) to classify features. The detailed proposed flow is shown in **Figure 2**.

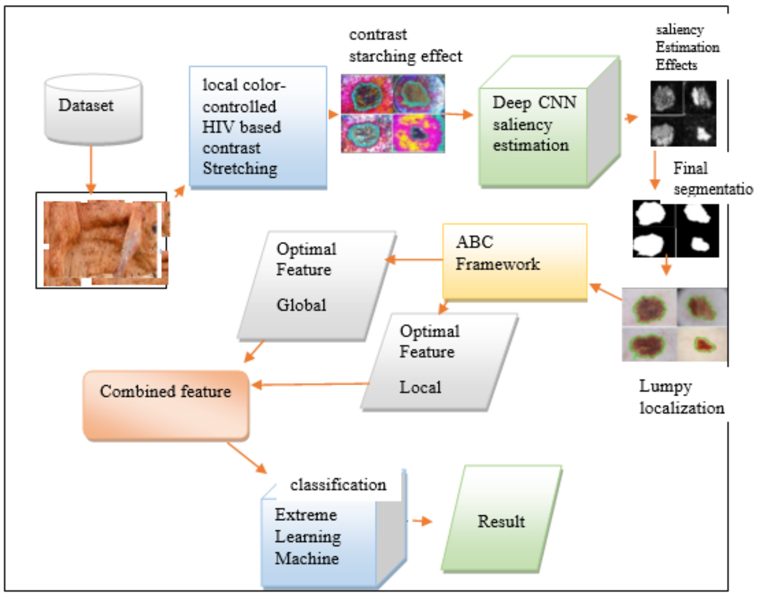


Figure 2. *Proposed systematic diagram of lumpy skin disease classification.*

3.2. Dataset

The proposed framework is validated using the CLSD dataset. There are 1100 image samples in this dataset.

Of these 800 images are for training, 200 images are for testing, and 100 images are for validation. Lumpy skin disease and non-lumpy skin disease classes are included in the training examples.

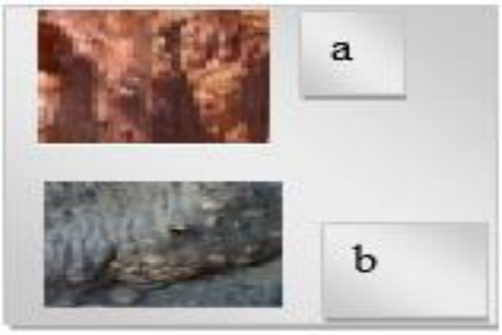


Figure 3. *(a) and(b) Lumpy affects various parts of the body.*

To enhance the size of the dataset, all the cattle bodies that can be affected by the disease are included. **Figures 3**, a, and b show some samples from the dataset, representing different body sections that can be afflicted by lumpy diseases, such as the ear, back, pin, tail, thigh, toe, stomach, elbow, chest, brisket, and neck. **Table 1** shows the lumpy affected body samples used in training.

Table 1. *Lumpy skin disease dataset detail*

Class	Ear	Back	pin	Tail	Tight	Toe	Stomach	elbow	chest	Brisket	Neck
Sample	60	54	60	60	55	50	60	50	41	50	60

During the training of the model on the datasets, there are three primary steps: lumpy detection, lumpy segmentation, and lumpy classification. Image quality is critical for detecting lumpy affected regions. Intensity enhancement is the most effective approach to improving image quality. Improving the quality of an image by enhancing a few features or lowering the amount of blockage between various image pixels. The main goal of this stage is to boost the contrast of the affected area so that the lumpy skin disease-affected region of interest (ROI) can be done easily. We build a histogram equalization (HE) and refine the findings to detect the bumpy pixels in the input image. The intensity values are then increased and changed according to the lumpy and background regions using a fitness function. A brief description of the procedure is as follows: The input image x, y, which has N x M dimension and where (x, y) εR. The histogram of the image is computed using equation 1.

$$h_f\,(k) = Oj \tag{1}$$

where hf(k) is the histogram of an image, f represents the frequency of occurrences, Oj represents the occurrence of gray levels, and j Ɛ = 0, 1, 2, . . . K - 1. Based on hf(k), we find

$$\sim h_f(k) = \quad h_f\,(k)[I_j]k1,\ kn, \tag{2}$$

the range of infected pixel, where I represent the infected region and j represents the pixel values. The ~hf (k) is the entire infected region, and the range of the infected region is represented by k1 to kn. Equation 3 is used to calculate the overall image Varian minuses

$$\sigma(\xi xy) = \frac{1}{MN}\sum_{i=0,j=0}^{N-1,M-1}(\xi ij)2 - \mu^2 \tag{3}$$

$$\mu = \sum_{i=0,j=0}^{M-1,N-1}(\xi ij)\ *\ \frac{1}{MN} \tag{4}$$

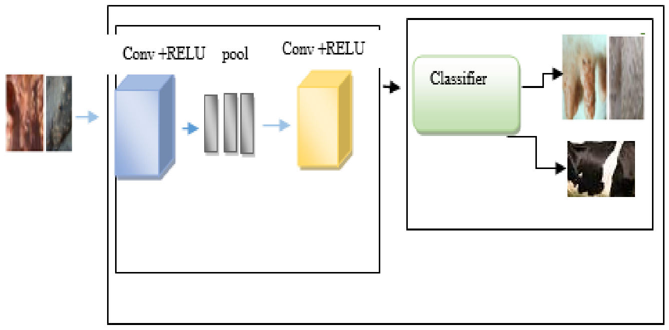


Figure 4. *Proposed model for lumpy skin disease*

The convolutional layer weight matrix and bias matrix are represented as follows

$$C = \sum \xi xy + w + b \tag{5}$$

where C denotes features of the first convolutional layer, x, y is an enhanced image, W is the weight matrix of the lth layer, and b is the bias matrix of the lth layer. After that, the ReLu activation layer was applied. In the second convolutional layer, the filter size was [3, 3], the number of channels was 64, the number of filters was 64, and the stride was [1, 1]. The features of this layer were normalized using the ReLu activation function. Next, a max-pooling layer was applied of filter size [2, 2] and stride of [2, 2]. The main purpose of this layer was to obtain more active features and minimize the feature-length. Equation 2 (see above) is multiplied by the output variance value obtained from this formula. Thereafter, we combine the results. In addition, the infected patch is subjected to histogram equalization before being fused with the original image.

Before being fused with the original image, the infected patch is also subjected to histogram equalization. Later, the image is loaded into (i) a ten-layer CNN model; (ii) features of the final convolutional layer are visualized and concatenated in one image; (iii) super pixels of the concatenated image are computed; (iv) a threshold is applied for final segmentation; and (v) boundaries are drawn on segmented regions using an active contour approach for the localization of lumpy affected areas.

We used these images to create a simple CNN model using output enhanced images with the size of 512 x 512 x3. This approach is mostly used to learn and visualize picture attributes. **Figure 4** depicts the designed model visually. One input layer, three convolutional layers, including the ReLu layer, one max-pool layer, one fully connected layer (FC), one SoftMax layer, and finally an output layer make up this model. We scaled all photos to 224 x 224 x3 because the input layer's size was 224 x 224 x 3. The filter size was [3, 3], the number of channels was 3, the filter size was 64, and the stride was [1, 1] in the first convolutional layer. We got two feature matrices after this layer: the weight matrix and the bias matrix. The weight matrix was 3 x 3 x 3 x 64 bytes in size, and the bias matrix was 1 x 1 x 64 bytes in size.

4. Experiment result

In this section, we discuss the steps and parameters that were employed during the computation of results. In the lumpy segmentation process, we tested the proposed model. During testing, the proposed model accuracy and error rate were both considered while segmenting lumpy. Multiple classifiers were employed in the classification process to compare the Extreme Learning Machine (ELM)'s effectiveness. Naive Baye, Multiclass Support Vector Machine and Fine K-Nearest Neighbor were among the classifiers used to validate the selected classifier.

We employed various classifiers (Naive Bayes, SVM, FKNN) to validate the performance of our model throughout the validation phase. In Naive Bayes, the classifier employs the Gaussian function. Later, we used a Multi-class Support Vector Machine, which combined the kernel function with one versus the rest. Nearest Neighbor was utilized in Fine KKN. In FKNN we computed Euclidean distance. We sated k to 10, and the learning rate to 0.001. we used mini-batch gradient descent where a mini-batch size is set to 28. The tensor flow was used as a simulation tool. The proposed segmentation result has an overall accuracy of 95.38 percent. The findings were obtained utilizing the proposed framework, and the values obtained are listed in **Tables 2** and 3. Here, the lumpy segmentation numerical results are presented in **Figure 5**.

Table 2. Classifier Performance Measures

Classifier (%)	Naïve Bayes	SVM	ELM	Fine KNN
Accuracy (%)	0.7624	0.7750	0.906	0.7494

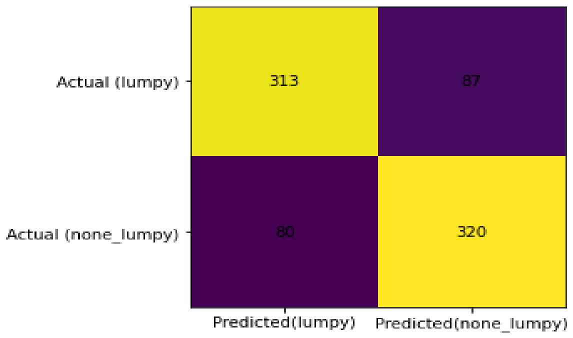


Figure 5. *Confusion matrix of the classifier model on lumpy skin disease dataset*

Table 3. *Confusion matrix of ELM classifier on cattle lumpy skin dataset*

Ear	0.90	0.00	0.01	0.02	0.01	0.01	0.02	0.01	0.02	0.01	0.02
Back	0.00	0.91	0.00	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.01
Pin	0.01	0.01	0.90	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
Tail	0.00	0.00	0.01	0.91	0.00	0.01	0.02	0.01	0.02	0.01	0.02
Thigh	0.00	0.00	0.02	0.01	0.90	0.00	0.01	0.00	0.01	0.00	0.01
Teo	0.01	0.00	0.00	0.00	0.01	0.91	0.00	0.01	0.00	0.01	0.00
Stomach	0.01	0.02	0.00	0.02	0.02	0.01	0.91	0.01	0.02	0.01	0.02
Elbow	0.01	0.01	0.00	0.01	0.01	0.01	0.02	0.90	0.01	0.00	0.01
Chest	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.89	0.01	0.00
Brisket	0.01	0.02	0.00	0.02	0.02	0.01	0.00	0.01	0.02	0.90	0.02
Neck	0.00	0.01	0.00	0.01	0.01	0.00	0.02	0.00	0.01	0.01	0.89
	Ear	Back	Pin	Tail	Thigh	Teo	Stomach	Elbow	Chest	Brisket	Neck

4.1. State-of-the-art Comparison

We also tested our dataset to compare our results to the current state of the art. The outcome is shown in below. **Table 4** describes the classification performance of ResNet101 deep features. The ResNet101 deep features were extracted. The result shows the best accuracy of 80.46%. MSVM gave the second-best accuracy of 77.50%; however, it is noted that only the prediction time for ResNet101 features increased. Similarly, the classification performance while using only DenseNet201 CNN deep features is given in **Table 5**. The best accuracy in this experiment was 79.34%, while the worst accuracy was 74.30%.

Table 4. *Lumpy skin classification results using only the Faster- RCNN model.*

Classifier	Accuracy (%)	FNR (%)	Prediction Time(s)
NaïveBayes	82.36	26.36	161.2031
ELM	89.42	23.76	139.9897
KELM	85.34	19.54	142.0120
XGBoost	89.00	19.50	143.00
MSVM	87.50	22.5	138.9210
Fine KNN	78.94	25.06	146.7980

Table 5. *Lumpy skin classification results using the DenseNet201 CNN model.*

Classifier	Accuracy (%)	FNR (%)	Prediction Time(s)
NaïveBayes	75.64	24.36	172.6420
ELM	89.24	19.68	140.9260
KELM	88.46	21.82	142.3364

MSVM	88.16	19.78	141.2064
XGBoost	89.00	19.50	141.00
Fine KNN	78.30	24.01	135.3092

In **Table 6** the ELM and SoftMax classifiers were evaluated. Based on the results, the ELM algorithm increases the classification accuracy. The best accuracy achieved in this experiment was 83.04% on the ELM classifier, whereas the worst accuracy was achieved by a Fine KNN of 76.04%. Additionally, the prediction time was minimized after this experiment due to the reduction in irrelevant features. The best time of this experiment was 96.3248 (s) on MSVM, whereas the ELM was executed in 103(s).

Table 6. *Comparison of the proposed model.*

optimization Technique	Accuracy (%)	Sensitivity (%)	Error (%)
ELM	90.50	89.98	9.5
Softmax	87.45	87.52	12.55

The classification results of the proposed system are presented in **Table 7**. A 50:50 strategy is utilized for recognition purposes. The ELM shows superior performance as compared to other classification methods and achieved an accuracy of 94.1%. The few other measures include sensitivity, specificity, precision, AUC, and FP rate 94.50%, 94.70%, 94.68%, 0.998, and 0.0020, respectively.

For this work, Extreme Gradient Boosting (XGBoost) was used due to its tendency to yield incredibly accurate findings. As a result, XGBoost is preferred over other traditional classifiers for enhancing classification quality. As can be seen in **Table 7**, where XGBoost scored the highest result, it is one of the most effective approaches for classifying and showed promising results over the dataset.

Table 7. *Proposed classification*

Method	Sensitivity (%)	Specificity (%)	Precision (%)	Accuracy (%)
C-SVM	88.89	89.98	989.51	89.08
C-KNN	87.80	88.00	88.10	87.82
Q-SVM	89.01	89.20	89.00	89.03
ESD	89.00	89.89	90.00	89.80
M-SVM	89.00	89.00	90	89.11
XGBoost	90.01	89.00	89.25	89.92
ELM	90.01	90.05	90.19	90.06

5. Conclusion

This article proposed a model for cattle's lumpy skin disease segmentation and classification. In the framework, a deep learning-based segmentation method and CNN feature optimization were described. The proposed method was evaluated on the well-known datasets for cattle's lumpy skin disease. The result shows the model performance is promising. The best classification result considered in this work is the ELM classifier having an accuracy of 0.9012. The ELM is found to be the overall best, having better performance on the dataset. Yet, one of our work's constraints is computational time, which will be investigated in the upcoming work. Additionally, in future studies, we will enhance our segmentation technique to prevent training our deep models on irrelevant visual features.

Declaration of interest

The authors declare that there is no conflict of interest.

Acknowledgements

Not Applicable

Nomenclature

CLSD	Cattle's lumpy Skin Disease
CNN	Convolutional Neural Network
DFRCN	Deep full resolution convolution network
ELM	Extreme learning machine
FC	Fully Connected Layer
HE	Histogram Equalization
KNN	K Nearest Neighbor
LCCHIV	local color-controlled histogram intensity values
RIO	Region of Interest
ReLu	Rectified Linear Units
SVM	Support Vector Machine
QSVM	Quantum-enhanced Support Vector Machine
XGBoost	Extreme Gradient Boosting

References

[1] Thornton PK. Livestock production: recent trends, future prospects." *The Royal Society*", 56 (2010) 121-128.

[2] Thomas M. "Bayesian latent modeling of Spatio-temporal variation in small-area health data". *Theory Into Practice,* 56 (2017) 121-128.

[3] Kang Y, Fang Y and Lai X, "Automatic detection of diabetic retinopathy with the statistical method and Bayesian classifier" *J. Med. Imag. Health Information,* 10(5) (2020) 1225–1233.

[4] Piekarski M, Jaworek-Korjakowska J, Wawrzyniak A.I, Gorgon M, "Convolutional neural network architecture for beam instabilities identification in Synchrotron Radiation Systems as an anomaly detection problem". *Measurement*, 165 (2020) 108116.

[5] Nisa M, Shah J.H, Kanwal S, Raza M, Khan M.A, Damaševiˇcius R, Blažauskas T. "Hybrid malware classification method using segmentation-based fractal texture analysis and deep convolution neural network features" *Appl. Sci*. 10(14) (2020) 4966.

[6] Wei Z, Song H, Chen L, Li Q, Han G. "Attention-based DenseUnet network with adversarial training for skin lesion segmentation". *in IEEE Access,* 7, (2019) 136616-136629; doi: 10.1109/ACCESS.2019.2940794.

[7] Tang J, Alelyani S, Liu H. "Feature selection for classification: A review. In Data Classification: Algorithms and Applications", *CRC Press: Boca Raton, FL*, USA, (2014) 37–64.

[8] Genemo, M.D. "Suspicious activity recognition for monitoring cheating in exams". *Proc.Indian Natl. Sci. Acad.* 88 (2022) 1–10.

[9] Farra D, Nardi MD, Lets V, Holopura S, Klymenok O, Stephan R, Boreiko O. "Qualitative assessment of the probability of introduction and onward transmission of lumpy skin disease in Ukraine", *Microbial Risk Analysis*, 20 (2022), 100200; https://doi.org/10.1016/j.mran.2021.100200.

[10] Vigier, M., Vigier, B., Andritsch, E. et al. Cancer classification using machine learning and HRV analysis: preliminary evidence from a pilot study. *Sci Rep* 11 (2021) 22292.

[11] Mehta P. and Shah B., "Review on techniques and steps of computer aided skin cancer diagnosis", *Procedia Comput. Sci., vol*. 85, pp. 309–316, Jan. 2016.

[12] Abbas Q, García I.F, and Rashid M. "Automatic skin tumour border detection for digital dermoscopy using a new digital image analysis scheme", *Brit. J. Biomed. Sci.,* 67(4), (2010) 177–183.

[13] Lee T.K "Measuring border irregularity and shape of cutaneous melanocytic lesions", Ph.D. *dissertation, Simon Fraser Univ., Burnaby*, BC, Canada, 2001.

[14] Schaefer G, Rajab M.I, Celebi M.E, and Iyatomi H, "Colour and contrast enhancement for improved skin lesion segmentation", *Computerized Med. Imag. Graph*., 35(2), (2011), 99–104.

[15] Murugan A, Nair SAH, and Kumar KPS, "Detection of skin cancer using SVM, random forest and kNN classifiers", *J. Med. Syst.,* 43(8), (2019), 269.

[16] Yuan X, Yang Z, Zouridakis G, and Mullani N, "SVM-based texture classification and application to early melanoma detection", *in Proc. Int. Conf. IEEE Eng. Med. Biol. Soc.,* (2006), pp. 4775–4778.

[17] Kang C, Yu X, Wang S.-H, Guttery D. S, Pandey H. M, Tian Y., and Zhang Y.-D, "A heuristic neural network structure relying on fuzzy logic for images scoring", *IEEE Trans. Fuzzy Syst. Leicester, U.K.: Univ. of Leicester, School of Informatics*, (2020), doi: 10.1109/TFUZZ.2020.2966163.

[18] Wang S, Sun J, Mehmood I, Pan C, Chen Y, and Zhang Y, "Cerebral micro-bleeding identification based on a nine-layer convolutional neural network with stochastic pooling", *Concurrency Comput., Pract. Exp.,* 32(1), (2020) p. e5130.

[19] Wang S, Tang C, Sun J, and Zhang Y, "Cerebral micro-bleeding detection based on densely connected neural network", *Frontiers Neurosci.,* 13, (2019), p. 422.

[20] Esteva A, Kuprel B, Novoa R.A, Ko J, Swetter S. M, Blau H. M, and Thrun S, "Dermatologist-level classification of skin cancer with deep neural networks", *Nature,* 542(7639), (2017), 115–118.

[21] Al-masni M. A, Al-antari M. A, Choi M.-T, Han S.-M, and Kim T.-S, “Skin lesion segmentation in dermoscopy images via deep full resolution convolutional networks”, *Comput. Methods Programs Biomed.*, 162, (2018), 221-231.

[22] Miglani V, Bhatia M. “Skin lesion classification: A transfer learning approach using efficientnets”, In Proceedings of the International Conference on Advanced Machine Learning Technologies and *Applications (AMLTA 2020), Jaipur, India*, 13–15 February 2020, 315–324.

[23] Nachbar F, Stolz W, Merkle T, Cognetta A.B, Vogt T, Landthaler M, Bilck P, Braun-Falco O, and Plewig G, “ The ABCD rule of dermatoscopy: High Prospective value in the diagnosis of doubtful melanocytic skin lesion”, *J.Amer.Acad.Dermatol*., 30(4), (1994), 551-559.

[24] Mahbod A, Schaefer G, Ellinger I, Ecker R, Pitiot A, Wang C. “Fusing fine-tuned deep features for skin lesion classification”, *Comput. Med. Imaging Graph,*71, (2019), 19–29.

[25] Garcia-Garcia A, Orts-Escolano S, Oprea S, Villena-Martinez V, Martinez-Gonzalez P and Garcia-Rodriguez J, “A survey on deep learning techniques for image and video semantic segmentation”, *Applied Soft Computing*, 70, (2018), 41-65.

[26] Zujovic J, Gandy L, Friedman S, Pardo B and Pappas T.N, “Classifying paintings by artistic genre: An analysis of features & classifiers”, *IEEE International Workshop on Multimedia Signal Processing,* IEEE, (2009), 1-5.

[27] Kobylin O.A, Gorokhovatskyi V.O, Tvoroshenko I.S, and Peredrii O.O, “The application of non-parametric statistics methods in image classifiers based on structural description components”, *Telecommunications and Radio Engineering*, 79(10), (2020), 855-863.

[28] Dredze M, Gevaryahu R and Elias-Bachrach A. “Learning fast classifiers for image spam”, International Conference on Email and Anti-Spam, (2007).

[29] Gorokhovatskyi V.O, Tvoroshenko I.S and Vlasenko N.V, “*Using fuzzy clustering in structural methods of image classification”, Telecommunications and Radio Engineering*, 79(9), (2020), 781-791.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 1), pp. 36-44, 2023
doi: 10.54569/aair.1200695
Published online: Feb 15, 2023

Prediction of Water Quality with Ensemble Learning Algorithms

Faten Aljarah [1*], Aydin Cetin [2]

[1] Graduate School of Informatics, Gazi University, Ankara, Türkiye
[2] Computer Engineering Department, Faculty of Technology, Gazi University, Ankara, Türkiye

Abstract

Monitoring and controlling the quality of the water is one of the most important issues in the world since only 74% of the world's population use safely managed water where the water is treated well to reach the minimum limit of safety and quality standards. To observe the water potability and take immediate actions to improve the water quality, real-time monitoring and classification process are required. However, monitoring and controlling the water quality is not an easy task since it has many requirements such as the collection and analysis of data and calculations to be made. In this paper, we focus on applying machine learning for the evaluation of the water quality. We have chosen five ensemble learning algorithms namely, Adaptive Boosting (AdaBoost), Random Forest (RF), Extremely randomized Trees (Extra Tree), Gradient Boosting (GB), and Stacking Classifier to evaluate their classification performances in determining the water quality. The Stacking Classifier had achieved the highest accuracy (0.67) and F-score (0.64).

Keywords: *Water Potability; artificial intelligent; production WQ; Machine Learning; ML*

1. Introduction

Human life, in general, depends on the availability of water, and the water quality is one of the important factors affecting the practical improvement of daily life. A specific standard of water quality needs to be reached in order for the water to be considered potable water, which is safe for humans to drink. In contrast, non-potable water is the water that is used for everything except human use. Many large-scale procedures are carried out on non-potable water before use, although it remains unfit for direct human consumption [1]. According to the World Health Organization, in 2020 more than 74% of the world's population (5.8 billion people) use safely managed water where the water is treated well to reach the minimum limit of safety and quality standards. However, more than 2 billion people in the World have access only to polluted or undrinkable water [2, 3]. For example, according to [4] it was reported that approximately 1.8 billion people worldwide use non-potable water sources. As a result, it affects the lives of people especially children, resulting in their death. According to a 2017 report from the World Health Organization, about 525,000 children under the age of five die from diarrhea every year [5]. The process of obtaining fresh water from ground and surface water in the past was easier than now. This is due to the increased dependence of human life on the availability of water. A rise in the issues of water pollution is also a result of the industrial and economic growth that humanity has reached, as well as a lack of knowledge about the right use of wastewater and adequate water use.

It is important to monitor water quality to find out the degree of water pollution, ensure access to clean water resources and apply effective guidelines for the protection of Water Resources [6]. Predicting water quality is a difficult task. Many researchers have made a great effort in determining water quality because of its importance to human life [2]. Therefore, it is an urgent necessity for humans to provide safe drinking water as it maintains the health of the kidneys, and the intestines nourish the muscles and help to maintain body fluid [7]. To ensure the potability of water, it is important to devise new technologies and methods. The water quality index (WQI) is a method used for measuring water quality which reflects the impact of different water standards on its quality. The calculation of WQI is necessary to determine the usability of water and to know the water specifications [8] It converts complex analyses of water properties and huge amounts of data into easy information that can be understood and used by specialists and non-specialists [9]. The water quality measurement index has been evaluated by many international studies as the basis for measuring various water indicators [10].

2. Background

A certain degree of water quality must be attained for water to be considered drinkable and safe for human consumption. In contrast, non-potable water is used for reasons other than drinking. The research [11] separated its data into 84% training and 16% tests and utilized the RF model and other models to estimate the quality of

*Corresponding author
E-mail address: fatina99aljarah@gmail.com

Received: 08/Nov/2022; Received in revised form: 07/Dec/2022; Accepted: 08/Dec/2022.

the water. It obtained favorable results after collecting data with Elsevier's Data in Brief (DiB). The KNN model was used in another study [12] that looked at the following parameters: dissolved oxygen (do), pH, conductivity, biological oxygen demand (bod), nitrates, fecal Escherichia coli, and total Escherichia coli form. This study used synthetically generated data. To achieve optimum water quality, the stacking model was employed by multiple researchers [13-15]. In other research, the KNN, RF, and Adaboost have been utilized and resulted in good evaluation [16, 17]. While GB has been compared with RF, KNN, ANN, and other models in Kelantan River, Malaysia [18].

The water quality index has been predicted in a study by [19] using neural networks at the Tigris River in Baghdad city. In many other studies, the importance of using the water quality index in measuring the potability of water on the Tigris River in the city of Mosul / Iraq is discussed. Also, the factors affecting the water quality index are indicated using the weighted mathematical model [20]. Other studies also aim to implement the Canadian water quality of Environment Ministers (CCME WAI) (CCME WAI) in the Tigris River [21, 22].
Recently, several research articles [23, 24]. have discussed the development of machine learning to assess water quality [25]. Studies have revealed various types of machine learning models applied to water quality, such as fuzzy logic, artificial neural networks, neural inference models, and others [26]. However, there are many variations of machine learning that have not yet been explored in water quality studies [24]. Although machine learning models are common in assessing water quality, they still face some shortcomings, such as the need for human intervention during the modeling process, time-consuming algorithms, and the need for flexible models in solving some environmental problems [27].

From the reviewed literature two major conclusions can be drawn. First, there have been many attempts to improve the performance of machine learning algorithms in water quality assessment. However, the current results still need to be improved. This study focuses on establishing an effective water quality assessment with ensemble methods. Secondly, most studies divide the parameters into chemical, biological, physical, and others for the application of one or more models to monitor the water and predict its quality, but the results are still insufficient. Consequently, this study is trying to fill this gap as well by exploring the ability of five models of machine learning (Adaptive Boosting, Random Forest, Extra trees classifier, Gradient Boosting, and Stacking Classifier) to predict water potability. This paper is organized as follows: After the introductory section, section 2 discusses the methodology and materials for the system that was investigated. Section 3 presents the evaluation metrics and the results of machine learning models considered for determining the potability of water samples and then concluding remarks are given in section 4.

3. Materials and Method

Through the water quality monitoring system and platform, one is able to determine whether the water is fit for a human drink or not. The modeling steps used in this study are presented as a flowchart in Figure 1, with each step being discussed.

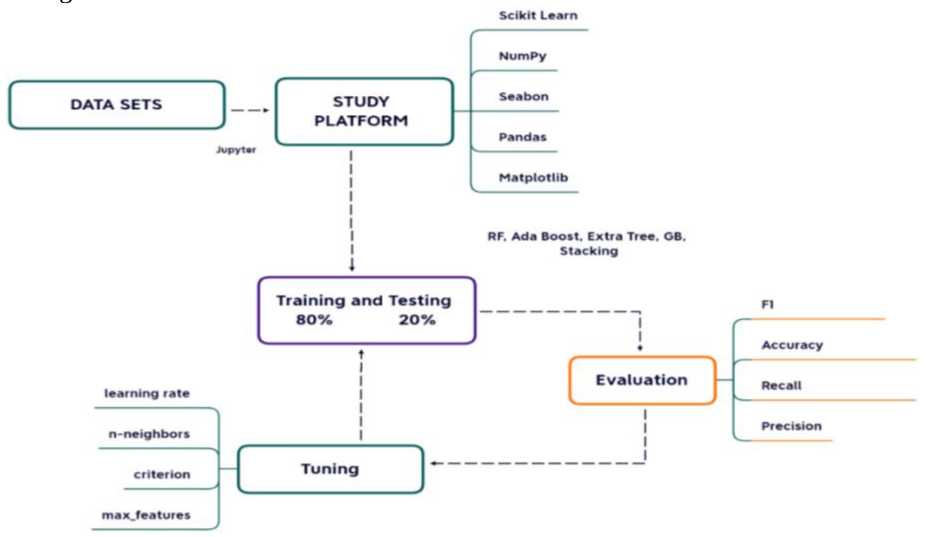


Figure 1. *Flowchart of the proposed model.*

3.1 Data set

The dataset was obtained from Kaggle in a data frame [28]. The information was collected two years ago and contains water quality measurements for 3276 different bodies of water. This dataset includes nine critical parameters: PH, Hardness, Solids, Chloramines, Sulfate, Conductivity, Organic carbon, Trihalomethanes, and Turbidity. Information was laid out in ten columns. This data was obtained from an industrial generator. It represents synthetic data containing information that is created artificially rather than via real-world happenings. Artificial data is made public in order to train machine learning algorithms and validate mathematical principles.

A number of operations and modifications have been performed in order to prepare the data and produce results with fewer errors. To achieve good predictive results, the dataset was divided into training and testing, an 80% training collection, and a 20% test collection. Since the data is not standardized, there is a gap between its values. Thus, the standard numerical is used to measure the data, which varies between 1 and -1 and used to standardize the data. Following a thorough examination of the data, it was discovered that it had empty values (Null) in data characteristics. The null was discovered in the PH, Sulfate, and Triethanolamine values, as illustrated in figure (2). To get rid of noise and unpredictable data, the null is eliminated and corrected using the average data rate.

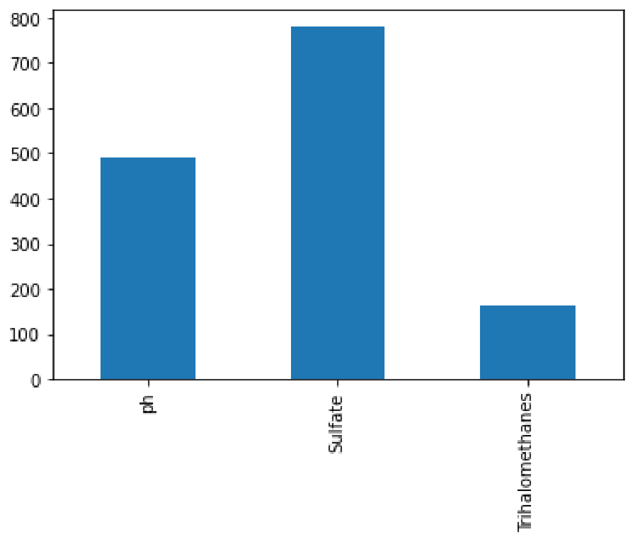


Figure 2. *The number of null in PH, sulfate, and triethanolamine.*

3.2 Development environment

To conduct all the necessary experiments, Jupyter notebook was used as an advanced work environment [29]. The project uses Python language, and as known, the Jupyter environment is one of the most compatible environments with Python.

Scikit_learn, NumPy, Pandas, Seaborn, Matplotlib libraries were used to develop the algorithm. The scikit_learn library includes a set of powerful data extraction and analysis capabilities. It is used to create a collection of machine learning, preprocessing, cross-validation, and visualization algorithms through a uniform interface. [30]. NumPy aims to supply many supporting functions. provides an array object up to 50 times faster than traditional Python lists [31]. Pandas library has been used because it helps to facilitate many time-consuming and repetitive tasks, including data normalization, data visualization, and others [32]. Seaborn library helps in understanding and exploring the data. It is mainly used for making statistical graphics in Python [33]. Matplotlib provides the user with the ability to visualize the data through a set of plots Such as scatter plots, graphs, etc. [34].

3.3 Parameter tuning

The model includes several hyperparameters (external parameters) that can optimize the model by changing its value manually. However, the learning algorithm itself cannot update or change hyperparameters [35]. In this research, we have adjusted the hyperparameters to get the best possible results. The main hyperparameters that contributed to the change in the results are:

- Max_depth: It is applied to determine the depth of the tree. It is the number of nodes from the root to the most distant leaf node. The greater the maximum depth value, the more complex the tree, and thus the greater the likelihood of overfitting. As a result, it is preferable to keep its value low [36]. Practical experiments show that as the maximum depth increases, the training error decreases, while the testing results in very poor accuracy. For example, when the depth is set to 50, the gradient algorithms accuracy is very poor. The best results are achieved by using the rest of the parameters with a depth value equal to 20.

- N_estimators: It is used to control the number of trees used in the ensemble model. This is the number of trees that must be constructed before averaging the final prediction [36]. The value of n-estimators should be set to 100 in order to achieve the greatest results in RF, Ada, Extra, and Gradient. While the algorithm bagging showed different results, the best results were between 220 and 230.
- N_neighbor: determines the number of data points by classifying them into groups. It represents the nearest number of data points that can be found and placed in the same category [36].. When k=1, a high training score shows, but the test result is quite low which results in overfitting. After multiple testing on n-neighbor based on the data of this research, it was discovered that when k=9 in the KNN model, the results are the best.
- L2_regularization: It controls the complexity to get rid of overfitting. This parameter is intended to limit the complexity, noise, and generalization issues that cause overfitting. Because complexity cannot be determined from training data, this parameter solves the complexity problem by determining the right level of complexity in the model [37]. After doing several trials, it was shown that the optimal amount of regulation in the Hist algorithm is 14.9.
- Loss: it is used in the boosting method. and it is a measure of the error that occurs between the output of the algorithm and the target function. This parameter calculates the probability of the expected positive class, to minimize losses [38]. It has been observed that the performance of the Hist model is better when its value is 'auto'. The 'exponential' value in the case of the gradient model is chosen for the best performance. The primary goal is to achieve a balance between under and over-fitting.
- Learning_rate: It can also be called shrinkage, and it works based on taking large steps but after several iterations, it takes smaller steps to reduce the error rate. To reach optimal results, different values of the learning rate have been experimented. The rate of learning is important because it contributes to determining the rates of weight change. It is also used to evaluate the splitting quality [38, 39]. It was found that the Hist and Adaboost algorithms performed better when the learning rate value was larger. On the other hand, the optimal learning value for GB is 0.1, which yields the best outcomes.
- Criterion: This parameter is used to measure the quality of the splitting, so it could stop the algorithm that is running [40]. It is seen that the result in RF and Extra tree is the best when the Criterion is set to “gini”.
- Max_features: It is used when searching for the best splitting, considering the number of features in the data. If this value is not specified, all features are permitted in each division [41]. When max features are set to ”auto”, the best results are obtained in some models including GB and DT. Nonetheless, the finest outcomes are frequently obtained when this variable is set to “sqrt” as in the Extra tree. Using this value, the model is instructed to choose a specified number of features at random. In this situation, the number of features equals the square root of the entire number of features in the dataset.
- Class_weight: It defines classifications by certain categories (0 and 1). This hyperparameter is used, for models of unbalanced classification [42]. The “balance” value produced the best results in the Extra model. During the typical training process, it modifies the balances of majority and minority group classes.

It is also worth noting that all these hyperparameters’ results change depending on each other. For example, the “gini” option may be the worst or the best depending on different parameters.

3.4 Machine Learning algorithms

Various learning algorithms are employed to generate artificial intelligence. Due to the numerous distinct types of algorithms, 5 ensemble learning methods have been discussed. while the models are constructed according to the data supplied and the type of application. The five models are implemented in this research, and the performance of each model is discussed below. To ensure the accuracy of the findings, 4 different performance metrics were used. Specifically, Precision (Precis), Recall (R), Accuracy (ACC), and F-score (F1 score).

Ada boosting: it mainly works on compiling multiple weak workbooks and gradually learning each of them from the previously wrongly classified objects. We made several manual changes to find out the best suitable parameters for the Ada boosting algorithm. It is noted that the best result is when the parameters of n_estimators= 100, the learning rate= 1.0, and the algorithm= SAMME.R. There is a clear change in the results when the parameters change, and the difference is noticeable.

Random forest: It is an algorithm used for classification, developed from a decision tree. Individual tree decisions are collected to create random forests. After doing many experiments, it is noted that the best results are when using n_estimators = 100 with criterion = gini.

Extra Tree: It is an algorithm that is very similar to the Random Forest algorithm. The difference is that it uses the entire sample instead of the reduction and substitution that the Random Forest algorithm does. It is noted from the extra tree experiment that the change of each of the parameters affects the whole results and the best result is when n_estimater= 100, criterion =gini, max_features= sqrt, and class_weight = balanced.

Gradient boosting: The gradient boosting model is done by building the new model on previous errors and predictions to check if there are any wrong patterns missed from the previous model. The best result is achieved with the parameters: 100, friedman_mse, 0.1, none, 20, exponential. When the parameters are 100, squared_error, 0.1, none, 20, deviance, results have become much worse.

Stacking: the method of action in stacking is to use a different set of models one after the other (sequentially), where the prediction of each of the models is added to produce a new feature. In the end, a final dataset is obtained (new feature), which is fed by the last model called meta-learner as shown in figure (3). The best result is when the meta = LR and the algorithms that are used in the stack are KNN, DT, RF, Adaboost, ET, GB, and Hist Gradient Boost (HGB).

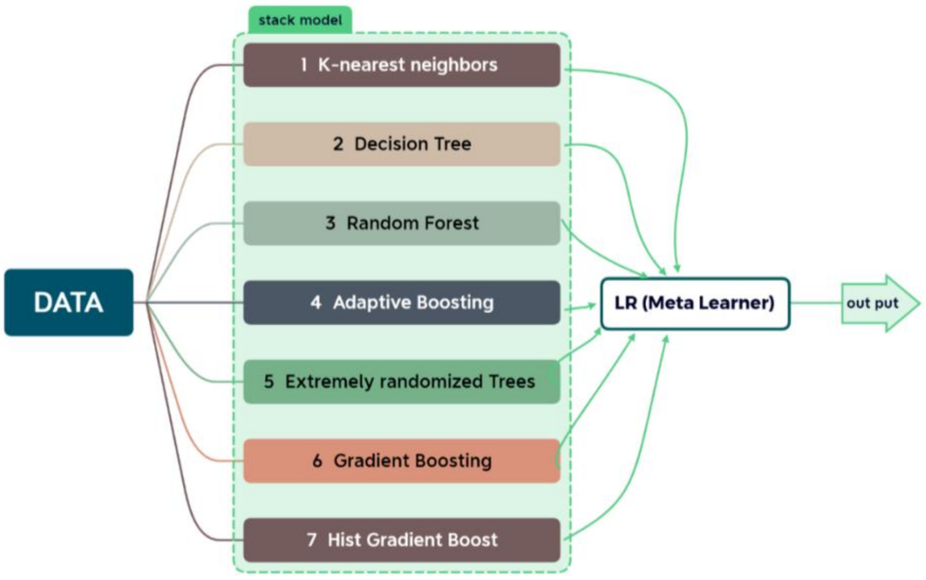


Fig 3: *Stacking in machine learning.*

3.5 Evaluation metrics

To determine whether the forecast results are positive or negative and evaluate the results of the models in predicting water quality, the research [43] used a set of rating performance measures. Confusion matrix was counted based on the next four settings:

1) TP, which means the number of ‘‘true positives’’ predictions.

2) FP, which means the number of ‘‘false positives’’ predictions.

3) FN, which means the number of ‘‘false negatives’’ predictions.

4) TN, which means the number of ‘‘true negative’’ predictions.

To separately examine the performance of each category of the model, the model performance has been analyzed into four categories so that it can be compared with other models [44].

a. **Accuracy** is one of the important and main tools in evaluating the model before practical application [45]. It is the arranger of the actual results among the total number of cases tested [46].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

b. **Recall** is the ratio of correct positives to all actual positives in the data that measures the test's ability to measure the state when the state exists [47].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

c. **Precision** is the ratio of the correct positive among all expected positives. It refers to how truthful the machine is about the real positives [48].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

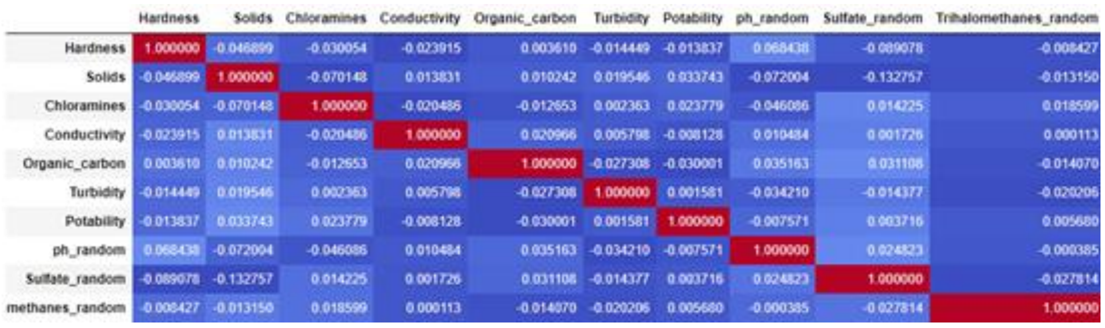
d. **F_score** is defined as an average (harmonic mean) between precision (P) and recall (R) [49] when there is a difference between FP and FN then F1_score will be needed. So, if you have various class distributions F1_score is the best [48].

$$\text{F\_score} = \frac{2\,\text{P}\,\text{R}}{\text{P} + \text{R}} \tag{4}$$

4. Experimental studies

After defining the algorithms to be trained according to the previous section under the methodology, we have performed parameter tuning and then evaluated the results according to evaluation metrics. In this study, the model performance results were evaluated and compared based on 4 criteria including, Precis, R, ACC, and F1 score. The choice of parameters affects obtaining more beneficial results and overall performance which may be more useful than the choice of the model itself. The performance of the model and the selection of parameters are evaluated according to the data set containing various data characteristics, including, PH, Hardness, Solids, Chloramines, Sulfate, Conductivity, Organic carbon, Trihalomethanes, and turbidity.

To apply the methodology, various models and templates were used with the help of Jupiter Notebook as a simple and fast working environment with the help of several important libraries in Python. One of the important steps that will be examined after data collection is the analysis of the relationship between the data. This is done through the correlation matrix, and as shown in Figure (4), there is no correlation between the data available to us, so we do not need to do any additional work to improve the results.



	Hardness	Solids	Chloramines	Conductivity	Organic_carbon	Turbidity	Potability	ph_random	Sulfate_random	Trihalomethanes_random
Hardness	1.000000	-0.046899	-0.030054	-0.023915	0.003610	-0.014449	-0.013837	0.068438	-0.089078	-0.008427
Solids	-0.046899	1.000000	-0.070148	0.013831	0.010242	0.019546	0.033743	-0.072004	-0.132757	-0.013150
Chloramines	-0.030054	-0.070148	1.000000	-0.020486	-0.012653	0.002363	0.023779	-0.046086	0.014225	0.018599
Conductivity	-0.023915	0.013831	-0.020486	1.000000	0.020966	0.005798	-0.008128	0.010484	0.001726	0.000113
Organic_carbon	0.003610	0.010242	-0.012653	0.020966	1.000000	-0.027308	-0.030001	0.035163	0.031108	-0.014070
Turbidity	-0.014449	0.019546	0.002363	0.005798	-0.027308	1.000000	0.001581	-0.034210	-0.014377	-0.020206
Potability	-0.013837	0.033743	0.023779	-0.008128	-0.030001	0.001581	1.000000	-0.007571	0.003716	0.005680
ph_random	0.068438	-0.072004	-0.046086	0.010484	0.035163	-0.034210	-0.007571	1.000000	0.024823	-0.000385
Sulfate_random	-0.089078	-0.132757	0.014225	0.001726	0.031108	-0.014377	0.003716	0.024823	1.000000	-0.027814
methanes_random	-0.008427	-0.013150	0.018599	0.000113	-0.014070	-0.020206	0.005680	-0.000385	-0.027814	1.000000

Figure 4. *The Correlation matrix of water potability data parameter.*

Machine learning models are used to predict whether water is safe or unsafe for consumption. This section describes the model's execution of the Adaboost, RF, Extra Tree, GB, and Stacking. Figure (1) shows the step-by-step execution of the models to achieve the desired results. It can be said that each test presented a different method for estimating water potability. Changing the parameters played an important role in improving the results of the models in general. It is noted that the stacking model performed so well, having the best results in F1 Score for estimating potability of water. It is important to mention that the "F1 Score" and "ACC" have importance to demonstrate the credibility of the results. Therefore, these two are given priority since the F1 Score represents the Precis and the R together. When classification ACC is compared, the performance of Random Forest (0.66) Extra Tree (0.66) Gradient Boosting (0.66), and Stacking (0.67) is very close and better than Adaboost (0.61). ACC allows the decision maker to understand the level of accuracy since accuracy is a numerical estimation of the performance of the model. The metrics for WQ predictive models according to ACC and F1 Score are given in detail in Table 1.

Table 1. *Performance metrics for WQ predictive models.*

Models	Accuracy	F_score
Ada boosting	0.61	0.55
Random forest	0.66	0.62
Gradient Boosting	0.66	0.62
Stacking	0.67	0.64
Extra Tree	0.66	0.63

The correlation matrix displays all the levels in the dataset. Since the ACC and F1 Score were discussed, the remainder is the R and Precis. Table (2) shows that the Extra tree represents the best Precis among the remaining models with a value of 0.69.

Table 2. *Performance metrics for WQ predictive models according to recall and precision.*

Models	Precision	Recall
Ada boosting	0.54	0.57
Stacking	0.61	0.69
Gradient Boosting	0.61	0.64
Extra Tree	0.63	0.64
Random forest	0.66	0.60

There have been several research in machine language to analyze water quality. As a result of one of the studies based on the same data that we are working on in the research [50], the model's findings were DT = 0.61 and RF=0.69, It is extremely close to our results. However, the Stacking Classifier model outperforms the other models according to our findings. Other studies [11], using different data and settings, achieved an accuracy of RF= 0.96.

5. Conclusions

Classification prediction methods were used in this study (i.e., Adaptive Boosting, Random Forest, Extra Trees classifier, Gradient Boosting, and Stacking Classifier) to predict water potability. The performances of these five models have been compared using the confusion matrix which includes (TP, FP, TN, and FN) with cross-validation to find the reliability of the models. Our dataset shows recorded values of 9 water parameters such as PH, Hardness, Solids, and turbidity. To eliminate noise and unstable data, null is eliminated and compensated with the average data rate. The parameters were changed continuously, and there was a noticeable difference in the results. From the results of the models obtained, it was observed that the Stacking C lassifier achieved better performance than the other models we have tested in estimating water quality. It is believed that this study can help researchers develop integrated artificial intelligence and machine learning models that will help water system managers in real-time monitoring of the quality of water for future applications.

References

[1] Varila M., “What Is Potable Water? Your Guide to Understanding Types of Water”, viralrang, 2020. [Online]. Available: https://viralrang.com/what-is-potable-water-your-guide-to-understanding-types-of-water/#. [Accessed: Nov 8, 2022]

[2] UNECE, “miyah alshrob,” who, (2022). [Online]. Available: https://www.who.int/ar/news-room/fact-sheets/detail/drinking-water. [Accessed: Oct 19, 2022].

[3] Fluence news team, “What Is Potable Water?”, fluencecorp, 2019. [Online]. Available: https://tinyurl.com/2qj936u9. [Accessed: Nov 8, 2022].

[4] World Health Organization, “Preventing diarrhoea through better water, sanitation and hygiene: exposures and impacts in low- and middle-income countries,” World Health Organization (Report), Villars-sous-Yens, Switzerland, 2014.

[5] World Health Organization, “Diarrhoeal disease,” who, 2017. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/diarrhoeal-disease. [Accessed: Dec 3, 2022].

[6] Li D., Liu S., “System and Platform for Water Quality Monitoring Chapter 3,” in Water Quality Monitoring and Management, *China: Academic Press*, 2019, p. 101.

[7] Edition F., Guidelines for Drinking-water Quality - 4th ED., Malta: World Health Organization WHO Library Cataloguing, 2011.

[8] Al safaw Y.A., R. Al Shanouna R.A.A., Messer, N., "Takeem hasaas naweet almeah w hesab muamel WQI le baaz masader almeah fe karyat abo marya kazaa talefar\ muhafazat nainawa," *Journal of Education and Science*, 27(3), 87, 2018.

[9] Al Safawi A. Y. T., "Tatbik almuasher alkndy (WQI CCME) le takeem jawdet almeyah le agrad alshrub: dirasat halat jawdet almeyah aljawfeia fe nahiat almehalabia\ muhafazat nainawe," *Journal of Rafidain Sciences*, 27(4), 199, 2018.

[10] Dilip P.V., Dnyaneshwar, M. S., Rajendra, L. D., Suresh, N. P., "Assessment of Ground Water Quality In Gajanan Colony, Ahmednagar. By Water Quality Index (WQI)," in Second Shri Chhatrapati Shivaji Maharaj QIP Conference on Engineering Innovations, Ahmednagar, India, 105, 2019, ISSN: 2581- 4230.

[11] Ajayi O.O, Bagula A.B, Maluleke H.C., "Water Net: A Network for Monitoring and Assessing Water Quality for Drinking and Irrigation Purposes", *IEEE Access*, 10, 48318- 48337. 2022, doi: 10.1109/ACCESS.2022.3172274, 2022.

[12] Aldhyani T.H.H., Al-Yaari M., Al kahtani H., "Water Quality Prediction Using Artificial Intelligence Algorithms," *Applied Bionics and Biomechanics*, vol.2020, 1-10. doi: 10.1155/2020/6659314, 2020.

[13] Nasir N, Kansal A, Aishalton O, "Water quality classification using machine learning algorithms", *Journal of Water Process Engineering*, vol.48. doi: 10.1016/j.jwpe.2022.102920, 2022.

[14] Wang L, Zhu Z, Sassoubre L, "improving the robustness of beach water quality modeling using an ensemble machine learning approach", *Science of the Total Environment*, 765, 1-4, doi: 10.1016/j.scitotenv.2020.142760, 2021.

[15] Rosly R, Makhtar M, Awang M.K, "Comparison of Ensemble Classifiers for Water Quality Dataset," in Proceedings of the UniSZA Research Conference 2015 (URC '15), Terengganu, Malaysia, 1-6, 2015

[16] Mogaraju J.K, "Application of machine learning algorithms in the investigation of groundwater quality parameters over YSR district, India," *Turkish Journal of Engineering*, 7(1), 64 - 72. doi: 10.31127/tuje.1032314, 2023.

[17] El Bilali A, Taleb A, Brouziyne Y, "Groundwater quality forecasting using machine learning algorithms for irrigation purposes", *Agricultural Water Management*, 245, 106625. doi: 10.1016/j.agwat.2020.106625 , 2021.

[18] Abdul Malek N.H, Wan Yaacob W.F, Md nasir S.A, "Prediction of Water Quality Classification of the Kelantan River Basin, Malaysia, Using Machine Learning Techniques", *Water*, 14(7), 1067. doi: 10.3390/w14071067, 2022

[19] Al-Musawi N, "Prediction and Assessment of Water Quality Index Using Neural Network Model and Gis Case Study: Tigris River in Baghdad City", *Applied Research Journal*, 3(11), 343-353, 2018.

[20] Talat R.A, Al-Assaf A.Y, Al-Saffawi A.Y.T, "Valuation of water quality for drinking and domestic purposes using WQI: Case study for groundwater of Al-Jameaa and Al-Zeraee quarters in Mosul city/Iraq", Journal of Physics Conference Series, 1294(7). doi: 10.1088/1742-6596/1294/7/072011, 2019

[21] Safawi A.Y.T.A, "tatbiq al muasher al kanadi(WQI CCME) le taqeem javed almeyah le agrade alshrub", in The third Scientific Conference of life sciences, Iraq, 27(5), 193-202, 2019.

[22] Mahmood A, "Evaluation of raw water quality in Wassit governorate by Canadian water quality index", in *Environmental Engineering and Sustainable Development*, Iraq, 162, 1-8. 2018, doi: 10.1051/matecconf/201816205020.

[23] Mosavi A, Ozturk P, Chau K, "Flood Prediction Using Machine Learning Models: Literature Review", *Water*, 10(11), 1536. doi: 10.3390/w10111536, 2018.

[24] Chen Y, Song L, Liu Y, "A Review of the Artificial Neural Network Models for Water Quality Prediction," *Applied Sciences*, 10(17), 5776. doi: 10.3390/app10175776, 20 8 2020.

[25] Koranga M., Pant P, Pant D, "SVM Model to Predict the Water Quality Based on Physicochemical Parameters," *International Journal of Mathematical, Engineering and Management Sciences*, 6(2), 645-659. doi: 10.33889/IJMEMS.2021.6.2.040, 2021

[26] Al-Adhaileh M. H, Alsaade F. W, "Modelling and Prediction of Water Quality by Using Artificial Intelligence," *Sustainability*, 13(8), 4259. doi: 10.3390/su13084259, 2021

[27] Park S, Jung S, Lee H, "Large-Scale Water Quality Prediction Using Federated Sensing," *Sensors*, 21(4), 1462. doi: 10.3390/s21041462, 2021.

[28] Kadiwal A., "Water Quality, Drinking water potability," Kaggle, 2019. [Online]. Available: https://www.kaggle.com/datasets/adityakadiwal/water-potability. [Accessed: March 9, 2022].

[29] Pérez F, Granger B, "jupytercon," jupyter, 2014. [Online]. Available: https://jupyter.org/. [Accessed: March 5, 2022].

[30] Scikit-learn authors, "1. Supervised learning," scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/supervised_learning.html#supervised-learning. [Accessed: April 10, 2022].

[31] Developers, "NumPy 1.23.0 released," numpy, 2022. [Online]. Available: https://numpy.org. [Accessed: April 10, 2022].

[32] Developers, "pandas: powerful Python data analysis toolkit," pypi, 2022. [Online]. Available: https://pypi.org/project/pandas/. [Accessed: April 10, 2022].

[33] Developers, "seaborn: statistical data visualization," seaborn, 2021. [Online]. Available: https://seaborn.pydata.org/. [Accessed: April 10, 2022].

[34] Developers, "Matplotlib: Visualization with Python," matplotlib, 2022. [Online]. Available: https://matplotlib.org/. [Accessed: April 10, 2022].

[35] Brownlee, J., "What is the Difference Between a Parameter and a Hyperparameter?," machine learning mastery,

26 6 2017. [Online]. Available: https://machinelearningmastery.com/difference-between-a-parameter-and-a-hyperparameter/. [Accessed; June 10, 2022].

[36] Yıldırım S, "6 Must-Know Parameters for Machine Learning Algorithms," towards data science, 2022. [Online]. Available: https://towardsdatascience.com/6-must-know-parameters-for-machine-learning-algorithms-ed52964bd7a9. [Accessed: June 10, 2022].

[37] Yıldırım S, "L1 and L2 Regularization — Explained," towardsdatascience, 2020. [Online]. Available: https://towardsdatascience.com/l1-and-l2-regularization-explained-874c3b03f668. [Accessed: June 10, 2022].

[38] Developers, "sklearn.ensemble.HistGradientBoostingClassifier," scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingClassifier.html. [Accessed: June 11, 2022].

[39] DigitalSreeni, Director, 184 - Scheduling learning rate in keras. [Video]. United States: Site: YouTube, 2020. URL: https://youtu.be/drcagR2zNpw.

[40] Developers, "Sklearn.tree.DecisionTreeClassifier," scikit-learn, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html [Accessed: June 11, 2022].

[41] Bhatt B, Director, Decision Tree Hyperparameters : max_depth, min_samples_split, min_samples_leaf, max_features. [Video]. India: Site: YouTube, 2019. URL: https://www.youtube.com/watch?v=XABw4Y3GBR4&t=365s.

[42] Paper D, "Scikit-Learn Classifier Tuning from Complex Training Sets," in Hands-on Scikit-Learn for Machine Learning Applications, Logan, UT, USA, Apress, Berkeley, CA, 2020. doi: 10.1007/978-1-4842-5373-1_6.

[43] Alwanas A.A.H, Al-Musawi A.A, Salih S.Q, "Load-carrying capacity and mode failure simulation of beam-column joint connection: Application of self-tuning machine learning model," *Engineering Structures*, 194, 220-229. doi: c10.1016/j.engstruct.2019.05.048, 2019.

[44] Tung T. M, Yaseen Z. M, "A survey on river water quality modelling using artificial intelligence models: 2000--2020", *Journal of Hydrology*, vol. 585, 124670. doi: 10.1016/j.jhydrol.2020.124670, 2020.

[45] QI C, Huang S, Wang X, "Monitoring Water Quality Parameters of Taihu Lake Based on Remote Sensing Images and LSTM-RNN," *IEEE Access*, vol. 8, 188070. doi: 10.1109/ACCESS.2020.3030878, 2020.

[46] Soumik S.K, "How to Calculate Confusion Matrix Manually.", medium, (2020). [Online]. Available: https://medium.com/analytics-vidhya/how-to-calculate-confusion-matrix-manually-14292c802f52. [Accessed: June 22, 2022].

[47] Ho J.Y, Afana H.A, El-Shafie A.H, "Towards a time and cost-effective approach to water quality index class," *Journal of Hydrology*, vol. 575, 148-165. doi: 10.1016/j.jhydrol.2019.05.016, 2019.

[48] Atha R, "Building Classification Model with Python," medium, (2021). [Online]. Available: https://medium.com/analytics-vidhya/building-classification-model-with-python-9bdfc13faa4b. [Accessed: June 22, 2022].

[49] Sasaki Y., "The truth of the F-measure," School of Computer Science, University of Manchester, 2007.

[50] Wiryaseputra M, "Water Quality Prediction Using Machine Learning Classification Algorithm", *International Journal of Scientific & Engineering Research*, 8(9). doi: 10.14299/000000, 2022.