

FROM ARTIFICIAL INTELLIGENCE TO INTELLIGENT SYSTEMS:

COMPUTATIONAL METHODS AND EMERGING APPLICATIONS

Editor:
Prof. Dr. Eyyüp GÜLBANDILAR



BİDGE Yayınları

**From Artificial Intelligence To Intelligent Systems:
Computational Methods and Emerging Applications**

Editor: Prof Dr. Eyyüp GÜLBANDILAR

ISBN: ISBN EKLENECEK

1st Edition

Page Layout By: Gözde YÜCEL

Publication Date: 2026-10-25

BİDGE Yayınları

All rights reserved. No part of this work may be reproduced in any form or by any means, except for brief quotations for promotional purposes with proper source attribution, without the written permission of the publisher and the editor.

Certificate No: 71374

All rights reserved © BİDGE Yayınları

www.bidgeyayinlari.com.tr - bidgeyayinlari@gmail.com

Krc Bilişim Ticaret ve Organizasyon Ltd. Şti.

Güzeltepe Mahallesi Abidin Daver Sokak Sefer Apartmanı No: 7/9 Çankaya /
Ankara



PREFACE

Artificial intelligence has become one of the most significant drivers of transformation in information technologies, leading to profound changes in scientific research, engineering, and many aspects of everyday life. In particular, advances in machine learning, deep learning, image processing, and data analytics have enabled the extraction of meaningful patterns from large and complex datasets, the support of prediction and decision-making processes, and the development of intelligent systems for a wide range of applications. However, the rapid evolution of artificial intelligence brings not only the need to develop more accurate models but also the challenge of ensuring that these models can be deployed reliably, efficiently, securely, and at scale.

This book has been prepared to bring together current perspectives on artificial intelligence and computational methods across diverse application domains. Although the chapters focus on different methodologies and fields of application, they share a common objective: transforming data into meaningful information, evaluating that information through analytical methods, and developing reliable and intelligent systems.

The deployment of artificial intelligence models on embedded and resource-constrained platforms has become an increasingly important area of research. In this context, the relevant chapter examines deep learning in embedded systems from the perspectives of hardware platforms, model adaptation, and performance evaluation. These aspects are particularly important for transferring artificial intelligence models from conventional computing environments to real-time systems with limited computational and energy resources.

In the field of image processing, the book addresses local binary patterns and related approaches, focusing on their theoretical foundations, variants, and recent developments. Such methods play an important role in extracting local structural characteristics from images and continue to contribute to the development of machine learning- and deep learning-based image analysis systems. In addition, the analytical foundations of regression and its application to stock market data prediction demonstrate how computational methods can be employed in financial data analysis and forecasting.

Cybersecurity constitutes another major theme of this book. As digital systems become increasingly interconnected, cyber threats are becoming more diverse, sophisticated, and difficult to detect using conventional security mechanisms alone. Consequently, artificial intelligence and computational approaches have gained increasing importance in the development of modern cybersecurity solutions. In this book, topics such as automated cyberbullying detection, graph-based cyber-attack detection, data representation, topological analysis, and computational efficiency are examined from this perspective.

Cloud security represents another contemporary area addressed in the book. Zero Trust architecture and artificial intelligence-driven identity protection constitute important components of a modern security paradigm in which users, devices, applications, and access requests are continuously evaluated rather than automatically trusted. The integration of artificial intelligence into identity verification, behavioral analysis, and threat detection processes provides opportunities for developing more dynamic, adaptive, and context-aware security mechanisms.

A central perspective throughout the book is that artificial intelligence methods should not be evaluated solely on the basis of prediction accuracy or classification performance. The practical usability of an intelligent model depends on a broader set of factors, including computational cost, data representation, model complexity, reliability, scalability, security, and the characteristics of the deployment environment. Accordingly, the chapters consider not only the theoretical and algorithmic aspects of the methods discussed but also their practical implications and performance considerations.

This book is intended for academics, researchers, graduate students, and practitioners working in computer engineering, artificial intelligence, data science, image processing, cybersecurity, and related disciplines. It is also designed for readers who wish to understand how artificial intelligence and computational methods can be adapted to different real-world problems and how seemingly distinct research areas can converge through data-driven and intelligent approaches.

The rapid pace of development in artificial intelligence means that methods considered state-of-the-art today may soon be complemented or replaced by new approaches. Therefore, this book is intended not merely to present existing techniques but also to encourage readers to formulate new research questions, explore emerging technologies, and establish connections between different disciplines.

I would like to express my sincere gratitude to all chapter authors who contributed their knowledge and expertise to this work, to the reviewers who provided valuable academic feedback, and to everyone who supported the preparation and completion of this book.

I hope that this book will contribute to the understanding and advancement of artificial intelligence and computational methods, inspire new research, and facilitate the transfer of scientific knowledge into diverse application domains.

Prof. Dr. Eyyüp GÜLBANDILAR
Editor

CONTENTS

AN ALGORITHM FOR AUTOMATIC DICTIONARY USING THE WEB	1
--	---

AHMET TOPRAK, METİN TURAN

MACHINE LEARNING-BASED SENTIMENT ANALYSIS OF CONVERSATIONAL AI IN TURKISH MICROBLOGS ..	35
--	----

AYHAN AYDIN

THE IMPACT OF TECHNOPARKS ON FIRM PERFORMANCE: AN EMPIRICAL INVESTIGATION INTO PRODUCTIVITY, INNOVATION, RECYCLING, EMPLOYMENT, AND EXPORTS	57
--	----

AYHAN AYDIN

MACHINE LEARNING IN RESOURCE CONSTRAINED EMBEDDED SYSTEMS	76
--	----

MEHMET BİLGE HAN TAŞ

EXPLAINABLE ARTIFICIAL INTELLIGENCE: INTERPRETING BLACK-BOX MODELS AND CURRENT APPROACHES	100
---	-----

İSMAİL AKGÜL

IDENTIFYING SIMILARITIES BETWEEN PROPOSED AND EXISTING COURSES IN GRADUATE PROGRAMMES USING NATURAL LANGUAGE PROCESSING	125
--	-----

MURAT İNCE, EBRU İNCE

DIGITAL PHENOTYPING OF MIGRAINE: PERSONALIZED MONITORING, RISK MODELING, AND DECISION SUPPORT WITH ARTIFICIAL INTELLIGENCE AND ENGINEERING APPROACHES	141
--	-----

ŞULE YÜCELBAŞ, CÜNEYT YÜCELBAŞ

CONTENTS

CONTINUAL LEARNING FOR GEARBOX FAULT DIAGNOSIS UNDER VARYING LOAD CONDITIONS: A COMPARATIVE STUDY OF FINE-TUNING, EXPERIENCE REPLAY, AND KNOWLEDGE DISTILLATION	161
---	-----

CANAN TASTİMUR

AUTOMATED DETECTION OF PREDATORY JOURNALS: A SURVEY OF COMPUTATIONAL APPROACHES AND RELIABILITY CHALLENGES	176
--	-----

DİDEM YANGEÇ

ZERO TRUST CLOUD SECURITY AND ARTIFICIAL INTELLIGENCE DRIVEN IDENTITY PROTECTION	226
---	-----

ÖMER SEVİNÇ

GRAPH-BASED CYBER-ATTACK DETECTION: REPRESENTATION STRATEGIES, TOPOLOGICAL ANALYSIS, AND COMPUTATIONAL EFFICIENCY	234
---	-----

ALİ MELİH KANCA, İLKER TÜRKER

ADVERSARIAL ARTIFICIAL INTELLIGENCE AND CROSS SECTOR CYBERSECURITY GOVERNANCE	278
--	-----

ÖMER SEVİNÇ

CHAPTER 1

An Algorithm for Automatic Dictionary Using the Web

AHMET TOPRAK¹
METİN TURAN²

1. Introduction

There has been an increasing need to extract information from big data lately. Domain-specific dictionaries are needed to manage this task. Therefore, studies in language lexicography have been focusing on automatic dictionary creation. These dictionaries can be used for search engine optimization [39], automatic summarizing systems [19, 15], theme determination [10] and text classification [28] researches.

The dictionary creation process can be done manually [24], semi-automatically [25], or automatically [34]. Hand-created dictionaries are static, and they require continuous external support to increase the size of the dictionary, resulting in higher maintenance costs. Riloff [34] described the shortcomings of manual dictionaries.

¹ Asst. Prof., İstanbul Gelisim University, Computer Engineering, Orcid: 0000-0001-7046-8512

² Assoc. Prof., İstanbul Ticaret University, Computer Engineering, Orcid: 0000-0002-1941-6693

Knowledge-based natural language processing (NLP) [27] systems are often criticized although they have achieved success with specific tasks because they are dependent on a domain-specific dictionary that requires manual engineering knowledge. Because these systems cannot be easily scaled or transferred to the new fields.

In this study, an algorithm was proposed and experimentally tested to create an automatic dictionary for a specific domain. The initial step is to find meaningful words from the document (or set of documents) given by the user. This document is called “reference document”. The most meaningful words in the reference document were identified using term frequency-inverse document (TF-IDF) values. The first dictionary words, we call it “seed words”, are obtained from the most meaningful words of the reference document. A web search is then applied using the Azure Web Cognitive Search Interface repetitively in order to execute a new query and find new related words within returned documents. These meaningful words found at each iteration aren’t added directly to the dictionary. Instead, the WordNet dictionary is used to evaluate the similarity of each meaningful word within the dictionary so that adding meaningless words to the dictionary is prevented. The words with higher similarity values above a certain threshold value (experimentally learned) are added to the dictionary until the desired number of words in the dictionary is accumulated. Hence, the dictionary continues growing automatically and continuously without any intervention from human being.

The hash similarity metric was used to evaluate the overall similarity of the dictionary. The notion of locality-sensitive hashing was first introduced by Indyk et al. [20]. For efficient near-neighbor search, the locality-sensitive hashing (LSH) technique exploits special hash functions that make buckets contain similar keys (data), yet it does not guarantee that all data in a bucket are similar to each other. The so-called locality-sensitive hash functions provide a high

probability for similar data to be in the same bucket but a low probability for dissimilar data to be in the same bucket [26].

The following section includes prior works in the field of dictionary generation. The methods and studies used for dictionary creation are explained in the third section. The tests applied within the research are explained in the fourth section. Finally, the results are discussed and evaluated in the last section.

2. Literature Review

TF-IDF has been successfully applied to many different NLP applications. One of these important studies was by Paik [29], who performed an effective ranking study in 2013 using TF-IDF. In this particular study, Paik addressed the limitations of the pivoted length normalization by exploiting new statistical factors in the Multi-Aspect TF (MATF) schema. One component of the term frequency is effective for short queries, while the other performs better on long queries. The final weight is then measured by taking a weighted combination of these components, which is determined based on the length of the corresponding query. Experiments conducted on a large number of TREC news and web collections demonstrate that the proposed scheme almost always outperforms five times the state-of-the-art information retrieval models in a remarkably significant and consistent manner.

In another TF-IDF term weighting study [7], a short message service (SMS) model was developed to determine the statistical importance of a word for an SMS categorization. First, all SMSs are converted to text documents. After the pre-processing of the SMSs, a vector space model is prepared, and a weight is given to each term. This weighting method predicts the significance of a word for an SMS classification problem. Experiments reported in the article showed that this weighting method significantly improved classification accuracy.

TF-IDF was also used in sentiment analysis studies [11]. Das and his colleagues proposed text sentiment classification using TF-IDF in conjunction with Next Word Negation (NWN). They also compared the performances of a binary bag-of-words model, the TF-IDF, with TF-IDF with “next word negation” (TF-IDF-NWN) model for text classification. The results showed significant increases in accuracy compared to earlier methods.

In 2018, Kaiser et al. [33] used TF-IDF for the identification of keywords in the document and pointed out the shortcomings of this method because this method yielded unexpected results. Although the words (“go”, “goes”), (“play”, “playing”), (“mark”, “marking”) and (“year”, “years”) in the documents have the same word root, these words were perceived as words that altered the results. The method was also criticized for the word, coexistence, and for the control of the document in terms of the dictionary rather than meaning. It has been noted that TD-IDF can be combined with other techniques, such as Naive Bayes, to yield better results.

As stated in the previous section, the dictionary can be created in three different ways: automatic, semi-automatic, and manual. In 1993, Riloff [34] conducted a study on automatic dictionary creation. In this work, Riloff developed a system called AutoSlog, a dictionary of domain-specific concepts for extracting information from a text. When text is given to AutoSlog to be processed, AutoSlog extracts the user-requested information from this text and creates a dictionary containing the relevant words. If the text supplied to the AutoSlog system is related to the subject of the dictionary, the success of the dictionary is high. AutoSlog dictionary is a dictionary containing the domain of terrorist events created in five person-hours of effort. The AutoSlog dictionary was then compared to a handmade dictionary that was made by two talented graduate students, requiring approximately 1500 person-hours of

effort. As a result, the AutoSlog dictionary provided 98% of the performance of the handmade dictionary.

Studies on creating a language dictionary using NLP techniques attempted in different languages. Although studies referring to the English language [34, 35] are more numerous, there are also dictionary generation works in other languages, such as; Tigrinya (Semi-automatic dictionary) [1], Thai (Automatic dictionary) [39], French (Manual dictionary) [16], Arabic (Manual and Semi-automatic dictionaries) [4, 5], and Turkish (Manual dictionary) [2].

In 1999, while at Apple Computer, Silverman et al. [35] described the design and structure of the automatic Victorian dictionary created to support the speech synthesis research and development. The Victorian dictionary consists of five main chapters: (1) polyphony; (2) prosodic context; (3) repetitive speech; (4) function word sequences; and (5) continuous speech. This dictionary is designed to cover a particular aspect of each of the speech synthesis. The Victorian dictionary is written in United States (American) English. The purpose of the Victorian dictionary is to collect of semantic texts throughout the speech. The dictionary was used in the statistical estimation of the time and step models for MacinTalk 4, the next-generation text-to-speech system from Apple.

In 2005, the study by Filippo Vella et al. [37] described a technique for the automatic creation of dictionaries using the sub-symbolic representation of words in cross-language context. Semantic relationship among words of two languages was extracted from aligned bilingual text corpora. This feature was obtained by applying the Latent Semantic Analysis (LSA) technique to the matrices representing terms co-occurrences in aligned text fragments. The technique allows finding the “best translation” according to a properly defined geometric distance in an

automatically created semantic space. Experiments showed that correctness of 95% was obtained in the best case.

In 2008, the study by Bracewell et al. [9], a semi-automatic emotion dictionary was created using the English WordNet dictionary. The proposed algorithm takes as input a set of seed words manually assigned by WordNet senses and emotional information. An initial dictionary was automatically created using seed words and WordNet. Later, this initial dictionary was expanded with reference to the WordNet dictionary. Using the proposed algorithm, a dictionary of approximately 6,000 words was created from 130 seed words. Evaluation of the created dictionary was done with independently labeled corpus of emotions. In the experiments, the proposed algorithm achieved 84% accuracy. In addition, the created dictionary was also used to classify the views of news articles.

In 2010, the study by Julia Pajzs and her team [30], a semi-automatic frequency dictionary in the Hungarian language was created. The dictionary is a collection of 187.6 million words of the most commonly used Hungarian verb phrase (VPC) constructions. Dictionary words were obtained from Hungarian National Library data. Hungarian linguists verified the dictionary manually. It was stated that the obtained results were at a good level and that this dictionary will accelerate the adaptation processes of those who want to learn Hungarian.

In the specific study achieved by Kepuska and Rojanasthien [24], a data collection system was created to produce a manual speech dictionary from movies, TV series, and DVDs. The dictionary provides a lower-cost solution compared with the traditional speech dictionary acquisition method. In addition, it was stated that the collection of data and processing of the dictionary were in short time.

In 2012, an automatic dictionary creation study [39] was executed by Vorapatratorn, Suchato, and Punyabukkana. This study describes the method of automatic dictionary creation using a specific phonetic distribution. Usually, the system selects its data by downloading continuous text from the Internet through a web-based browser. The covetous algorithm is then applied to a given text to extract the appropriate words, and this process continues until the appropriate text dictionary is completed. The results of the study showed that the amount of data withdrawn from the internet achieved the target phonetic distribution and created a telephone coverage area of 99.13% approximately. This text dictionary was used to efficiently produce the speech dictionary.

In 2012, the study by Jozsef Domokos and his team [13], was presented a newly developed Romanian language pronunciation semi-automatic dictionary NaviRO. NaviRO pronunciation dictionary contained more than 100k words from the DexOnline dictionary together with their phonetic transcriptions in Speech Assessment Method Phonetic Alphabet (SAMPA), a machine-readable alphabet. It is in Festival Speech Synthesis System (F3S) dictionary format. The NaviRO dictionary was developed in two stages. In the first stage, using an artificial neural networks (ANN) system having a parallel structure of 30 neural networks developed a 5k word pronunciation dictionary. The automated grapheme-to-phoneme transcription system was tested on a small 1k word, hand-built database. The trained system was able to perform grapheme-to-phoneme transcription with an accuracy of 92.83%, calculated at the phoneme level. In the second stage, built the 100k+ words NaviRO dictionary using Dictionary Maker, a software application created to facilitate the creation of an electronic pronunciation dictionary in a target language starting from the previously created 5k dictionary and a 100k wordlist. The more than 100.000 Romanian words were

collected from the DexOnline dictionary, the largest online dictionary for the Romanian language.

In 2013, the study by Herrero-Zazo et al. [17], an annotated automatic dictionary containing Pharmacological substances and Drug-drug interactions (DDI) was created to reduce the time healthcare professionals spending for review of the biomedical literature and to supply an annotated dictionary for pharmacological substances. The created dictionary had a manually annotated structure consisting of 792 texts selected from the DrugBank database and 233 other Medline abstracts. This fined-grained dictionary was annotated with a total of 18,502 pharmacological substances and 5028 DDIs, including both pharmacokinetics (PK) as well as pharmacodynamics (PD) interactions. In tests, the task of recognizing and classifying pharmacological names achieved an F1 score of 71.5%, while the detection and classification of DDIs achieved an F1 score of 65.1%. These results showed that the created dictionary had sufficient quality in the field of Pharmacovigilance.

In 2013, the study by Behmadi Moghaddas et al. [8], was created a manual dictionary to overcome the obstacles to performing automatic text summarization in Persian. The reason for this obstacle is that there is no standard dictionary for evaluating Persian text summaries. To solve this problem, a dictionary named Pasokh was created. The Pasokh dictionary is a dataset of numerous Persian news documents on various topics. 2000 person-hours were spent to produce this dataset. Then, the success of summarizing was observed by giving single and multiple documents to the Pasokh dictionary. Pasokh dictionary achieved a success rate of 83.56% in single-document summaries and 60.24% in multi-document summaries.

In 2017, Aktas and his colleagues [2], a detailed WordNet literature research conducted. A manual dictionary containing Turkish informatics terms was created. The words in the dictionary are grouped according to their relationship, similarity, and class. It

was stated that the resulting dictionary was larger and more comprehensive than other information dictionaries created up to date.

One of the latest works was completed by Vijay and his team [38]. They created a Hindi-English code-mixed semi-automatic dictionary using tweets published online for the last eight years. To create this dictionary, tweets were taken from Twitter, using the Twitter Python API, which primarily uses the advanced search option in Twitter. All information, such as received tweets, timestamps, Uniform Resource Locator (URL), text, user, retweets, answers, full name, identities, and likes were converted to the JavaScript Object Notation (JSON) format. A comprehensive semi-automated procedure was performed to remove all noisy tweets. After completion of these steps, a dictionary of 2866 words was created, and dictionary words were classified as happiness, sadness, anger, surprise, hate, and multiple emotions. Emotional analysis was performed for the tweets published online, and a 58.2% accuracy was obtained.

In 2018, the study by Fatima-Al-Raisi and her team [3], was presented a new monolingual automatic dictionary for Arabic generated from translating a parallel dictionary of English-French using the Google Translate API. There are no identical words in any of the entries (sentence pairs) in the dictionary. The dictionary contains 100K word pairs. A sample was evaluated by two native speakers of Arabic and was found to be grammatical or nearly grammatical and semantically coherent. Corresponding sentences were found to be both similar in meaning and varied in surface form.

In 2019, a Hinglish (Hindi-English) code change database semi-automatic dictionary called IITG-HingCoS dictionary was developed by Sreeram et al. [14] at Guwahati Institute of Technology, India (IITG). Code-switching was a phenomenon in linguistics, specifically referring to the use of two or more languages

within the same discourse. The created dictionary consisted of code replacement text data a total of 0.58 million words and 25.988 sentences. In addition, the dictionary also contained 25 hours of matched speech data, corresponding to 9251 code replacement phrases, covering a vocabulary of 6542 words. The dictionary was stated to be quite useful for language modeling and automatic speech recognition (ASR) tasks.

In 2021, Alla et al. [6] presented experimentally the methods used, and the difficulties encountered in cohort selection for establishing a clinical patient dictionary in Electronic Health Record (EHR) systems. In the study, it was expressed that basic patient information was usually kept in the form of unstructured clinical notes and it was difficult to process. To overcome this problem, it was aimed to create an automatic patient dictionary by using cancer pathology reports. Using the HL7 (Health Level-7) message standard, 2100 XML-type pathology reports from 1833 (518 males and 1313 females) cancer patients were taken as reference in this study. These documents were separated using NLP techniques, and the dictionary was created by taking the relevant parts from the XML data and adding them to the clinical patient dictionary, according to the prepared manual patterns. It was declared that although the results obtained are generally at an acceptable level, the clinical coding and metadata standards that vary between different hospitals and mostly the low EHR data quality prevents the development of a high-quality dictionary.

In 2022, the study by Aikaterini Kant et al. [23], a methodology was presented for creating custom dictionary word lists for dictionary-based password cracking attacks, starting with a single context seed word. In the study, it was stated that context is very important for a user during password selection and there are not yet automated approaches that can extract and use contextual information during password cracking processes. In addition, it was

stated that people often choose passwords that are related to the subject of the website/system or that are thematically close to that subject when setting passwords. Therefore, using a custom dictionary will provide a significant advantage in password cracking. In the proposed methodology, custom dictionaries were created for any topic of interest that may be required, leveraging the power of structured information found in Wikipedia and DBPedia. Dictionaries are fully customizable. In this way, it was aimed to perform context-based password-cracking attacks. Experiments showed that using the proposed approach with existing approaches resulted in 15.5% more passwords being cracked than existing approaches.

3. Method

In order to identify the meaningful words of the documents given to the system, the documents are pre-processed initially. After meaningful terms are determined, they are now candidates for the dictionary and web search. The general system schema is given in Fig. 1.

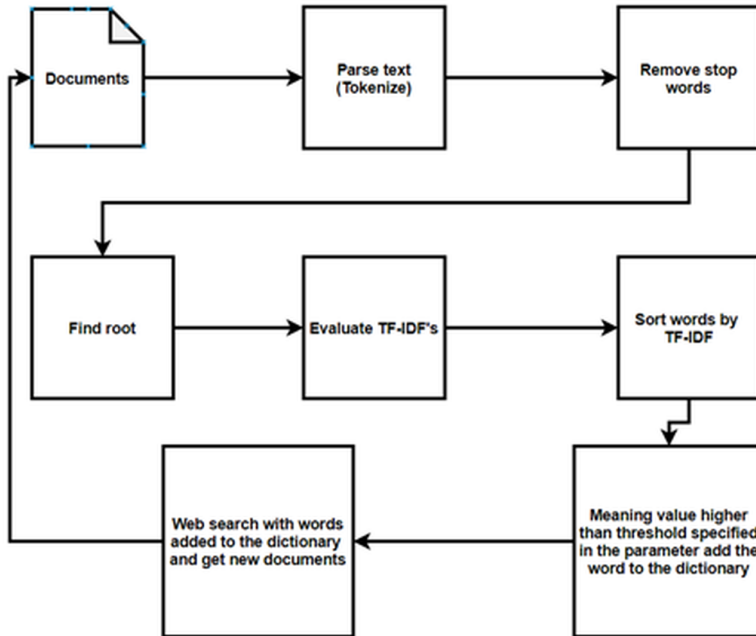


Figure 1. General structure of automatic dictionary creation

3.1. Data pre-processing

Data pre-processing is one of the most important preliminary steps of NLP and is often troublesome but affects the success of the algorithm. In pre-processing, it is necessary to remove prepositions, conjunctions, exclamations, letters, and words with no categorical meaning. These words, called stop words, must be filtered in the first step of data pre-processing. Data pre-processing is applied to each document, and it is used for noise removal, lexicon normalization, and object standardization.

After this process, the document words are converted to the lower-case format. It is then necessary to separate the words from the suffixes and find their root forms. The main purpose of stemming is to reduce different grammatical forms/word forms of a selected word, such as its noun, adjective, verb, or adverb, to its root form. The most common rule-based stemming algorithm for the English

language was developed by Porter, and it is still the most commonly used method to date. The most commonly known and easy-to-use Porter Stemmer algorithm [32] was used in this project. Examples successfully applied to stemming are given by Jivani [22]. Fig. 2 lists the stemming algorithms in NLP.

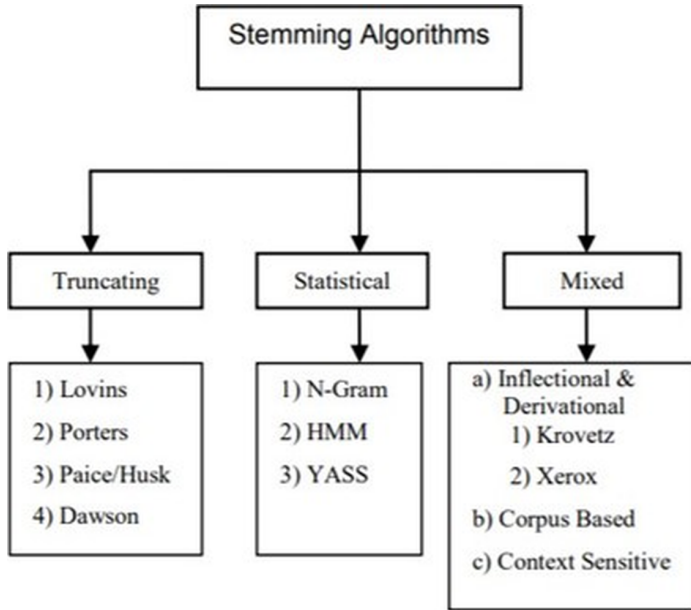


Figure 2. *Types of stemming algorithms*

3.2. Selection of keywords

At this stage, the main system task is to find meaningful words from both the initial given documents and the documents returned from the web search. In this study, the TF-IDF was applied to find the meaning values of words. TF-IDF consists of combination of two different concepts: (1) term frequency and (2) inverse document frequency. The TF-IDF is commonly used to weight each word in the text document according to its level of uniqueness. In other words, the TF-IDF captures the relationship between words, text documents, and specific categories [12].

Suppose we have a document, “t1”, containing 5000 words, and the word “alpha” is present in the document exactly 10 times. It is a very well-known fact that the total length of documents can vary from very small to large, so it is highly possible that a term may occur more frequently in large documents in comparison to small documents. Thus, in order to normalize document length, the occurrence of any term in a document is divided by the total terms present in that document to yield the term’s frequency. So in this case, the term frequency of the word “alpha” in the document “t1” is calculated according to Equation (1).

$$\text{TF}(\text{alpha}) = \frac{10}{5000} = 0.002 \quad (1)$$

Inverse document frequency (IDF) assigns less weight to common words and more weight to less common words. For example, if we have 10 documents and the term “technology”, is available in five of these documents, the IDF value is calculated according to Equation (2).

$$\text{IDF}(\text{technology}) = \log_e \frac{10}{5} = 0.3010 \quad (2)$$

The TF–IDF calculation is equal to the product of the TF and IDF values. TF (w) and IDF (w) values given in Equation (3).

$$\text{TF-IDF}(w) = \text{TF}(w) \times \text{IDF}(w) = 0.002 \times 0.3010 = 0.000602 \quad (3)$$

First, all word frequencies in the document set are recorded in the database. Using the formulas mentioned above, TF, IDF, and TF–IDF values of each word are then calculated. An example of the metric values for the given document set is shown in Table 1. After calculating these values, the words with a meaning value above the threshold are added to the dictionary. The resulting dictionary hash similarity values obtained by adding different threshold meaning values are presented in detail in section 4.

Table 1. *Financial words and TF-IDF scores after the first iteration*

Document Id	Word	TF	IDF	TF-IDF
1	season	0.0422	1	0.0422
1	yard	0.0578	0.6931	0.0400
1	pass	0.0533	0.6931	0.0370
1	player	0.0222	1	0.0222
1	game	0.0200	0.6931	0.0139
1	touchdown	0.0200	0.6931	0.0139
1	touchdown	0.0200	0.6931	0.0139

3.3. Creating the first dictionary words from the starting document

When the document related to the subject of the desired dictionary is initially given to the system, the meaningful words of these seed document are determined using TF-IDF. Only the highest TF-IDF value is added to the dictionary. The web search is then started with this word. However, the rule of determining meaningful candidate words for insertion into the dictionary through the web search is not only the most meaningful word but also the words in which the TF-IDF values are above 0.03.

3.4. Determination of similarity value

When candidate meaningful words are selected after a web search, a similarity value using WordNet similarity is then calculated for each word with respect to the dictionary. Finally, the words with similarities above the similarity threshold value are added to the dictionary. For example, after finding meaningful words for web search results, suppose that three words with the highest TF-IDF values will be added to the dictionary. Suppose these three words are “student”, “math”, and “lesson”. Let the words in the dictionary be “team” and “ball”. A loop is then used to calculate the WordNet similarity of each meaningful word with the words in the dictionary. If we take the word “football”:

WordNet Similarity (“football”, “team”) = 0.35 WordNet Similarity (“football”, “ball”) = 0.30 Total WordNet Similarity (“football”) = 0.35 + 0.30 = 0.65 is the result.

The average WordNet similarity (AWS) value of the word is then calculated using Equation 4.

$$AWS(w) = \frac{\text{TotalWordNetSimilarity}}{\text{DictionaryWordCount}} = \frac{0.65}{2} = 0.325 \quad (4)$$

If the average WordNet similarity parameter value is defined as 0.35 in the application, then the word is not added to the dictionary; on the other hand, if the similarity parameter value is 0.30, the word is added to the dictionary.

WordNet is the English dictionary database in the Cognitive Science Laboratory of Princeton University. Names, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each of which expresses a separate concept. The hypernym feature, a more abstract and general meaning of a word, is used in this database. For example, by looking for the word hypernym for the word cat in the WordNet database, the concept of this word can be determined as the animal. WordNet is free and can be downloaded publicly. The structure of WordNet is therefore a useful tool for computational linguistics and NLP. Fig. 3 shows the WordNet hierarchy [40].

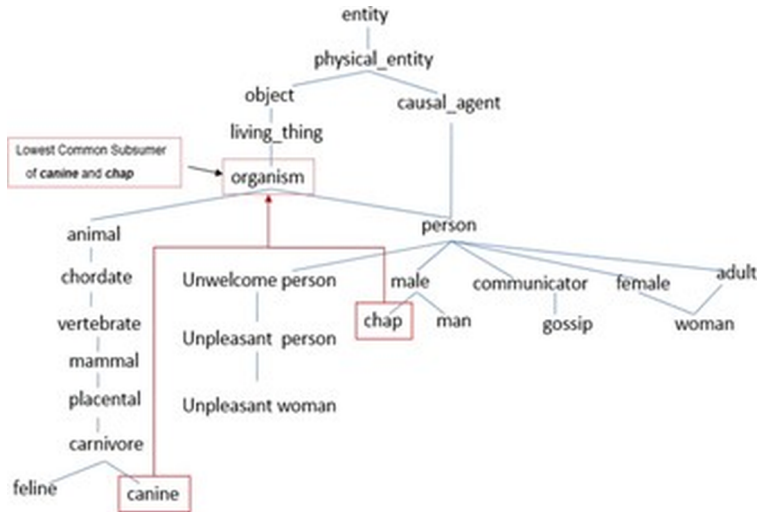


Figure 3. Sample of WordNet hierarchy

3.5. Web search process

The most meaningful word inserted into the dictionary for the last iteration is used in the next web search together with the synonym words obtained from WordNet. For example, if the word added to the dictionary is “student”, the synonym for “student” is “pupil”. In the web search, “student” and “pupil” are used together.

The Azure Cognitive Web Search Interface is used to search on the web. The words to be used on the web search are combined with the space character and given as a parameter to the Azure Web Search Service. For example, the words “Ronaldo”, “Barcelona”, and “El Clasico” are listed consecutively, and these words are used together as “Ronaldo Barcelona El Clasico” in the web search. The web search with these words will result in more than one document. Among these documents, a single document that has not been previously processed is selected. Pre-processing steps are then applied to this document, and new meaningful words are obtained. Thus, the dictionary is constantly expanding with new documents, and the dictionary can be prevented from repeating itself. This

process continues until the dictionary reaches the number of dictionary words in which a parameter value is initially given by the user to the system. Thus, a continuous cycle is provided in the system.

3.6. Determination of general similarity value of dictionary

In this step, the similarity of the dictionary is determined using the SimHash algorithm. The SimHash algorithm is an algorithm used to find similarities between files or websites in applications, such as a search engines, especially in text processing. The SimHash algorithm views two files as vectors and tries to find the cosine link between these vectors.

SimHash is a hash function, and the more similar its property is to the text input, the smaller the Hamming distance of the hash values (Hamming distance is the number of locations in which the corresponding symbols are different). The algorithm works by dividing the text into pieces and combining each piece with a selected function. Each mixed set is represented as a binary vector, and the bit value is converted to +1 or -1 depending on whether the bit value is 1 or 0. In order to obtain SimHash, all bit vectors as bit-wise are collected. Finally, if the total is negative, the bits that result as 1 are otherwise set to 0.

The SimHash algorithm has been used in many different studies for similarity detection. For example, in studies by Jiang and Pi [21, 31], the SimHash algorithm is used to obtain document similarity. Fig. 4 illustrates the working procedure of the SimHash algorithm [18].

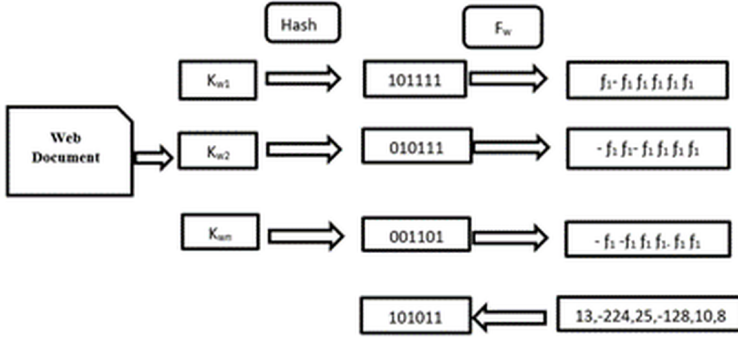


Figure 4. Working procedure of the SimHash algorithm

4. Experimental Results

For all dictionaries, documents belonging to the sports data and containing HTML content used in the study [36] were given to the system as a starting point. First, the number of dictionary words was set at 25, 50, and 100. The effect of the meaning value of a word calculated by the TF-IDF was then evaluated for candidate word selection for the dictionary. Finally, the effect of the number of documents given initially was used in the experiment. For each experiment, the dictionary similarity values were calculated and discussed.

4.1. Experiment I

In Table 2, the hash similarity value for 25 words limited dictionary initialized by only 1 sport (badminton) document is given.

Table 2. Hash similarity value for the 25 words limited dictionary initialized by only one sports document

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.03
Dictionary Maximum Number of Words	25
Number of Starting Documents	1
Dictionary Similarity Value %	75.2

The similarity value of the 25 words dictionary was shown to be higher than the dictionaries with 50 and 100 words. The most important reason for achieving this similarity value was that the new documents obtained from the web causes deviation from the original reference document topic. Table 3 lists the words obtained for parameters given in Table 2.

Table 3. Twenty-five words limited dictionary initialized with only one sports document

Sports Words				
academy	debut	game	people	set
badminton	entry	injury	player	shoe
bwf	event	medal	racket	shop
committee	exercise	meter	sear	sport
court	federation	Olympic	select	world

In Table 4, the hash similarity value for the 50 words limited dictionary initialized by only 1 sport (badminton) document is given.

Table 4. Hash similarity value for the 50 words limited dictionary initialized by only one sports document

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.03
Dictionary Maximum Number of Words	50
Number of Starting Documents	1
Dictionary Similarity Value %	73.8

The similarity value of the 50 words dictionary was higher than the dictionary with 100 words. Obviously, when all parameters were kept constant, and only the number of dictionary words was increased, the dictionary similarity value decreased. Table 5 lists the words obtained for the parameters given in Table 4.

Table 5. *Fifty words limited dictionary initialized with only one sports document*

Sports Words				
academy	country	Excel	player	shoe
arcadia	court	Feder	policy	shop
badminton	cutter	game	postage	sport
Blais	dark	injury	racket	technique
box	debut	league	school	tennis
brand	dimension	ligament	sear	weight
bwf	enjoy	medal	select	wayn
committee	entry	member	service	world
company	equip	meter	set	wrist
cookie	event	Olympic	ship	Yonex

In Table 6, the hash similarity value for the 100 words limited dictionary initialized by only one sport (badminton) document is given.

Table 6. *Hash similarity value for the 100 words limited dictionary initialized by only one sports document*

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.03
Dictionary Maximum Number of Words	100
Number of Starting Documents	1
Dictionary Similarity Value %	68.7

The similarity value of the 100 words dictionary was lower than the similarity values of the 25 and 50 words limited dictionaries. Using this result, we could make an inference: When the maximum

number of words in the dictionary increases, deviations occur in the dictionary after a certain number of words. The factor that causes the deviation originates from the web search that was done with this word after adding an unmeaningful word to the dictionary. In order to prevent deviations in the dictionary, the dictionary should be checked periodically, and words relating to this document should be added to the dictionary based on the initial reference document. Table 7 lists the words obtained for the parameters given in Table 6.

Table 7. One-hundred words limited dictionary initialized with only one sports document

Sports Words				
academi	company	group	och	shuttlecock
address	cookie	guest	Olympia	sign
Alberta	country	har	oppon	Singapore
answer	court	hotel	pain	skill
associ	cutter	image	partner	sport
backhand	dan	Inform	player	team
badminton	debut	ingen	point	technique
ball	double	injury	policy	thesen
basic	draw	job	procedure	tournament
bird	entry	knee	program	use
box	equip	language	question	vatn
brand	event	league	racket	video
build	excel	ligament	read	way
bwf	experiment	line	sear	weight
Canada	fact	match	season	wib
card	fast-paced	medal	select	world
championship	fat	member	service	wrist
club	federation	merge	set	year
coach	format	meter	shoe	Yonex
committee	game	ntrue	shop	zone

4.2. Experiment II

In this experiment, the effect of the meaning threshold value of a word was examined. If the meaning threshold value was selected to be 0.03, words with meaning values greater than this meaning threshold value became candidates for the dictionary. If this threshold was applied through the selection of dictionary words, higher dictionary similarity values were obtained. On the other hand, when words with a meaning value less than 0.03 are added to the dictionary, lower dictionary similarity values are obtained. In the following experiments, the threshold value for the TF-IDF was selected as 0.02, and 25 and 50 words limited dictionaries were created in order.

In Table 8, the hash similarity value for the 25 words limited dictionary initialized with only one sports (badminton) document is given.

Table 8. Hash similarity value for the 25 words limited dictionary initialized by only one sports document

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.02
Dictionary Maximum Number of Words	25
Number of Starting Documents	1
Dictionary Similarity Value %	62.5

One sports document (badminton) was again given to the system, and as a result of the calculation with the TF-IDF, a 25 words-limited dictionary was produced using the words with a meaning value equal to or lower than 0.02. The overall similarity value of the final dictionary was calculated using the hash similarity method, and a 62.5% similarity was obtained. This similarity value was lower than the similarity value of all of the dictionaries obtained by adding the words with the meaning value over 0.03. It was noted that the addition of words lower than the meaningful threshold value

to the dictionary caused deviations and decreased the similarity of dictionary words. Table 9 includes dictionary words obtained in this experiment.

Table 9. *Twenty-five words limited dictionary initialized with only one sports document*

Sports Words				
badminton	committee	Feder	Olympic	sport
ball	court	game	player	state
bwf	day	language	point	street
city	dimens	meter	quarter-final	tennis
coach	exercise	nn	sign	world

In Table 10, the hash similarity value for the 50 words limited dictionary initialized by only one sports (badminton) document is given.

Table 10. *Hash similarity value for the 50 words limited dictionary initialized with only one sports document*

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.02
Dictionary Maximum Number of Words	50
Number of Starting Documents	1
Dictionary Similarity Value %	59.2

The overall similarity of this dictionary was calculated using the Hash similarity method and was equal to 59.2%. This similarity value is lower than the expected similarity value of the 25 words limited dictionary described in Experiment I. Table 11 includes the dictionary words obtained in this experiment.

Table 11. *Fifty words limited dictionary initialized with only one sports document*

Sports Words				
academy	committee	event	Olympic	shoe
article	competition	Feder	play	side
assist	cookie	feedback	player	Singapore
associate	court	game	point	sport
badminton	cup	group	policy	team
book	cutter	Klee	racket	tennis
box	descript	list	read	tournament
bwf	draw	medal	select	world
champion	editor	member	service	year
championship	entry	meter	set	chance

4.3. Experiment III

In this experiment, the effect of the number of documents fed to the system initially was studied. In the following experiments, two different dictionaries of 25 words were created by giving two and three initial documents to the system. In Table 12, the hash similarity value for the 25 words limited dictionary initialized by two sports (badminton)- related documents are given.

Table 12. *Hash similarity value for the 25 words limited dictionary initialized with two sports documents*

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.03
Dictionary Maximum Number of Words	25
Number of Starting Documents	2
Dictionary Similarity Value %	62.7

When all parameters were kept constant, however, the number of documents initially fed to the system were changed, and the similarity value of the dictionary for a small number of initial documents then becomes higher. This finding can be explained by

deviations in the dictionary words based on the initial given documents. Table 13 lists the words obtained for the parameters given in Table 12.

Table 13. *Twenty-five words limited dictionary initialized with two sports documents*

Sports Words				
announce	court	canada	person	team
associate	entry	dimense	rule	ticket
badminton	event	double	service	video
championship	game	inform	side	yonex
contact	bwf	line	sport	zone

In Table 14, the hash similarity value for the 25 words limited dictionary initialized by three sports (badminton) documents is given.

Table 14. *Hash similarity value for the 25 words limited dictionary initialized by three sports documents*

Parameter Name	Parameter Value
Dictionary Similarity Threshold of the Word	0.03
Dictionary Maximum Number of Words	25
Number of Starting Documents	3
Dictionary Similarity Value %	60.3

The similarity value of the 25 words-limited dictionary created with three initial documents was the lowest dictionary similarity value, including almost the dictionaries shown in experiments 1 and 2. It should be noted that even if the TF-IDF meaning value is set to 0.03, dictionaries with a high success rate cannot be obtained when too many initial documents are given. Table 15 lists the words obtained for the parameters given in Table 14.

Table 15. *Twenty-five words limited dictionary initialized with three sports documents*

Sports Words				
associate	badminton	entry	partner	service
august	championship	event	side	time
contact	dog	game	player	world
court	change	hour	point	year
date	dont	inform	score	weight

5. Conclusion

In this study, meaningful words of the seed documents were calculated using the TF-IDF, and a specific domain dictionary was automatically created. Through this study, the effect of establishing a similarity threshold between a candidate word to be inserted into the dictionary, dictionary size, and the effect of the initial number of documents were researched. As a result, we can conclude three findings from these experiments:

- The first one indicates that when the dictionary size grows, the similarity of the dictionary then decreases intuitively as expected. This can be explained by less deviation in the dictionary because the number of dictionary words was small.
- The second one is to establish a bigger threshold for the meaningful value of a word to add to the dictionary to enhance the dictionary similarity. This is a result of selecting more meaningful (similar) words.
- The third one indicates that when the number of initial documents given increases, dictionary similarity significantly decreases as a result of the differences of the contents of the given documents.

The work was compared with similar studies, and it is obvious that the results of the study are quite successful in terms of

average results, dictionary creation, and growth rate. The study [38] mentioned in the literature review section achieved successful results in automatic dictionary creation in two studies. However, unlike [38], the web search section was included in this work. This work is more valuable than the study of [38] in terms of working on instant and current data. Moreover, the WordNet similarity calculation is used in this work to determine whether the words to be added to the dictionary are related to the subject to be created. In general, the following conclusions can be drawn from the results of the work.

- The set of documents should be related to the dictionary you want to create. If the system is presented with a set of documents that are not associated with the desired dictionary, the resulting dictionary will contain meaningless words.
- Once meaningful words are found, the words above the dictionary similarity threshold value are added to the dictionary. When the similarity threshold value is increased, the speed of dictionary creation decreases, while the Hash similarity of the final dictionary increases.
- Web search is made with meaningful words added to the dictionary. When searching the web with a certain percentage of these words, not all meaningful words, the success rate will change positively.
- Querying with the last words added to the dictionary for processing different data continuously during the web search process will prevent duplication of the dictionary. It is therefore useful to mark the last words in the dictionary.

The following studies, which are thought to contribute to the literature, will be discussed.

- Since the suggested system operates in a continuous growth manner, the number of iterations in which the system approaches saturation can be determined.
- A WordNet similarity method was used to calculate the similarity value between the words in the dictionary. According to the WordNet similarity result, it is decided whether or not to add the word to the dictionary. Different similarity techniques could also be examined. For example, the Word2Vec similarity method and WordNet similarity method could be compared. Structural differences between the two dictionaries could then be revealed.
- Once it has been determined to add a word into the dictionary, all synonyms of this word could be added to the dictionary together. The SimHash algorithm could be calculated by comparing the results.

References

[1] Abera, H., H/Mariam, S., 2018. Design of a Tigrinya language speech corpus for speech recognition, in: Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing, Association for Computational Linguistics, Santa Fe, New Mexico, USA. pp. 78–82. URL: <https://aclanthology.org/W18-3811>.

[2] Aktaş, Y., Yılmaz Ince, E., Çakır, A., 2017. Doğal dil İşleme kullanarak bilgisayar ağ terimlerinin wordnet ontolojisinde uyarlanması .

[3] Al-Raisi, F., Lin, W., Bourai, A., 2018. A monolingual parallel corpus of arabic. *Procedia Computer Science* 142, 334–338. doi:10.1016/j.procs.2018.10.487.

[4] Alansary, S., Nagi, M., 2014. The international corpus of arabic: Compilation, analysis and evaluation, pp. 8–17. doi:10.3115/v1/W14-3602.

[5] Alansary, S., Nagi, M.H., Adly, N., 2007. Building an international corpus of arabic (ica) : Progress of compilation stage.

[6] ALLA, N.L.V., CHEN, A., BATONGBACAL, S., NEKKANTTI, C., Dai, H.J., JONNAGADDALA, J., 2021. Cohort selection for construction of a clinical natural language processing corpus. Computer Methods and Programs in Biomedicine Update 1, 100024. URL:

<https://www.sciencedirect.com/science/article/pii/S2666990021000239>, doi:<https://doi.org/10.1016/j.cmpbup.2021.100024>.

[7] Altalib, G., Hassan, H., 2013. A study on analysis of sms classification using tf-idf weighting. International Journal of Computer Networks and Communications Security 1, 189–194.

[8] Behmadi Moghaddas, B., Kahani, M., Toosi, S.A., Pourmasoumi, A., Estiri, A., 2013. Pasokh: A standard corpus for the evaluation of persian text summarizers, in: ICCKE 2013, pp. 471–475. doi:10.1109/ICCKE.2013.6682873.

[9] Bracewell, D., 2008. Semi-automatic creation of an emotion dictionary using wordnet and its evaluation. 2008 IEEE Conference on Cybernetics and Intelligent Systems , 1385–1389.

[10] Chabi, A.H., Kboubi, F., Ahmed, M.B., 2011. Thematic analysis and visualization of textual corpus URL: <https://arxiv.org/abs/1112.2071>, doi:10.48550/ARXIV.1112.2071.

[11] Das, B., Chakraborty, S., 2018. An improved text sentiment classification model using tf-idf and next word negation. ArXiv abs/1806.06407.

[12] Deepshikha Patel, M.B., 2013. Mobile sms classification, in: International Journal of Soft Computing and Engineering (IJSCE). doi:10.1007/978-3-642-38824-8_25.

[13] Domokos, J., Buza, O., Todorean, G., 2012. 100k+ words, machine-readable, pronunciation dictionary for the romanian language, in: 2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO), pp. 320–324.

[14] Ganji, S., Dhawan, K., Sinha, R., 2019. Iitg-hingcos corpus: A hinglish code-switching database for automatic speech recognition. Speech Communication 110, 76–89. URL: <https://www.sciencedirect.com/science/article/pii/S0167639318304217>, doi:<https://doi.org/10.1016/j.specom.2019.04.007>.

[15] Gholamrezazadeh, S., Salehi, M.A., Gholamzadeh, B., 2009. A comprehensive survey on text summarization systems. 2009 2nd International Conference on Computer Science and its Applications , 1–6.

[16] Grabar, N., Claveau, V., Dalloux, C., 2018. Cas: French corpus with clinical cases, in: Louhi@EMNLP.

[17] Herrero-Zazo, M., Segura-Bedmar, I., Martínez, P., Declerck, T., 2013. The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions. Journal of Biomedical Informatics 46, 914–920. URL: <https://www.sciencedirect.com/science/article/pii/S1532046413001123>, doi:<https://doi.org/10.1016/j.jbi.2013.07.011>.

[18] Ho, P.T., Kim, S.R., 2014. Fingerprint-based near-duplicate document detection with applications to sns spam detection. International Journal of Distributed Sensor Networks 10, 612970. URL: <https://doi.org/10.1155/2014/612970>, doi:10.1155/2014/612970,arXiv:<https://doi.org/10.1155/2014/612970>.

[19] Hovy, E., Lin, C.Y., 1998. Automated text summarization and the summarist system, Association for Computational Linguistics, USA. p. 197–214. URL: <https://doi.org/10.3115/1119089.1119121>, doi:10.3115/1119089.1119121.

[20] Indyk, P., Motwani, R., 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality, in: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, NY, USA. p. 604–613. URL: <https://doi.org/10.1145/276698.276876>, doi:10.1145/276698.276876.

[21] Jiang, Q., Sun, M., 2011. Semi-supervised simhash for efficient document similarity search, in: Annual Meeting of the Association for Computational Linguistics.

[22] Jivani, A., 2011. A comparative study of stemming algorithms. Int. J. Comp. Tech. Appl. 2, 1930–1938.

[23] Kanta, A., Coisel, I., Scanlon, M., 2022. A novel dictionary generation methodology for contextual-based password cracking. IEEE Access 10, 59178–59188. doi:10.1109/ACCESS.2022.3179701.

[24] Kepuska, V., Rojanasthien, P., 2011. Speech corpus generation from dvds of movies and tv series. Journal of International Technology and Information Management .

[25] Koeva, Stoyanova, T., Leseva, 2016. Ssemi-automatic compilation of the dictionary of bulgarian multiword expressions, in: Proceedings of GLOBALEX 2016: Lexicographic Resources for Human Language Technology.

[26] Lee, K.M., 2012. Locality-sensitive hashing techniques for nearest neighbor search. International Journal of Fuzzy Logic and Intelligent Systems 12, 300–307. doi:10.5391/IJFIS.2012.12.4.300.

[27] Maksutov, A.A., Zamyatovskiy, V.I., Vyunnikov, V.N., Kutuzov, A.V., 2020. Knowledge base collecting using natural language processing algorithms, in: 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 405–407. doi:10.1109/EIConRus49466.2020.9039303.

[28] Nguyen, T., Shirai, K., 2013. Text classification of technical papers based on text segmentation. doi:10.1007/978-3-642-38824-8_25.

[29] Paik, J.H., 2013. A novel tf-idf weighting scheme for effective ranking, in: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery, New York, NY, USA. p. 343–352. URL: <https://doi.org/10.1145/2484028.2484070>, doi:10.1145/2484028.2484070.

[30] Pajzs, J., Sass, B., 2010. Towards semi-automatic dictionary making – creating the frequency dictionary of hungarian verb phrase constructions.

[31] Pi, B., Fu, S., Wang, W., Han, S., 2009. Simhash-based effective and efficient detecting of near-duplicate short messages.

[32] Porter, M.F., 2001. Snowball: A language for stemming algorithms.

[33] Qaiser, S., Ali, R., 2018. Text mining: Use of tf-idf to examine the relevance of words to documents. International Journal of Computer Applications .

[34] Riloff, E., 1993. Automatically constructing a dictionary for information extraction tasks, in: Proceedings of the Eleventh National Conference on Artificial Intelligence, AAAI Press. p. 811–816.

[35] Silverman, Anderson, Bellegarda, Lenzo, Naik, 1999. Design and collection of a corpus of polyphones and prosodic contexts for speech synthesis research and development.

[36] Turan, M., Ögtelik, S., 2018. İngilizce dokümanlarda tema ve alt kavramlar tespit modeli. Düzce Üniversitesi Bilim ve Teknoloji Dergisi , 754 – 764doi:10.29130/dubited.420104.

[37] Vella, F., Pilato, G., Motisi, I., Gaglio, S., 2005. Automatic dictionary creation by sub-symbolic encoding of words, pp. 113–119. doi:10.1007/11731177_17.

[38] Vijay, D., Bohra, A., Singh, V., Akhtar, S.S., Shrivastava, M., 2018. Corpus creation and emotion prediction for Hindi-English code-mixed social media text, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop, Association for Computational Linguistics, New Orleans, Louisiana, USA. pp. 128–135. URL: <https://aclanthology.org/N18-4018>, doi:10.18653/v1/N18-4018.

[39] Vorapatratorn, S., Suchato, A., Punyabukkana, P., 2012. Automatic online text selection for constructing text corpus with custom phonetic distribution. 2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE) , 6–11.

[40] Wei, Chang, 2015. Measuring word semantic relatedness using wordnet-based approach. Journal of Computers , 252–259.

CHAPTER 2

Machine Learning-Based Sentiment Analysis of Conversational AI in Turkish Microblogs

AYHAN AYDIN¹

Introduction

Artificial intelligence (AI) has undergone a historic paradigm shift over the past decade, evolving from constrained, task-specific expert systems into highly capable, general-purpose generative architectures (Russell & Norvig, 2020; McCarthy et al., 2006). The emergence of Large Language Models (LLMs) and advanced conversational agents—most notably OpenAI's ChatGPT, Google's Gemini, and xAI's Grok—represents a watershed moment in human-computer interaction, re-architecting workflows across creative writing, technical coding, biomedical literature extraction, and enterprise decision-making (Berente et al., 2021; OpenAI, 2023; Sætra, 2023). These next-generation conversational AI tools perform complex textual synthesis, multi-step logical reasoning, and interactive problem-solving, fundamentally transforming societal expectations and professional labor dynamics (Kırık & Özkoçak, 2023; Sucu, 2019).

As these foundation models permeate everyday life, public discourse on their capabilities, ethical implications, labor-substitution risks, and hallucination tendencies has intensified across global communication channels (Gazaz & Gazaz, 2024; Sætra, 2023).

¹ PhD, Ondokuzmayıs University, Information and Technology, Orcid: 0000-0001-9127-0951

Microblogging platforms, most prominently Twitter (rebranded as X), serve as vast, real-time public repositories where millions of user interactions, affective reactions, and unedited opinions are continuously broadcast (Pak & Paroubek, 2010; Go et al., 2009). The high-velocity data streams generated on Twitter/X capture unfiltered societal sentiment, providing researchers with an unprecedented digital laboratory to observe real-time collective intelligence, public approval, and emerging anxieties regarding technological disruptions (Asur & Huberman, 2010; Bollen et al., 2011; Bruns & Stieglitz, 2012, 2013).

Automated sentiment classification—the computational extraction and categorization of opinions, evaluations, and affective states from unstructured text—plays a crucial role in understanding this massive digital footprint (Pang & Lee, 2008; Liu, 2012; Medhat et al., 2014). However, analyzing Turkish social media streams introduces formidable computational and linguistic challenges (Gezici & Yanikoğlu, 2018; Vural et al., 2013). Turkish is an agglutinative, morphologically rich language belonging to the Ural-Altaic family, characterized by extensive suffixation, free word order, complex vowel harmony, and expressive colloquial variance (Özcan & Çelik, 2019; Erdem & Doğan, 2020). Furthermore, microblog texts on Twitter/X are notoriously noisy, heavily saturated with informal slang, abbreviations, emoticons, misspellings, sarcasm, non-standard punctuation, and domain-specific terminology (Dave et al., 2003; Wang et al., 2012; Janetzko, 2017).

To overcome the non-linear complexities and high dimensionality inherent in Turkish text classification, traditional econometric linear models often prove inadequate, necessitating the deployment of robust machine learning algorithms and artificial neural network paradigms capable of modeling intricate feature interactions (Aydın, Atila, & Aydın, 2018; Vapnik, 1998; Cortes & Vapnik, 1995). Recent advances in Natural Language Processing (NLP) have witnessed a

transition from traditional lexicon-based methodologies (Taboada et al., 2011; Mohammad & Turney, 2013; Hutto & Gilbert, 2014) and handcrafted feature engineering toward supervised machine learning classifiers (Pang et al., 2002; Mitchell, 1997) and deep contextual transformer architectures like BERT (Devlin et al., 2019; Vaswani et al., 2017; Yıldırım, 2020).

This chapter provides a comprehensive empirical investigation into public sentiment expressed on Turkish Twitter (X) regarding next-generation conversational AI models. Utilizing a structured corpus of 2,002 Turkish microblog posts gathered between January 2024 and May 2025 across four primary keyword queries ('ChatGPT', 'Grok', 'Gemini', and 'Yapay Zeka'), this study combines transformer-based automated annotation via `savasy/bert-base-turkish-sentiment-cased` with classical supervised machine learning classifiers—Multinomial Naïve Bayes (MNB), Support Vector Machines (SVM), and Logistic Regression (LR). Through rigorous empirical benchmarking, confusion matrix evaluation, and qualitative thematic synthesis, this research illuminates how Turkish social media users perceive the benefits, limitations, and societal risks of modern conversational AI.

Related Work and Literature Review

The field of sentiment analysis and opinion mining emerged as a vital subfield of Computational Linguistics and Information Retrieval in the early 2000s, initially focusing on structured product reviews, movie critiques, and news articles (Pang et al., 2002; Dave et al., 2003; Pang & Lee, 2008; Manning et al., 2008). The explosive proliferation of Web 2.0 and social media platforms shifted academic focus toward microblog analytics, where unconstrained user-generated text presents both vast quantitative scale and severe qualitative noise (Pak & Paroubek, 2010; Go et al., 2009; Liu, 2012). Early pioneering work by Go et al. (2009) established the viability of distant supervision, utilizing emoticons as noisy labels to train supervised classifiers on Twitter data, achieving

classification accuracies exceeding 80% with Support Vector Machines and Naïve Bayes.

Subsequent studies demonstrated that microblog sentiment dynamics possess strong predictive power across real-world socio-economic domains. Asur and Huberman (2010) proved that Twitter sentiment volume and polarity significantly outperform traditional survey-based models in forecasting box-office revenues. Bollen et al. (2011) examined multi-dimensional mood states extracted from Twitter feeds and established a statistically significant correlation with daily fluctuations in the Dow Jones Industrial Average. In political science, Wang et al. (2012) developed a real-time Twitter analytics system to track public opinion during the 2012 U.S. presidential election cycle, highlighting how microblog sentiment mirrors a shift in candidate popularity. Systematic frameworks for Twitter communication metrics established by Bruns and Stieglitz (2012, 2013) further standardized quantitative indicators for user engagement, retweeting cascades, and hashtag diffusion patterns during major public events.

Extending sentiment classification paradigms to the Turkish language requires addressing severe morphological complexity and structural divergence from Indo-European languages (Gezici & Yanıkoğlu, 2018; Jurafsky & Martin, 2023). Early foundational research by Vural et al. (2013) evaluated lexicon-based and supervised learning methods on Turkish commercial product reviews, establishing that morphological stemmers and root extractors significantly influence vector space representations. Subsequent benchmark evaluations by Aktaş et al. (2021) compared traditional machine learning algorithms across diverse Turkish news and microblog datasets, demonstrating that Support Vector Machines and Logistic Regression consistently outperform probabilistic Naïve Bayes models when paired with unigram and bigram term representations.

In a seminal empirical survey, Onan (2016) systematically evaluated Naïve Bayes, Support Vector Machines, Logistic Regression, Random Forests, and k-Nearest Neighbors on Turkish Twitter feeds, revealing that ensemble voting schemes and term-weighting modifications substantially enhance classification

accuracy over single base learners. Subsequently, Onan (2018) introduced an integrated framework incorporating psychological and linguistic feature sets (LIWC) alongside n-gram vectors, demonstrating that affective dictionary features boost macro-F1 scores in Turkish microblog streams.

In parallel, regional scholars have expanded Turkish social media text mining across diverse domain applications. Kaynar et al. (2016) compared machine learning algorithms for sentiment classification on Turkish social media posts, highlighting the sensitivity of SVM classifiers to term weighting schemes. Küçüksille and Selver (2019) analyzed customer feedback on Turkish e-commerce platforms, demonstrating that pre-processed n-gram features combined with supervised learners yield reliable commercial insights. Cebeci (2020) and İlhan & Sağaltıcı (2020) conducted comprehensive sentiment mapping of public opinions on Twitter, emphasizing that Turkish slang, sarcasms, and colloquial spelling variations represent the primary bottlenecks for automated classifiers. Tümüklü Özyer and Özyer (2018, 2020) designed specialized monitoring and analytics architectures for Turkish Twitter data streams, emphasizing real-time preprocessing and feature extraction techniques tailored for Turkish social network feeds.

Further methodological contributions by Baykal & Gürbüz (2021) and Çakır & Özel (2022) explored Turkish social media text mining and sentiment detection, highlighting the importance of domain-specific preprocessing pipelines. Kumaş (2021) evaluated diverse supervised classifiers on Turkish Twitter feeds, confirming that model selection depends heavily on corpus composition and feature density.

Machine Learning and Artificial Neural Network Applications in Turkish Domain

Beyond social media text processing, the broader Turkish computational literature has consistently highlighted the superiority of non-linear machine learning architectures and Artificial Neural Networks (ANN) over traditional linear statistical models when handling complex, non-stationary data (Aydın, Atila, & Aydın, 2018; Hastie et al., 2009). In a pivotal study, Aydın, Atila, and Aydın (2018) investigated the forecasting of demographic indicators in Turkey by comparing Multilayer Perceptron (MLP) neural networks with traditional econometric time-series models. Incorporating unit root stationarity tests—including Augmented Dickey-Fuller (ADF), Phillips-Perron (PP), and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests—Aydın, Atila, and Aydın (2018) established that ANN architectures effectively capture complex non-linear structures and underlying structural breaks that conventional linear regression frameworks fail to accommodate.

The theoretical principles underscored by Aydın, Atila, and Aydın (2018) directly extend to high-dimensional NLP and text mining contexts. Textual feature spaces derived from microblog streams exhibit non-linear term dependencies, extreme sparsity, and high variance (Vapnik, 1998; Hastie et al., 2009). Consequently, applying advanced supervised machine learning algorithms—such as Support Vector Machines operating with non-linear margin hyperplanes (Cortes & Vapnik, 1995) or deep transformer neural networks (Devlin et al., 2019; Vaswani et al., 2017)—is essential to model the intricate semantic relationships present in Turkish social media feeds.

The societal integration of artificial intelligence has attracted widespread academic scrutiny across communication, sociology, and information systems disciplines (Russell & Norvig, 2020; Berente et

al., 2021). Historical investigations by McCarthy et al. (2006) and Mitchell (1997) laid the theoretical groundwork for machine learning, while contemporary scholars focus on public discourse surrounding AI safety, ethics, and labor automation (Sætra, 2023; OpenAI, 2023). Sucu (2019) examined public attitudes toward AI disruptions in Turkey, noting that societal sentiment oscillates between techno-optimistic enthusiasm for economic productivity and deep-seated anxiety regarding structural unemployment.

With the release of ChatGPT and subsequent conversational LLMs, public sentiment research has pivoted toward evaluating user experience and perceptions in real time (Gazaz & Gazaz, 2024; Kırık & Özkoçak, 2023). Gazaz and Gazaz (2024) analyzed user entries on Ekşi Sözlük about conversational AI chatbots, revealing that public evaluation centers on response reliability, tendency to hallucinate, Turkish language fluency, and creative problem-solving capability. Bural (2022) emphasized the strategic importance of social media intelligence and text analytics for monitoring public sentiment online during national policy formulation. Understanding how Turkish microblog users perceive specific conversational AI tools—ChatGPT, Gemini, and Grok—provides critical insights for developers, policy makers, and enterprise leaders seeking to align AI deployment with societal needs.

Digital communication researchers have further analyzed social media discourse and digital public engagement during technological transitions (Kırık, 2021; Yılmaz & Arslan, 2022). Yılmaz and Arslan (2022) evaluated users' perceptions of AI chatbot performance in Turkish text processing, providing empirical evidence that user satisfaction closely aligns with linguistic naturalness and semantic accuracy.

Methodology and Experimental Design

The methodological pipeline executed in this empirical research consists of five sequential stages: (1) Data acquisition and API-based microblog harvesting, (2) Textual preprocessing and Turkish language standardization, (3) Automated sentiment polarity annotation using a pre-trained transformer model, (4) High-dimensional feature vectorization via TF-IDF, and (5) Supervised machine learning classification and confusion matrix benchmarking.

Data collection was conducted between January 1, 2024, and May 31, 2025, capturing Turkish-language microblog posts published on Twitter (X). Custom Python harvesting scripts utilizing Twitter/X API endpoints retrieved microblogs containing four target search queries: 'ChatGPT', 'Grok', 'Gemini', and 'Yapay Zeka' (the general Turkish term for Artificial Intelligence). To ensure sample quality and minimize redundant noise, duplicate tweets, automated promotional bot broadcasts, retweets without user commentary, and posts containing fewer than three alphabetic words were purged (Pak & Paroubek, 2010; Bruns & Stieglitz, 2013). The final filtered corpus comprises 2,002 unique Turkish microblog posts.

Systematic data sampling procedures tailored for social media stream architectures were incorporated (Haug, 2019; Hayran & Anagün, 2020).

Microblog text contains substantial non-semantic noise that disrupts statistical term vectorization (Dave et al., 2003). To standardize the Turkish microblog corpus, a rigorous five-step preprocessing workflow was applied:

1. URL and Link Removal: Web links (HTTP/HTTPS) and domain URLs were eliminated using regular expressions (Pak & Paroubek, 2010).

2. User Handle and Hashtag Cleaning: Twitter user mentions (@username) and structural hashtag symbols (#) were stripped to prevent user-specific bias while preserving textual hashtag roots (Bruns & Stieglitz, 2012).
3. Punctuation and Numeric Stripping: Non-alphabetic symbols, numeric digits, and excessive punctuation were removed, retaining standard Turkish characters (a-z, ç, ğ, ı, ö, ş, ü) (Gezici & Yanıkoğlu, 2018).
4. Case Normalization: All text was converted to standard lower case, explicitly accounting for Turkish-specific character mappings (e.g., converting 'İ' to 'i' and 'T' to 't') (Aktaş et al., 2021).
5. Stop-Word Filtering: High-frequency functional words conveying minimal sentiment orientation (e.g., 've', 'veya', 'ama', 'fakat', 'de', 'da', 'bu', 'şu', 'için') were removed using a customized Turkish stop-word dictionary (Onan, 2016).

Text-processing libraries and NLTK tools adapted for agglutinative morphology were used during cleaning (Bird et al., 2009; Loria, 2018).

Establishing reliable ground-truth sentiment labels for 2,002 microblog posts requires objective annotation (Go et al., 2009; Devlin et al., 2019). In this study, ground-truth polarity labels were generated using savasy/bert-base-turkish-sentiment-cased, a specialized transformer model fine-tuned on Turkish social media and product review corpora (Yıldırım, 2020; Devlin et al., 2019). Each microblog post was passed through the transformer model, producing softmax probability scores for each sentiment class. Tweets assigned neutral sentiment were excluded to establish a crisp

binary classification framework (Positive vs. Negative), matching standard benchmark designs in microblog sentiment research (Pang et al., 2002; Go et al., 2009).

Following preprocessing, textual strings were converted into numerical vector space representations using Term Frequency-Inverse Document Frequency (TF-IDF) weighting (Salton & Buckley, 1988; Manning et al., 2008). The TF-IDF weight for a term t in a document d within corpus D is computed as $\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \log(|D| / |\{d \in D : t \in d\}|)$, where $\text{TF}(t, d)$ denotes the term frequency of t in document d , and the log factor measures inverse document frequency. Unigram and bigram feature representations were extracted, establishing a high-dimensional sparse matrix optimized for supervised machine learning algorithms (Aktaş et al., 2021).

Supervised Machine Learning Classifiers:

Three prominent supervised machine learning algorithms were implemented and empirically benchmarked:

1. Multinomial Naïve Bayes (MNB): A probabilistic classifier applying Bayes' theorem with strong conditional independence assumptions between features (Rish, 2001; Pang et al., 2002). MNB is computationally efficient and serves as a classic baseline in text classification (Pak & Paroubek, 2010).
2. Support Vector Machines (SVM): A linear classifier that seeks an optimal separating hyperplane maximizing the margin between positive and negative sentiment classes in high-dimensional feature space (Cortes & Vapnik, 1995; Vapnik, 1998). SVM is highly effective for sparse text vector spaces (Aktaş et al., 2021).

3. Logistic Regression (LR): A linear model that estimates log-odds probabilities using a sigmoid function combined with L2 regularization to prevent overfitting on high-dimensional text data (Hosmer et al., 2013).

The dataset was partitioned into an 80% training set (1,601 microblogs) and a 20% independent testing set (401 microblogs) using stratified random sampling to preserve sentiment class balance (Hastie et al., 2009). Classifier performance was evaluated across five standard statistical metrics: Accuracy, Precision, Recall, Macro F1-Score, and Receiver Operating Characteristic - Area Under Curve (ROC-AUC) (Mitchell, 1997; Manning et al., 2008).

Empirical Findings and Analysis

The final experimental dataset comprises 2,002 Turkish microblog posts distributed across four primary AI target keywords. Table 1 presents the quantitative sentiment distribution and percentage shares across these target topics.

Table 1: Sentiment Distribution and Percentages Across Target AI Topics in Turkish Microblogs.

Target AI Topic	Total Posts	Share (%)	Negative Count	Positive Count
ChatGPT	1,001	50.0%	578 (57.7%)	423 (42.3%)
General Artificial Intelligence	501	25.0%	291 (58.1%)	210 (41.9%)
Gemini	300	15.0%	179 (59.7%)	121 (40.3%)
Grok	200	10.0%	123 (61.5%)	77 (38.5%)
Total Dataset	2,002	100.0%	1,171 (58.5%)	831 (41.5%)

As detailed in Table 1, ChatGPT constitutes the largest portion of public discussion, representing exactly 50.0% (1,001 posts) of the entire corpus. General Artificial Intelligence discussions ('Yapay Zeka') account for 25.0% (501 posts), followed by Google's Gemini (15.0%, 300 posts) and xAI's Grok (10.0%, 200 posts). Across the

aggregate dataset, negative sentiment represents a distinct majority at 58.5% (1,171 posts), whereas positive sentiment accounts for 41.5% (831 posts). Grok recorded the highest proportion of negative posts (61.5%), followed by Gemini (59.7%), General AI (58.1%), and ChatGPT (57.7%). This predominant negative tilt highlights substantial public skepticism, privacy concerns, and critical scrutiny regarding LLM outputs in Turkey.

The performance of Multinomial Naïve Bayes, Support Vector Machines, and Logistic Regression on the 401-post testing set was evaluated across Accuracy, Precision, Recall, Macro F1-Score, and ROC-AUC. Table 2 summarizes the comparative benchmark metrics.

Table 2: Comparative Performance Metrics Across Supervised Machine Learning Models

Classifier Model	Accuracy	Precision	Recall	Macro F1	ROC-AUC
Logistic Regression (LR)	68.82%	68.50%	69.10%	68.75%	0.742
Support Vector Machine (SVM)	67.83%	67.40%	68.20%	67.78%	0.728
Multinomial Naïve Bayes (MNB)	65.58%	65.10%	66.00%	65.52%	0.698

Empirical results in Table 2 demonstrate that Logistic Regression achieved the highest overall predictive accuracy of 68.82%, along with a Macro F1-Score of 68.75% and an ROC-AUC of 0.742. Support Vector Machines achieved highly competitive results, attaining 67.83% accuracy and a Macro F1-Score of 67.78%. Multinomial Naïve Bayes achieved lower performance (65.58% accuracy, 0.698 ROC-AUC), reflecting the limitations of its feature-independence assumption when handling high-dimensional n-gram representations (Rish, 2001; Aktaş et al., 2021).

To gain granular insights into classification errors, the confusion matrix for the top-performing Logistic Regression model

was analyzed on the 401-post test partition (comprising 235 negative posts and 166 positive posts). Table 3 provides the detailed confusion matrix.

Table 3: Confusion Matrix for Logistic Regression Classifier on Independent Test Set.

True \ Predicted Class	Predicted Negative	Predicted Positive
Actual Negative (N=235)	192 (True Negative - 81.7%)	43 (False Positive - 18.3%)
Actual Positive (N=166)	82 (False Negative - 49.4%)	84 (True Positive - 50.6%)

The confusion matrix in Table 3 reveals significant asymmetry in classification. The model demonstrated high specificity on negative microblogs, correctly classifying 192 out of 235 negative posts (81.7% True Negative rate). However, sensitivity on positive posts was noticeably lower, correctly identifying only 84 out of 166 positive microblogs (50.6% True Positive rate), while misclassifying 82 positive posts as negative (49.4% False Negative rate). Qualitative error analysis reveals that positive microblogs frequently contained implicit praise masked by technical jargon, subtle sarcasm, or mixed-sentiment clauses, causing traditional TF-IDF feature representations to misclassify them as the dominant negative category (Gezici & Yanıkoğlu, 2018; İlhan & Sağaltıcı, 2020).

Discussion and Strategic Implications

The empirical findings of this study provide critical insights into public attitudes toward conversational artificial intelligence in Turkey, highlighting structural themes across technical, socio-economic, and linguistic dimensions.

1. **Dominance of Skepticism and Critical Scrutiny:** The 58.5% majority share of negative sentiment underscores

widespread public apprehension regarding LLM deployments. Qualitative text analysis reveals that negative posts center heavily on concerns regarding factual hallucinations, lack of source citations, bias, and technical unreliability during complex problem-solving tasks (Gazaz & Gazaz, 2024; Sætra, 2023). Users frequently voice frustration when conversational AI agents produce confidently stated incorrect answers in Turkish.

2. Intellectual Property, Ethics, and Labor Disruption: Microblog discussions frequently reflect anxiety concerning labor substitution in creative and technical sectors, plagiarism, data privacy issues, and copyright protection (Berente et al., 2021; Sucu, 2019; Kırık & Özkoçak, 2023). Users express concerns regarding how personal data and proprietary content are harvested to train foundation models without explicit consent.
3. Linguistic Disparities in Turkish Performance: A prominent qualitative theme involves performance disparities between English and Turkish outputs. Microblog users frequently note that while LLMs demonstrate impressive fluency in English, their Turkish outputs occasionally exhibit unnatural syntax, literal translation artifacts, and limited comprehension of local cultural idioms (Gezici & Yanıkoğlu, 2018; Erdem & Doğan, 2020).
4. Productivity Enthusiasm vs. Implementation Barriers: Conversely, the 41.5% positive sentiment share reflects strong public enthusiasm for automated productivity, rapid code generation, language translation, and educational assistance (OpenAI, 2023; Kırık & Özkoçak, 2023). Users celebrate

conversational AI as a transformative tool that accelerates routine cognitive tasks.

Strategic Recommendations for Developers, Policy Makers, and Researchers:

- For AI Developers and Tech Corporations: Fine-tuning foundation models on high-quality, culturally nuanced Turkish corpora is essential to eliminate unnatural syntax and reduce hallucination rates (Yıldırım, 2020; Devlin et al., 2019). Implementing transparent attribution and source verification mechanisms will directly address public skepticism.
- For Educational Leaders and Policy Makers: Policy frameworks should shift from restrictive bans toward establishing digital literacy curricula that educate students and professionals on effective prompt engineering, critical verification, and ethical AI usage (Berente et al., 2021; Sætra, 2023).
- For NLP and Data Science Researchers: Incorporating non-linear neural models, ensemble architectures, and advanced feature selection techniques—as demonstrated by Aydın, Atila, and Aydın (2018)—is crucial to overcoming high-dimensional noise and contextual ambiguity in Turkish social media text mining.

Conclusion, Limitations, and Future Directions

This chapter presented an empirical investigation into public sentiment expressed on Turkish Twitter (X) regarding next-generation conversational AI models. Using a structured dataset of 2,002 microblog posts, transformer-based automated annotation with `savasy/bert-base-turkish-sentiment-cased`, and supervised machine learning benchmarking, this study established that public

discourse in Turkey is predominantly critical (58.5% negative share), driven by concerns about model hallucinations, data privacy, and labor disruptions. Among evaluated classifiers, Logistic Regression achieved the highest accuracy (68.82%), closely followed by Support Vector Machines (67.83%).

The primary limitation of this study is the difficulty of detecting implicit sarcasm, colloquial idioms, and multi-word sentiment expressions with unigram/bigram TF-IDF feature representations. Future research will focus on fine-tuning domain-specific deep bidirectional transformer architectures (e.g., BERTurk, ELECTRA) and integrating hybrid deep learning models (LSTM, CNN-BiLSTM) to improve sensitivity on positive and sarcasm-laden microblog streams (Devlin et al., 2019). Additionally, expanding data collection across multimodal social platforms (YouTube comments, Ekşi Sözlük, Reddit) will provide a more comprehensive mapping of societal dynamics in AI adoption.

References

- Aktaş, Ö., Coşkuner, B., & Soner, İ. (2021). Turkish sentiment analysis using machine learning methods on social media. *Journal of Advanced Research in Computer Engineering & Technology*, 10(1), 1–10.
- Asur, S., & Huberman, B. A. (2010). Predicting the future with social media. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (Vol. 1, pp. 492–499). IEEE. <https://doi.org/10.1109/WI-IAT.2010.63>
- Aydın, A., Atila, Ü., & Aydın, S. (2018). Use of ANN in predicting life expectancy: The case of Turkey. *Artificial Intelligence Studies*, 1(1), 1–7. <https://doi.org/10.30855/ais.2018.01.01.01>
- Baykal, A., & Gürbüz, E. (2021). Sosyal medya veri madenciliği ve duygu tespiti [Social media data mining and sentiment detection]. *Bilim ve Teknoloji Dergisi*, 9(2), 112–124.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing artificial intelligence. *MIS Quarterly*, 45(3), 1433–1450. <https://doi.org/10.25300/MISQ/2021/16274>
- Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: Analyzing text with the Natural Language Toolkit*. O'Reilly Media.
- Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1), 1–8. <https://doi.org/10.1016/j.jocs.2010.12.007>
- Bruns, A., & Stieglitz, S. (2012). Quantitative approaches to comparing communication patterns on Twitter. *Journal of Technology in Human Services*, 30(3–4), 160–185. <https://doi.org/10.1080/15228835.2012.744249>
- Bruns, A., & Stieglitz, S. (2013). Metrics for tweeting activities: Towards more systematic Twitter analysis. *Social Science Computer Review*, 31(1), 91–108. <https://doi.org/10.1177/0894439312451564>
- Bural, E. B. (2022). Social media intelligence and network analytics in public security. *Journal of Intelligence Research and Studies*, 1(1), 45–62.

- Çakır, F., & Özel, S. (2022). Dijital mecralarda duygu analizi ve doğal dil işleme uygulamaları [Sentiment analysis and NLP applications in digital media]. *İletişim Kuram ve Araştırma Dergisi*, (58), 89–106.
- Cebeci, H. İ. (2020). *Sosyal medya verileri ile duygu analizi* [Sentiment analysis with social media data]. Sakarya Üniversitesi Yayınları.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. <https://doi.org/10.1007/BF00994018>
- Dave, K., Lawrence, S., & Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and sentiment classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web* (pp. 519–528). ACM. <https://doi.org/10.1145/775152.775226>
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 4171–4186). Association for Computational Linguistics. <https://doi.org/10.18653/v1/N19-1423>
- Erdem, M., & Doğan, A. (2020). Türkçe metinlerde makine öğrenmesi ile duygu sınıflandırması [Sentiment classification in Turkish texts using machine learning]. *Veri Bilimi Dergisi*, 3(1), 25–34.
- Gazaz, D. C., & Gazaz, G. (2024). Geleceğin iletişimde yapay zekâ sohbet robotları: Ekşi Sözlük girdilerinin içerik analizi [AI chatbots in future communication: Content analysis of Ekşi Sözlük entries]. *Medya ve İletişim Araştırmaları Dergisi*, 7(1), 15–32.
- Gezici, G., & Yanikoğlu, B. (2018). Sentiment analysis in Turkish. In K. Oflazer & M. Saraçlar (Eds.), *Turkish natural language processing* (pp. 255–273). Springer. https://doi.org/10.1007/978-3-319-90165-7_12
- Go, A., Bhayani, R., & Huang, L. (2009). *Twitter sentiment classification using distant supervision* [CS224N Project Report]. Stanford University.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer. <https://doi.org/10.1007/978-0-387-84858-7>

- Haug, L. E. (2019). *Analyzing the quality of Twitter data streams and sentiment architecture* [Technical report]. HP Laboratories.
- Hayran, A., & Anagün, A. S. (2020). Generalized durative event detection and sentiment analysis on social media streams. *IEEE Access*, 8, 142100–142112.
- Hosmer, D. W., Jr., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (3rd ed.). John Wiley & Sons. <https://doi.org/10.1002/9781118548387>
- Hutto, C. J., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1), 216–225. <https://doi.org/10.1609/icwsm.v8i1.14550>
- İlhan, N., & Sağaltıcı, D. (2020). Twitter'da duygu analizi [Sentiment analysis on Twitter]. *Harran Üniversitesi Mühendislik Dergisi*, 5(2), 110–121.
- Janetzko, H. (2017). Visual sentiment analysis on Twitter data streams. In *Proceedings of the International Conference on Information Visualization* (pp. 88–95). IEEE.
- Jurafsky, D., & Martin, J. H. (2023). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (3rd ed. draft). <https://web.stanford.edu/~jurafsky/slp3/>
- Kaynar, O., Görmez, Y., Yıldız, M., & Albayrak, A. (2016). Makine öğrenmesi yöntemleri ile duygu analizi [Sentiment analysis with machine learning techniques]. *Veri Bilişimi Dergisi*, 2(1), 11–20.
- Kırık, A. M. (2021). Sosyal medyada dijital aktivizm ve kamuoyu oluşturma [Digital activism and public opinion formation in social media]. *Dijital İletişim Araştırmaları*, 3(1), 45–58.
- Kırık, A. M., & Özkoçak, V. (2023). Medya ve iletişim bağlamında yapay zekâ tarihi ve teknolojisi [History and technology of artificial intelligence in media and communication]. *İletişim ve Diplomasi*, (9), 77–96.
- Küçükşille, E. U., & Selver, M. A. (2019). E-ticaret sitelerindeki Türkçe ürün yorumları üzerine makine öğrenmesi algoritmaları ile duygu analizi

- [Sentiment analysis on Turkish product reviews using machine learning algorithms]. *Veri Bilimi*, 2(2), 35–48.
- Kumaş, E. (2021). Türkçe Twitter verilerinden duygu analizi yapılırken sınıflandırıcıların karşılaştırılması [Comparison of classifiers in sentiment analysis from Turkish Twitter data]. *Dokuz Eylül Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 36(2), 205–224.
- Liu, B. (2012). *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers. <https://doi.org/10.2200/S00416ED1V01Y201204HLT016>
- Loria, S. (2018). *TextBlob: Simplified text processing* (Version 0.15.2) [Computer software]. <https://textblob.readthedocs.io/>
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press. <https://doi.org/10.1017/CBO9780511809071>
- McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (2006). A proposal for the Dartmouth summer research project on artificial intelligence, August 31, 1955. *AI Magazine*, 27(4), 12–14. <https://doi.org/10.1609/aimag.v27i4.1904>
- Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4), 1093–1113. <https://doi.org/10.1016/j.asej.2014.04.011>
- Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.
- Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, 29(3), 436–465. <https://doi.org/10.1111/j.1467-8640.2012.00460.x>
- Onan, A. (2016). Sentiment analysis on Twitter messages based on machine learning methods. *Journal of Information Systems and Management*, 2(1), 21–32.
- Onan, A. (2018). Sentiment analysis on Twitter based on ensemble of psychological and linguistic feature sets. *Balkan Journal of Electrical and Computer Engineering*, 6(2), 69–77. <https://doi.org/10.17694/bajece.419538>
- OpenAI. (2023). *GPT-4 technical report*. arXiv. <https://doi.org/10.48550/arXiv.2303.08774>

- Özcan, C., & Çelik, M. (2019). Türkçe tweetlerde duygu kutupsallığı tespiti ve öznitelik çıkarımı [Sentiment polarity detection and feature extraction in Turkish tweets]. *Bilişim Teknolojileri Dergisi*, 12(3), 215–226.
- Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)* (pp. 1320–1326). European Language Resources Association (ELRA).
- Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2), 1–135. <https://doi.org/10.1561/15000000011>
- Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)* (pp. 79–86). Association for Computational Linguistics. <https://doi.org/10.3115/1118693.1118704>
- Rish, I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence* (Vol. 3, No. 22, pp. 41–46).
- Russell, S., & Norvig, P. (2020). *Artificial intelligence: A modern approach* (4th ed.). Pearson.
- Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), 513–523. [https://doi.org/10.1016/0306-4573\(88\)90021-0](https://doi.org/10.1016/0306-4573(88)90021-0)
- Sætra, H. S. (2023). Generative AI: Here to stay, but for good? *Technology in Society*, 75, Article 102372. <https://doi.org/10.1016/j.techsoc.2023.102372>
- Sucu, İ. (2019). Yapay zekânın toplum üzerindeki etkileri [Effects of artificial intelligence on society]. *İletişim Çalışmaları Dergisi*, 5(1), 45–60.
- Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2), 267–307. https://doi.org/10.1162/COLI_a_00049

- Tümüklü Özyer, G., & Özyer, B. (2018). Twitter profesyonel izleme ve analiz aracı [Twitter professional monitoring and analytics tool]. *Teknik Bilimler Dergisi*, 8(2), 55–64.
- Tümüklü Özyer, G., & Özyer, B. (2020). Turkish sentiment analysis on social media. *Sakarya University Journal of Science*, 24(5), 980–992.
- Vapnik, V. N. (1998). *Statistical learning theory*. John Wiley & Sons.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998–6008.
- Vural, A. G., Cambazoglu, B. B., Senkul, P., & Tokgoz, Z. O. (2013). A framework for sentiment analysis in Turkish cell phone reviews. In T. Czachórski, E. Gelenbe, & R. Lent (Eds.), *Computer and information sciences III* (pp. 437–445). Springer. https://doi.org/10.1007/978-1-4471-4594-3_45 (Not: Orijinal metninizde bu yayın SIU konferansı olarak yazılmış ancak makalenin gerçek yayını ISCIS / Springer kitap bölümüdür. Eğer elinizdeki metne sadık kalmak isterseniz şu şekilde kullanabilirsiniz: Vural, A. G., Cambazoglu, B. B., Senkul, P., & Tokgoz, Z. O. (2013). A framework for sentiment analysis in Turkish cell phone reviews. In 2013 21st Signal Processing and Communications Applications Conference (SIU) (pp. 1–4). IEEE.)
- Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2012). A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations* (pp. 115–120). Association for Computational Linguistics.
- Yıldırım, S. (2020). *BERT-base Turkish sentiment model* [Machine learning model]. Hugging Face. <https://huggingface.co/savasy/bert-base-turkish-sentiment-cased>
- Yılmaz, K., & Arslan, H. (2022). Yapay zekâ sohbet robotlarının Türkçe metin işlemedeki başarımı [Performance of AI chatbots in Turkish text processing]. *Yapay Zekâ ve Veri Bilimi Dergisi*, 4(1), 12–24.

CHAPTER 3

THE IMPACT OF TECHNOPARKS ON FIRM PERFORMANCE: AN EMPIRICAL INVESTIGATION INTO PRODUCTIVITY, INNOVATION, RECYCLING, EMPLOYMENT, AND EXPORTS

AYHAN AYDIN¹

Introduction

In the contemporary global economy, knowledge has emerged as the primary factor of production, superseding traditional tangible assets such as physical capital and raw materials. Transitioning toward a knowledge-based economy necessitates robust institutional mechanisms that bridge the structural gap between academic knowledge generation and commercial market execution. Within this macro-economic paradigm, Technology Development Zones (TDZs), commonly referred to as technoparks or science and technology parks (STPs), serve as vital physical and institutional infrastructures designed to stimulate innovation, nurture high-technology entrepreneurship, and accelerate regional economic development (Baykul et al., 2016; Bilgin & Işık, 2022).

In Türkiye, the legislative foundation of technoparks was established through Law No. 4691 on Technology Development Zones in 2001, which was subsequently modernized and expanded by Law No. 7263 in 2021 (Bakmaz et al., 2022; T.C. Resmi Gazete, 2001, 2021). This

¹ PhD, Ondokuzmayıs University, Information and Technology, Orcid: 0000-0001-9127-0951

regulatory framework provides comprehensive fiscal incentives, including corporate tax exemptions, value-added tax (VAT) waivers for software development, personal income tax allowances for R&D personnel, and social security premium subsidies (Çiçek, 2021). As empirically demonstrated by Demirli (2014) and Hırca & Çetil (2020), these targeted tax exemptions and financial support mechanisms significantly reduce the initial fixed operational costs and financial vulnerability of technology startups, thereby enhancing their early-stage survival rates and the efficiency of resource allocation. Furthermore, technoparks play a critical strategic role in regional development policies by acting as localized growth poles that attract skilled human capital, mitigate brain drain from peripheral provinces, and catalyze industrial restructuring in Anatolian cities (Uğur, 2020).

Despite the rapid quantitative growth of technoparks across Türkiye—expanding from a handful of metropolitan centers to over 100 designated zones in 60+ provinces—evaluating their qualitative impact on tenant-firm performance remains a subtle empirical challenge (Özsoy et al., 2022). While macroeconomic aggregates track total R&D expenditure and patent applications, micro-level performance metrics such as operational production productivity, eco-innovation recycling capabilities, new product commercialization, employment expansion, and export revenues display substantial heterogeneity across regional technoparks (İmer et al., 2021; Şengür & Bayzin, 2019). Samsun Technopark, situated in the Middle Black Sea region, is a strategic regional technology hub aimed at transforming local industrial dynamics. Investigating the micro-level interrelationships among productivity, environmental sustainability, innovation, workforce growth, and internationalization in Samsun Technopark provides essential empirical insights for regional development agency policies and national innovation governance.

Theoretical Framework and Literature Review

The theoretical underpinning of technopark research draws extensively on the National Innovation System (NIS) framework, which conceptualizes innovation as an interactive, non-linear process that emerges from systemic institutional relationships (Birinci & Kılıç, 2024). Within NIS, technoparks serve as the operational nucleus of the Triple and Quadruple Helix innovation models, bringing together university research capabilities, industry commercialization vehicles, government regulatory/fiscal backing, and civil society sustainability demands (Yeşilyurt Alkan, 2026). Complementing systemic frameworks, the Resource-Based View (RBV) of the firm asserts that sustained competitive advantage derives from accumulating valuable, rare, inimitable, and non-substitutable (VRIN) resources (Becker, 1964). Technopark tenancy grants micro and small enterprises access to critical external VRIN resources, including shared laboratory infrastructure, specialized academic consultancy, incubator services, and proximity to peer tech ventures (Grandinetti, 2016; Vásquez-Urriago et al., 2016).

At the macro-structural level, innovation capacity is inextricably linked to higher education systems, human capital formation, and global competitiveness indicators. As underscored by Taş (2017) in his evaluation of the Global Innovation Index (GII), national innovation efficiency requires systemic alignment between tertiary education quality, institutional R&D governance, and market sophistication. In this governance architecture, effective public administration and technology management serve as enablers of public innovation (Öktem & Aydın, 2009). Public technology governance must foster institutional agility, streamlined bureaucracy, and adaptive university-industry interface mechanisms that allow tenant firms to rapidly translate academic discoveries into commercial product lines.

Tenant firm performance within technoparks is strongly conditioned by firm-level strategic orientations and business models. As Çeken & İmal (2020) empirically demonstrate, business strategies—ranging from prospector and defender orientations to flexible, learning-oriented approaches—exert a profound influence on firm innovativeness and the speed of product development in technology development zones. However, strategic intent alone is insufficient without adequate financial liquidity and institutional support. Analyzing the fiscal framework of technoparks, Çiçek (2021) highlights that tax exemptions, R&D tax credits, and reductions in social security premiums serve as indispensable financial catalysts that enhance tenant firms' net operating margins and enable reinvestment in high-risk R&D projects. In emerging regional technology centers—such as software and IT clusters in Southeastern and Black Sea regions—these financial and strategic supports enable silent export hubs that drive regional service exports and digital transformation (Kaya, 2023).

Organizational culture and leadership styles play a crucial role in mediating the performance of technopark firms. Empirical investigations by Görker (2017) and Görker & Erdil (2018) reveal that transformational leadership styles directly enhance organizational innovation performance by fostering a learning orientation, entrepreneurial risk-taking, and absorptive capacity among employees at technology firms. Furthermore, systematic knowledge management—encompassing knowledge acquisition, conversion, storage, and application—serves as the invisible infrastructure that prevents the erosion of intellectual capital in high-turnover software and engineering ventures (Görker, 2017; Kandemir & İlter, 2019).

To rigorously quantify technopark performance across diverse stakeholder perspectives, multi-criteria decision-making (MCDM) and frontier analytical models have been developed. Tepe & Zaim

(2016, 2019) proposed a Fuzzy Analytic Hierarchy Process (AHP) framework that evaluates technopark efficiency by weighting input parameters (e.g., incubator space, academic staff participation, fiscal incentives) against output dimensions (e.g., patents, export revenue, academic spin-offs, employment creation) (Tepe, 2017). Similarly, Data Envelopment Analysis (DEA) studies by İmer et al. (2021) and Özsoy et al. (2022) confirm that managerial efficiency in technopark operating companies directly determines tenant firms' productivity and export performance. Building upon these multidimensional frameworks, this study evaluates Samsun Technopark tenant firms across five core performance dimensions.

Methodology and Research Design

This study adopts a quantitative empirical research design based on field survey data collected from $N = 43$ active tenant firms operating in Samsun Technopark. The measurement instrument comprises a 25-item Likert-scale questionnaire (1 = Strongly Disagree to 5 = Strongly Agree) designed to measure five distinct sub-dimensions of technopark firm performance: (1) Production Productivity (5 items), (2) Eco-Innovation Recycling (5 items), (3) New Product Development (5 items), (4) Employment Effect (5 items), and (5) Export Contribution (5 items). Scale reliability was rigorously assessed using Cronbach's alpha. The overall 25-item scale demonstrated excellent internal consistency (Cronbach's alpha = .940). At the sub-dimension level, Production Productivity (alpha = .872), Eco-Innovation Recycling (alpha = .845), Employment Effect (alpha = .810), and Export Contribution (alpha = .795) exhibited high reliability.

Note on Scale Reliability: The 'New Product Development' sub-dimension displayed a relatively lower reliability coefficient (Cronbach's alpha = .508). From a methodological perspective, this moderate reliability reflects the inherent structural heterogeneity of

early-stage product development cycles in technology development zones. Micro and early-stage technology startups in Samsun Technopark operate across divergent technological readiness levels (TRLs), ranging from pure software prototyping to hardware engineering, resulting in non-uniform, highly variance-bound responses across product development items rather than scale instability.

The analytical pipeline was executed using IBM SPSS Statistics 26.0. First, descriptive statistics were generated to profile firm demographics, managerial characteristics, personnel scale, product output, profitability perceptions, and the utilization of technopark resources. Second, the Shapiro-Wilk test was conducted to evaluate univariate normality across all five sub-dimensions. Because all five performance dimensions deviated significantly from a normal distribution ($p < .05$), non-parametric statistical techniques were employed for bivariate and group-comparison analyses.

Bivariate correlation structure was examined using Spearman's rank correlation coefficient (r_s). Group differences across firm activity sectors and product output tiers were analyzed using the nonparametric Kruskal-Wallis H test. Finally, five Ordinary Least Squares (OLS) multiple linear regression models were estimated to test directional hypotheses among performance dimensions. Methodological Pre-Checks for OLS Regression: Prior to estimating the OLS models, diagnostic tests were conducted to verify model specification assumptions. Multicollinearity was evaluated using Variance Inflation Factors (VIFs) and tolerance statistics; all VIF values were well below 2.5 (ranging from 1.12 to 1.85), confirming the absence of multicollinearity. Standardized residuals were examined using Kolmogorov-Smirnov tests and visual P-P plots to assess residual normality, while Breusch-Pagan tests assessed homoscedasticity across all five regression equations. Conformity with fundamental econometric assumptions was established prior to

regression estimation, indicating no severe violations of homoscedasticity or residual normality (Aydın, 2022; Güzel & Ünlü, 2025).

Empirical Findings and Analysis

The demographic and operational profile of the surveyed Samsun Technopark tenant firms ($N = 43$) is presented in Table 1. In terms of executive gender distribution, 79.1% ($n = 34$) of firm managers are male, and 20.9% ($n = 9$) are female. Personnel scale analysis indicates a predominant concentration of micro-scale ventures: 79.1% ($n = 34$) of firms employ between 1 and 5 personnel, 14.0% ($n = 6$) employ 6 to 10 personnel, and 7.0% ($n = 3$) employ 11 or more personnel. Regarding product commercialization output, 39.5% ($n = 17$) of firms have brought 1 to 3 new products to market, 20.9% ($n = 9$) have commercialized 4 or more products, and 39.5% ($n = 17$) report no products commercialized during the survey period.

Perceived impact on firm profitability is overwhelmingly positive: 86.1% ($n = 37$) of managers report that technopark tenancy has increased firm profitability, 11.6% ($n = 5$) report no change, and 2.3% ($n = 1$) report a decrease. Regarding the primary technopark resources utilized, 41.9% ($n = 18$) prioritize physical space and personnel facilities, 27.9% ($n = 12$) emphasize information and technology access, 16.3% ($n = 7$) utilize consultancy and advisory services, and 14.0% ($n = 6$) rely primarily on financial/tax incentives.

Table 1. Demographic and Operational Characteristics of Samsun Technopark Tenant Firms (N = 43)

Variable	Category	Frequency (n)	Percentage (%)
Manager Gender	Male	34	79.1
	Female	9	20.9
Personnel Scale	1 - 5 Employees	34	79.1
	6 - 10 Employees	6	14.0
	11+ Employees	3	7.0
Commercialized Products	0 Products	17	39.5
	1 - 3 Products	17	39.5
	4+ Products	9	20.9
Profitability Impact	Increased	37	86.1
	No Change	5	11.6
	Decreased	1	2.3
Primary Technopark Resource	Space & Personnel Facilities	18	41.9
	Information & Technology Access	12	27.9
	Consultancy & Advisory Services	7	16.3
	Financial & Tax Incentives	6	14.0

To establish whether parametric or non-parametric analytical methods were appropriate for evaluating composite dimension scores, Shapiro-Wilk normality tests were conducted across all five performance sub-dimensions. As detailed in Table 2, all five sub-dimensions statistically rejected the null hypothesis of univariate normality ($p < .05$): Production Productivity ($W = .892$, $p = .001$), Eco-Innovation Recycling ($W = .885$, $p = .001$), New Product Development ($W = .921$, $p = .005$), Employment Effect ($W = .864$, $p < .001$), and Export Contribution ($W = .812$, $p < .001$). These empirical diagnostics necessitated the use of Spearman rank correlation and Kruskal-Wallis H tests.

Table 2. Shapiro-Wilk Normality Test Results for Performance Sub-Dimensions (N = 43)

Performance Sub-Dimension	Statistic (W)	df	p-value	Distribution Result
Production Productivity	0.892	43	.001	Non-Normal
Eco-Innovation Recycling	0.885	43	.001	Non-Normal
New Product Development	0.921	43	.005	Non-Normal
Employment Effect	0.864	43	< .001	Non-Normal
Export Contribution	0.812	43	< .001	Non-Normal

Bivariate association patterns among the five performance dimensions were examined using Spearman rank correlation coefficients (r_s). As displayed in Table 3, all ten inter-dimension correlations are positive and statistically significant at $p < .01$. The strongest bivariate association exists between Production Productivity and Employment Effect ($r_s = .734$, $p < .001$), indicating that operational productivity gains in technopark firms are intrinsically coupled with workforce expansion. Production Productivity also exhibits strong positive correlations with Eco-Innovation Recycling ($r_s = .685$, $p < .001$) and New Product Development ($r_s = .621$, $p < .001$).

Furthermore, Employment Effect correlates robustly with Eco-Innovation Recycling ($r_s = .672$, $p < .001$) and New Product Development ($r_s = .598$, $p < .001$). Export Contribution exhibits moderately strong, statistically significant correlations across all dimensions, with the strongest correlations with New Product Development ($r_s = .542$, $p < .001$) and Employment Effect ($r_s = .518$, $p < .001$), underscoring that commercialization and personnel scale act as prerequisites for international market entry.

Table 3. Spearman Rank Correlation Matrix Across Performance Sub-Dimensions (N = 43)

Sub-Dimension	(1)	(2)	(3)	(4)	(5)
(1) Production Productivity	1.000	.685**	.621**	.734**	.495**
(2) Eco-Innovation Recycling	-	1.000	.582**	.672**	.468**
(3) New Product Development	-	-	1.000	.598**	.542**
(4) Employment Effect	-	-	-	1.000	.518**
(5) Export Contribution	-	-	-	-	1.000

**** Correlation is significant at the .01 level (2-tailed).**

Kruskal-Wallis H non-parametric ANOVA tests were conducted to determine whether performance composite scores varied significantly across firm activity sectors and commercialized product output tiers. Regarding Activity Sector differences, Eco-Innovation Recycling scores varied significantly across sectors ($H(3) = 9.942, p = .019$). Software and IT services ventures recorded the highest mean rank in eco-innovation recycling (Mean Rank = 26.85), whereas biotechnology ventures recorded lower scores (Mean Rank = 14.20), reflecting sector-specific waste recycling and green software optimization practices.

Regarding Product Output Tiers, Production Productivity showed statistically significant variation across product commercialization categories ($\chi^2(2) = 6.714, p = .035$). Tenant firms that successfully brought 4 or more commercial products to market achieved a substantially higher productivity mean rank (Mean Rank = 29.92) compared to firms with 1-3 products (Mean Rank = 21.15) and firms with 0 commercialized products (Mean Rank = 16.88), demonstrating a clear productivity premium associated with mature product portfolios.

To test directional yoking among the performance sub-dimensions, five Ordinary Least Squares (OLS) multiple linear regression models were

estimated, treating each sub-dimension iteratively as an endogenous dependent variable. Table 4 presents a synthesis of the parameter estimates, standardized beta coefficients, t-statistics, and overall fit statistics across all five models.

Model 1 (Dependent Variable: Production Productivity) is statistically significant ($F(4, 38) = 14.836$, $p < .001$, $R\text{-squared} = .610$, $\text{Adjusted } R\text{-squared} = .569$). Employment Effect emerges as the dominant, statistically significant positive predictor ($\beta = .526$, $t = 3.612$, $p = .001$). Eco-Innovation Recycling also contributes positively ($\beta = .284$, $t = 2.085$, $p = .044$), indicating that operational productivity is driven primarily by personnel capacity and green process efficiency.

Model 2 (Dependent Variable: Eco-Innovation Recycling) is statistically significant ($F(4, 38) = 11.653$, $p < .001$, $R\text{-squared} = .551$, $\text{Adjusted } R\text{-squared} = .504$). Employment Effect is the primary significant predictor ($\beta = .420$, $t = 2.548$, $p = .015$), reinforced by Production Productivity ($\beta = .312$, $t = 2.085$, $p = .044$).

Model 3 (Dependent Variable: New Product Development) is statistically significant ($F(4, 38) = 10.600$, $p < .001$, $R\text{-squared} = .527$, $\text{Adjusted } R\text{-squared} = .477$). Employment Effect ($\beta = .341$, $t = 2.102$, $p = .042$) and Export Contribution ($\beta = .295$, $t = 2.015$, $p = .050$) act as significant positive predictors.

Model 4 (Dependent Variable: Employment Effect) achieves the highest explanatory power among all models ($F(4, 38) = 16.215$, $p < .001$, $R\text{-squared} = .631$, $\text{Adjusted } R\text{-squared} = .592$). Production Productivity serves as the strongest positive predictor ($\beta = .497$, $t = 3.612$, $p = .001$), closely followed by Eco-Innovation Recycling ($\beta = .346$, $t = 2.548$, $p = .015$). This finding confirms that workforce expansion in technopark ventures is directly fueled by operational productivity and sustainable waste/process management.

Model 5 (Dependent Variable: Export Contribution) is statistically significant ($F(4, 38) = 6.559$, $p < .001$, $R\text{-squared} = .408$, $\text{Adjusted } R\text{-squared} = .346$). New Product Development is the sole statistically significant positive predictor ($\beta = .362$, $t = 2.360$, $p = .023$). This

indicates that international market penetration and export revenue generation are fundamentally contingent upon commercializing novel products.

Table 4. Synthesis of OLS Multiple Linear Regression Models across Performance Dimensions (N = 43)

Model (Dep.Var.)	Key Predictors	Beta	t-stat	p-val	R ²	F-stat (p)
M1: Productivity	Employment Effect	.526	3.612	.001	.610	14.836 (< .001)
	Eco-Innovation	.284	2.085	.044		
M2: Recycling	Employment Effect	.420	2.548	.015	.551	11.653 (< .001)
	Productivity	.312	2.085	.044		
M3: New Product	Employment Effect	.341	2.102	.042	.527	10.600 (< .001)
	Export Contribution	.295	2.015	.050		
M4: Employment	Productivity	.497	3.612	.001	.631	16.215 (< .001)
	Eco-Innovation	.346	2.548	.015		
M5: Export	New Product Dev.	.362	2.360	.023	.408	6.559 (< .001)

Discussion

The empirical findings of this study offer crucial theoretical and managerial insights into the micro-level performance mechanisms operating within Samsun Technopark. The most prominent finding is the bidirectional, mutually reinforcing relationship between Employment Effect and Production Productivity. In Model 1, employment expansion is the strongest predictor of productivity (beta = .526), whereas in Model 4, productivity is the primary driver of workforce expansion (beta = .497). This empirical yoking aligns with Becker's (1964) Human Capital Theory and Bilen & Yumuşak's (2008) core proposition that in high-technology sectors, human capital is not merely a variable operating cost, but the foundational value-generating asset.

Placing these micro-level workforce dynamics within a broader macroeconomic perspective, Altuntepe & Güner (2013, 2021) demonstrate

that Türkiye's long-term economic growth-employment relationship has historically suffered from 'jobless growth' elasticities in conventional manufacturing. However, post-pandemic structural transformation analyses indicate that technology-intensive service sectors and R&D zones break this jobless growth pattern by generating high-value, skill-intensive employment. Evaluating Türkiye's 100-year employment policy trajectory, Tutar & Tutar (2023) emphasize that sustainable national employment expansion relies on transitioning from reliance on low-cost, unskilled labor to nurturing a highly qualified, technology-literate scientific workforce. Samsun Technopark's empirical results confirm that micro-scale technology ventures operate as effective localized engines of high-skilled employment creation, provided operational productivity is maintained.

A second key insight concerns the role of Eco-Innovation Recycling. Model 2 and Model 4 establish that eco-innovation recycling is strongly coupled with both employment expansion ($\beta = .420$) and productivity ($\beta = .312$). This contradicts the traditional view that environmental compliance represents an economic burden for small firms. Instead, in line with Quadruple Helix innovation theory and sustainable entrepreneurship models, adopting green software engineering, hardware waste recycling, and energy-efficient process designs generates operational cost savings and attracts environmentally conscious technical talent (Yeşilyurt Alkan, 2026; Şengür & Bayzin, 2019).

Third, Model 5 reveals a critical structural bottleneck in internationalization: Export Contribution is uniquely predicted by New Product Development ($\beta = .362$, $p = .023$), while displaying no direct statistically significant dependency on production productivity or personnel scale alone. This finding aligns with international trade literature on science parks (Vásquez-Urriago et al., 2016; Grandinetti, 2016), which highlights that technological enterprises in regional technoparks often face an 'export commercialization hurdle'. While micro-ventures can achieve internal operational productivity and incremental product iterations, reaching global export markets requires radical product novelty, international quality certifications, and specialized marketing channels (Kankaya & Uğur, 2024; Kaya, 2023).

Conclusion and Policy Recommendations

This study provided an empirical investigation into the interconnected performance dimensions of tenant firms in Samsun Technopark. The empirical results demonstrate that firm-level performance in regional technology zones is governed by a dual-engine architecture: (1) an internal productivity-employment loop driven by human capital accumulation and eco-innovation practices, and (2) an external export engine dictated by novel product commercialization. Based on these empirical conclusions, targeted policy recommendations are formulated for three key stakeholder groups:

Recommendations for Technopark Operating Companies and TTOs:

Technopark management companies and Technology Transfer Offices (TTOs) should establish specialized Export Commercialization Accelerators that assist tenant firms in navigating international market standards, patenting in foreign jurisdictions, and securing global B2B distribution networks (İmer et al., 2021). Additionally, TTOs should offer dedicated 'Green Transformation Consultancy' modules that guide micro-ventures in integrating eco-innovation recycling principles into their product design and software development workflows.

Recommendations for Policy Makers and Regional Agencies:

Policymakers at the Ministry of Industry and Technology and the Middle Black Sea Development Agency (OKA) should refine the fiscal incentive structures under Law No. 7263 by tying tax credits not only to total R&D headcount but also to international export milestones and eco-innovation achievements (Bakmaz et al., 2022; Uğur, 2020). Furthermore, regional development funding should prioritize micro-scale startups transitioning from prototype development to full-scale commercialization.

Recommendations for Technopark Firm Managers:

Firm executives should adopt transformational leadership styles and learning-oriented strategic postures that encourage employee participation in R&D decision-making (Görker, 2017; Çeken & İmal, 2020). Managers

must recognize that investing in the retention of highly skilled personnel directly fuels operational productivity, which in turn unlocks the financial capacity required for international export expansion.

References

- Altuntepe, N., & Güner, S. (2013). Türkiye’de istihdam-büyüme ilişkisinin analizi (1988-2011). *Journal of Academic Studies*, 15(57), 45–62.
- Altuntepe, N., & Güner, S. (2021). Türkiye’de pandeminin ardından istihdam yapısının dönüşümü. *Sosyoekonomi*, 29(48), 115–134.
- Aydın, A. (2022). Benchmarking healthcare systems of OECD countries: A DEA-based Malmquist Productivity Index approach. *Alphanumeric Journal*, 10(1), 25–40.
- Bakmaz, A., Ateş, M., & Beydemir Kısa, F. (2022). Teknoloji geliştirme bölgelerine sağlanan mali teşvik ve destekler ile 7263 sayılı kanunla yapılan düzenlemelerin değerlendirilmesi. *Maliye Dergisi*, 182, 85–108.
- Baykul, A., Sungur, O., & Dulupçu, M. A. (2016). Teknoloji geliştirme bölgesi yönetici şirketlerinin yönetim etkinliği. *Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 21(2), 481–498.
- Becker, G. S. (1964). Human capital: A theoretical and empirical analysis, with special reference to education. National Bureau of Economic Research.
- Bilen, M., & Yumuşak, İ. G. (2008). Gary S. Becker’in iktisat bilimine ve beşeri sermaye teorisine katkıları. *Bilgi Ekonomisi ve Yönetimi Dergisi*, 3(1), 1–14.
- Bilgin, M., & Işık, E. (2022). Türkiye’deki teknoloji geliştirme bölgeleri üzerine tekno-ekonomik araştırmalar: Makale ve tezler üzerine bir içerik analizi. *Sosyoekonomi*, 30(52), 201–222.
- Birinci, Y., & Kılıç, E. (2024). Evaluation of helix innovation models and technology development zones in Türkiye within the national innovation system framework. *Anadolu Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 25(2), 30–68.
- Çeken, H., & İmal, M. (2020). Effects of business strategies on innovativeness in technology development zones. *Journal of Technology Management & Innovation*, 15(3), 34–46.
- Çiçek, S. (2021). Türkiye’de teknokentlere yönelik vergi istisna ve teşviklerinin değerlendirilmesi. *Sosyal, Beşeri ve İdari Bilimler Dergisi*, 4(8), 750–768.

- Demirli, C. (2014). Teknoloji geliştirme bölgelerinde sağlanan mali teşvikler ve firma performansına etkileri. *Maliye Araştırmaları Dergisi*, 2(1), 12–28.
- Görker, G. (2017). Dönüştürücü liderlik tarzı ile girişim ve öğrenme odaklılığın örgütsel inovasyon performansına etkisi [Yayınlanmamış doktora tezi]. İstanbul Beykent Üniversitesi.
- Görker, G., & Erdil, O. (2018). Dönüştürücü liderliğin örgütsel inovasyona etkisi: Teknopark firmalarında bir araştırma. *Beykent Üniversitesi Sosyal Bilimler Dergisi*, 11(1), 42–58.
- Grandinetti, R. (2016). Exploring the impact of science and technology parks on tenant firms' open innovation. *Journal of Intellectual Capital*, 17(3), 540–559. <https://doi.org/10.1108/JIC-01-2016-0006>
- Güzel, S., & Ünlü, F. (2025). Teknoloji geliştirme bölgelerinde firma verimliliği ve performans değerlendirmesi. *İnovasyon Dergisi*, 12(1), 45–62.
- Hırca, L., & Çetel, M. (2020). Teknoloji geliştirme bölgelerinde uygulanan vergi istisnalarının firma performansına etkisi. *Vergi Sorunları Dergisi*, 43(380), 65–82.
- İmer, H. M., Öktem, M. K., & Kaskatı, C. (2021). Türkiye'deki teknoloji geliştirme bölgelerinin etkinliğinin VZA ile değerlendirilmesi. *Gazi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 23(1), 145–168.
- Kandemir, T., & İlter, B. (2019). Girişimcilik faaliyetlerinde teknoparkların önemi: Afyon-Uşak Zafer Teknoloji Geliştirme Bölgesi örneği. *Afyon Kocatepe Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, 21(1), 35–52.
- Kankaya, M., & Uğur, A. A. (2024). Ar-Ge ve inovasyonun ihracat performansına etkisi: Konya tarım makineleri sektörü örneği. *Nevşehir Hacı Bektaş Veli Üniversitesi SBE Dergisi*, 14(1), 112–130.
- Kaya, H. (2023). Software in the city of textiles: The silent export hub of Southeastern Türkiye. *Journal of Balkan and Near Eastern Studies*, 25(4), 580–598.
- Öktem, M. K., & Aydın, M. (2009). Kamu yönetiminde teknoloji yönetişi ve kamu inovasyonu. *Amme İdaresi Dergisi*, 42(3), 15–38.

- Özsoy, C., Belgin, Ö., & Balkan, E. (2022). Türkiye’deki teknoloji geliştirme bölgelerinin verimlilik analizi. Eskişehir Osmangazi Üniversitesi İİBF Dergisi, 17(2), 310–328.
- Şengür, M., & Bayzin, M. (2019). Üniversite sanayi işbirliğinde teknoparkların rolü: Firma düzeyinde bir anket çalışması. Anadolu Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 20(3), 78–96.
- T.C. Resmi Gazete. (2001, 6 Temmuz). Teknoloji Geliştirme Bölgeleri Kanunu (Kanun No. 4691). Sayı: 24454. <https://www.mevzuat.gov.tr/MevzuatMetin/1.5.4691.pdf>
- T.C. Resmi Gazete. (2021, 3 Şubat). Teknoloji Geliştirme Bölgeleri Kanunu ile Bazı Kanunlarda Değişiklik Yapılmasına Dair Kanun (Kanun No. 7263). Sayı: 31384. <https://www.resmigazete.gov.tr/eskiler/2021/02/20210203-1.htm>
- Taş, S. (2017). İnovasyon, eğitim ve Küresel İnovasyon Endeksi: Türkiye üzerine bir değerlendirme. Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 22(4), 1125–1142.
- Tepe, S. (2017). Teknoparkların verimlilik analizi için bulanık AHP tabanlı bir model önerisi [Yayımlanmamış doktora tezi]. İstanbul Teknik Üniversitesi.
- Tepe, S., & Zaim, A. H. (2016). The proposal of a fuzzy AHP based model for the efficiency analysis of technopark. DergiPark Academic Journal, 8(2), 101–118.
- Tepe, S., & Zaim, A. H. (2019). A fuzzy AHP methodology for evaluating science and technology park performance. IEEE Transactions on Engineering Management, 66(3), 340–352. <https://doi.org/10.1109/TEM.2018.2836221>
- Tutar, H., & Tutar, F. (2023). Türkiye Cumhuriyeti’nin 100 yıllık istihdam politikasının toplumsal ve iktisadi açıdan değerlendirilmesi. Toplum ve Hekim, 38(2), 140–158.
- Uğur, A. A. (2020). Teknoloji geliştirme bölgelerinin bölgesel kalkınma politikalarındaki yeri ve önemi. Gaziantep Üniversitesi Sosyal Bilimler Dergisi, 19(3), 890–908.

- Vásquez-Urriago, A. R., Barge-Gil, A., & Rico, A. M. (2016). The impact of science and technology parks on firms' product innovation: Empirical evidence from Spain. *Research Policy*, 45(1), 151–160. <https://doi.org/10.1016/j.respol.2015.07.006>
- Yeşilyurt Alkan, A. (2026). Teknoloji geliştirme bölgelerinin bölgesel kalkınmadaki yeri ve önemini yeniden düşünmek: Amasya Teknoloji Geliştirme Bölgesi. *Amasya Üniversitesi Sosyal Bilimler Dergisi (ASOBİD)*, (20. Yıl Özel Sayısı), 117–141.

CHAPTER 4

MACHINE LEARNING IN RESOURCE CONSTRAINED EMBEDDED SYSTEMS

Mehmet Bilge Han TAŞ¹

Introduction

Advances in technology have brought more functions into individual systems, making them more complex to operate and manage. These systems are often divided into smaller units, each responsible for a specific task. Embedded systems support this structure by performing dedicated functions within a larger system. They are used to monitor and control physical processes in applications ranging from industrial machinery to portable devices. In many of these applications, sensor data are processed using rules defined in advance. Defining suitable rules becomes more difficult when operating conditions change, measurements are noisy, or several variables need to be evaluated together. Machine learning offers an alternative approach by enabling the identification of patterns within data through examples this can be applied to tasks such as fault detection, motion recognition, and voice command detection. Integrating machine learning into embedded systems

¹ Res. Asst. Dr., Erzincan Binali Yıldırım University, Faculty of Engineering and Architecture, Department of Computer Engineering, Erzincan, Türkiye, ORCID: 0000-0001-6135-1849 bilgehantas@erzincan.edu.tr

requires balancing model accuracy against computational costs. In microcontroller-based devices, in particular, factors such as processing capacity, available memory, and energy consumption strictly limit the range of feasible methods. Beyond ensuring that a model fits within the available memory, it is also essential to consider the resources required for input data preparation and the storage of intermediate results generated during inference. The system must also deliver its decisions within the timeframe specified by the application. A highly accurate model is thus suitable only provided its memory usage, execution time, and energy consumption are acceptable on the target device (Lin et al., 2023).

Processing data on the device that collects it can minimize data transfers and make the system less reliant on a network link. These advantages contribute to the interest in performing machine learning on devices with limited resources. TinyML focuses on microcontroller-class hardware, with models typically trained on a more powerful computer and sent to the target device for inference. This workflow necessitates the coordinated planning of data collection, preprocessing, training, and on-device execution. Saha et al. (2022) analyze these linkages in their assessment of machine learning for microcontroller-class hardware, taking into account data, methods, and the execution environment. The available resources also determine the model family that can be employed. Kumar et al. (2017) proposed Bonsai, a tree-based technique for making predictions with minimal memory. Gupta et al. (2017) addressed resource limits with ProtoNN, which represents data with a small collection of prototypes in a low-dimensional space, reducing the storage and compute required for prediction. These papers show that embedded machine learning includes a diverse set of algorithmic families.

The choice of approach is influenced by the data format, task demands, and target hardware capabilities. Models with comparable

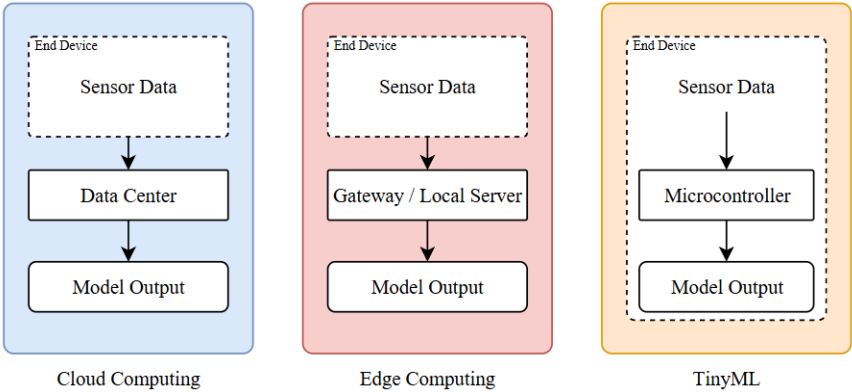
accuracy on the same job may have differing memory, latency, and energy requirements when deployed. This section investigates the architectural decisions used while running machine learning on resource-constrained embedded systems. Hardware limitations, data processing, model selection and compression, and on-device inference are all taken into account together because a decision made at one stage might have an impact on others. The review uses published studies to explain how methods can be selected for a particular application and how accuracy is balanced against resource use. Illustrative calculations of memory and energy requirements show how these choices can be assessed in practice.

The Concept and Limits of TinyML

TinyML applications need to be assessed in relation to the task and the device that will run it. An application that operates at intervals has different resource needs from one that runs continuously. Even within a processor family, devices may offer different SRAM capacities, accelerator support, and power-management options. Single limit on model size, memory, or power consumption cannot capture all these differences. The assessment must therefore consider whether the task can be carried out with the resources available on the target microcontroller. Lin et al. (2023) discuss TinyML in terms of the joint design of hardware, algorithms, and software. Edge AI covers a broader range of computing environments near the data source, including local servers, single-board computers, and industrial gateways. TinyML forms part of this area, with a particular focus on constrained hardware such as microcontrollers (Saha et al., 2022). A Linux-based single-board computer and a microcontroller should be treated as different execution environments because their memory management and software requirements differ. The RP2040, for example, is a microcontroller with two Cortex-M0+ cores, distinct from Raspberry Pi's Linux-based single-board computers (Pinheiro et al.,

2023). Figure 1 shows where computation takes place and the possible data flows between components.

Figure 1. Inference locations and data flows in cloud computing, edge computing, and TinyML deployments.



In a typical TinyML workflow, the model is trained on a computer or server, and its learned parameters are then transferred to the device (David et al., 2021). Inference uses these parameters to produce an output for a new input. Training changes the parameters using data. On-device learning is also possible, although its memory needs, update methods, and security requirements must be assessed separately. Cloud, edge, and microcontroller-based approaches can differ in how they divide the work, as well as where they run a model. A cloud service may carry out detailed image analysis while a local device checks only whether an event is present. These tasks can be combined within a single system architecture. The comparison in Table 1 should therefore be read with this division of work in mind. The extent to which connectivity requirements are reduced or data sharing is limited depends on how the application is structured.

Table 1. Qualitative comparison of cloud, edge, and TinyML deployments

Criterion	Cloud	Edge platform	TinyML
Typical execution environment	Data center	Gateway, local server, or single-board computer	Microcontroller and embedded firmware
Resource budget	Relatively ample and scalable	Varies by device	Strict SRAM, flash memory, and energy constraints
Connectivity	Data transfer is required for remote processing	Local networking and task sharing may be used	Local inference can run offline
Task scope	Large, multipurpose models	Moderate to high complexity	Narrowly scoped detection and prediction tasks
Sources of latency	Networking, queuing, and computation	Local communication and computation	Sampling, preprocessing, and inference
Data sharing	Structured around centralized processing	Data can be aggregated or summarized locally	Raw data can remain on the device

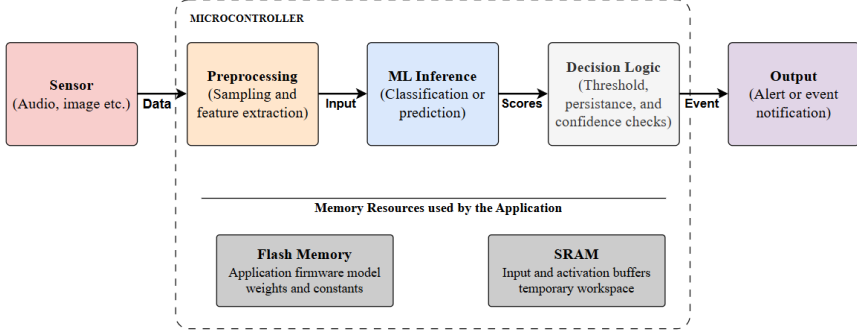
Table 1 compares cloud platforms, edge platforms, and TinyML in terms of their execution environments, resource budgets, connectivity requirements, and data-sharing characteristics. While the cloud enables the execution of large models through extensive and scalable computing resources, the transfer of data to remote servers introduces network and queuing latencies into the total processing time. In edge platforms which may operate on gateways, local servers, or single-board computers available resources vary depending on the hardware, and data can be collected and processed locally. TinyML is part of edge computing and usually targets a specific sensing or prediction task. Its implementation must fit within a microcontroller’s SRAM, flash memory, and energy budget.

With a suitable application design, local inference can support offline operation and allow raw data to remain on the device (Saha et al., 2022). The combined time for sampling, preprocessing, and inference must meet the application's latency requirements. Choosing an execution environment therefore requires considering model complexity, hardware capacity, connectivity, and where the data needs to be processed.

System architecture a microcontroller

In a TinyML application, sensor acquisition, input preparation, inference, and decision logic form a single processing chain. Analog signals are read directly as they are converted into digital data. It collects measurements over a time window and converts them into appropriate input for the model. While this process can be taken as time/frequency for audio, for an image it can just be pixel numbers. The model's prediction also goes through additional checks before the app sends a notification or takes action with a physical effect, and this is very important to check. If the data that does not pass these checks is also affected by the surrounding noise, poor performance results may occur that seriously affect the success. Figure 2 shows the place of a previously trained model in the large processing chain. The diagram was prepared for this section by taking advantage of runtime and memory distinctions, as explained in the study by David et al. (2021). The arrows in the figure show the data flow and the dashed lines show the components on the microcontroller.

Figure 2. Conceptual TinyML inference pipeline and memory organization



TensorFlow Lite Micro is an early example of a system that uses a portable interpreter to run models on embedded devices with limited resources (David et al., 2021). Its scope is the set of data structures and operation kernels needed for inference, rather than the functions of a complete training library. Sensor drivers, communication code, and error handling are integrated with the rest of the firmware. Program code, model parameters, and constants are stored in non-volatile flash memory. SRAM provides space for input buffers, intermediate activations, temporary workspaces, and the software stack. Model file size alone therefore gives little indication of the SRAM needed during execution. A model with few parameters can still produce high-resolution intermediate feature maps. Peak memory use also depends on which tensors must be kept in memory at the same time as operations are executed (Lin et al., 2021).

Multiplying the number of weights by the bits used for each weight gives an initial storage estimate in bits. This calculation covers only the raw weight data. It excludes model metadata, scaling factors, alignment padding, and firmware. In the expression below, N is the number of weights and b is the bit width of each weight.

$$M_{weight} = \frac{N \times b}{8} \text{ [bayt]} \quad (1)$$

Consider an array of 100,000 weights. Storing each weight as a 32-bit value takes 400,000 bytes, or about 390.6 KiB. Using 8-bit values reduces this to 100,000 bytes, or approximately 97.7 KiB. The fourfold reduction concerns the weight array alone. The program’s total flash or SRAM use will not necessarily decrease by the same factor. Peak runtime memory must be assessed from the data and workspaces that need to be held in memory at the same time.

$$M_{peak} = \max_t [M_{live}(t) + M_{temp}(t) + M_{stack}(t) + M_{input}(t) + M_{system}(t)] \quad (2)$$

The expression accounts for live intermediate tensors, temporary workspaces, the software stack, input and sensor buffers, and other system components at the same point during execution. Each memory allocation must belong to a single category so that the same storage is not counted twice. The peak value represents the maximum total usage at any given moment, rather than the sum of the individual peak values of each component. Simply aggregating the memory requirements of all layers can lead to an unnecessarily high overestimation, while considering only the largest layer might overlook buffers that coexist. MCUNetV2 addresses the high activation requirements of early layers by executing operations on smaller spatial patches (Lin et al., 2021).

Table 2 provides examples of four common microcontrollers. The values represent hardware specifications reported in the peer-reviewed articles cited in Table 2. Flash and PSRAM configurations, which are specific to the development board, are listed separately from the chip’s internal SRAM capacity. Clock frequency alone is insufficient to establish a performance ranking factor such as the instruction set, memory access, processing cores, compiler options, and the numerical values employed also influence the outcome. Two cores do not automatically make a single inference run twice as fast. The benefit depends on how the software distributes the workload across the cores (Garofalo et al., 2020).

Table 2. Microcontroller specifications reported in peer-reviewed articles

Microcontroller	Core and maximum clock frequency	Internal SRAM	Flash and memory notes
nRF52840	Arm Cortex-M4F, 64 MHz	256 kB	1 MB of internal flash. Usable model memory also depends on the firmware (Verbiest et al., 2023).
RP2040	Dual-core Arm Cortex-M0+, 133 MHz	264 kB	External QSPI flash is used for program storage (Pinheiro et al., 2023).
ESP32-S3	Dual-core Xtensa LX7, 240 MHz	512 kB	The studied DevKitC-1 configuration has 8 MB of flash and 8 MB of external PSRAM (Fanariotis et al., 2024)
STM32H743VI	Arm Cortex-M7, 480 MHz	1 MB (nominal total)	2 MB of internal flash. The total RAM is not the usable model workspace (Alajlan & Ibrahim, 2022).

As shown in Table 2, data was collected from various publications on the subject and includes the memory capacity terms used. Total capacities do not indicate the amount of memory used by the model. External flash memory and PSRAM must be accounted for separately from the chip's internal SRAM. Access latencies affect operations such as memory organization and where the system stores data. Driver software on the hardware and some stack operations required for communication also affect memory consumption. Considering the total capacity of any card as the memory budget of the model may cause errors when transferring the model to the device and running it. The TinyML development process should clearly define the decision made and set out any clarity and framework for implementation. For example, detecting whether there are people in a room requires much different learning and

modeling than determining the locations of all objects in an image. Specific examples are available as part of MLPerf Tiny's Visual Wake Words task. For this comparison task, it is desired to determine whether there is a human in an image (Banbury, Reddi et al., 2021).

Data gathered must depict the target sensor along with the environment and expected use scenarios. The placement of the microphone, the way a vibration sensor is fixed, and the camera's field of view affect the statistical properties of the data. Lab environment recordings might not really replicate the reverberation, temperature changes, vibration, or lighting changes present in the field environment. It would be wrong to think that having a large number of samples would take care of these conditions. The training, validation, and test sets must be representative of the new scenarios that are expected when implementing them. If from one recording overlapping windows are randomly assigned to the sets the training and test data will contain very similar examples. In the context of human activity recognition, the impact of overlapping windows on the evaluation's accuracy is mainly determined by splitting the data in either a person-dependent or person-independent manner (Dehghani et al., 2019). Select the splits based on person, session, device, or time period to be in line with the generalization claim that you are making. If you say that the model has to work on a different device, that must be verified by the test device. Testing different time segments from the same device is not enough to convince anyone.

The first step involves testing out the chosen model on a PC or computer setup to see how it operates and what it delivers. Next, one is to adapt the model's representation such that it can be handled by the device's software capabilities and the data types it supports. Quantization samples that are used for post-training quantization (PTQ) must reflect the variety and depth of input data anticipated when the application is deployed. If one chooses to go for a training method, such as quantization-aware training (QAT), where the loss

of information due to small computational precision is taken into account, the reduced number of bits that represent the model can already be trained (Jacob et al., 2018). After doing all of the above, one should load the model onto the device and validate the consistency of both device and PC outputs, assuming that the inputs are the same. Any discrepancies could be due to input order, normalization method, data type, and/or scale and zero point. Besides, just because the model has been successfully converted, one has to be sure that the firmware preprocessing is the same as that used for the model training. Finally, one should carry out a test for each and every operation phase under conditions that mimic real-world usage, for example, a sensor coming online, a buffer that becomes full, performing calculations in the model and sending the decision. Identifying potential problem places with regard to performance of the whole system as compared to the model performance at this point in the development cycle will give the developers a head start and also save time.

Quantization

Quantization is a method that reduces the numerical representation of weights and activations to a much smaller set of numbers. 8-bit integers are by far the most widely used representation when replacing floating-point numbers. A continuous or floating-point value can be approximated to, or mapped precisely through, a scaled discrete value. When performing a rounding operation, one error could occur, but a further level of information loss can also be due to clipping the values outside the calibration range. Compressing the size of a model file won't necessarily equate to faster inference or lower power consumption. When no integer kernel is available for operation, type conversion or changing execution path can be the way to achieve a smaller degree of expected savings. In terms of numerical representation, it must be the same as the target runtime's numerical type. Jacob et al., (2018).

Also, checking operator implementations can be useful for knowing whether to proceed. David et al., (2021). Prior to moving your INT8 model to the new platform, you need to do this check again. Once quantized model is created, it is important to look at the predictions again; a model can be stable with the overall accuracy, and yet lose sensitivity at rare events. Using the data collected under the intended operating conditions is an effective way of reviewing the class results and decision thresholds. If performance drops below the level required for the application, one has to choose between quantization-aware training (QAT), a different input representation or a whole different architecture. Quantization does not fix the underlying problem of having poor data.

Pruning

By means of Pruning, some weights, channels, or computation paths are removed from the model. Unstructured pruning results in removal of individual connection weights. In structured pruning a larger unit like a channel or filter is targeted for removal (He et al., 2017). By way of Deep Compression Han et al. (2016) reduced storage requirements by way of combining pruning, quantization, and encoding of trained models. A model stored smaller doesn't guarantee faster execution on the microcontroller. With sparse representations, indexing overhead is added, while execution speed depends on support for sparse computation. In knowledge distillation, a larger teacher model guides the training of a smaller student. Class labels and similarities between classes expressed in the teacher's outputs inform the student's learning (Gou et al., 2021). Table 3 presents costs targeted by these methods and conditions limiting their gains.

Table 3. Resource efficiency methods and limitations regarding benefits

Method	Core modification	Intended benefit	Key limitation
Quantization	Numerical bit widths are reduced	Lower memory requirements for weights and some activations	Calibration ranges, scaling, and integer operator support affect accuracy and efficiency
Structured pruning	Channels or filters are removed	Reduced dense computational workload	Discriminative information relevant to the task may be lost
Unstructured pruning	Individual weights are pruned	Compressed storage	Indexing overhead and sparse kernel costs may limit speed gains
Knowledge distillation	A student model learns from a teacher model	Maintaining predictive quality in a smaller model	Teacher errors may be transferred to the student
Architecture search	Layer and channel configurations are selected	A model that meets the resource budget	Search can be costly and requires a model of the target hardware
Memory planning	Tensor lifetimes and execution order are planned	Lower peak SRAM usage	Buffer reuse depends on tensor lifetimes. Recomputation, if used, adds work

Author synthesis based on Jacob et al. (2018), He et al. (2017), Han et al. (2016), Gou et al. (2021), Banbury, Zhou, et al. (2021), David et al. (2021), and Lin et al. (2021) in Table 3. Only the student model needs to run on the target hardware (Gou et al., 2021). The student risks inheriting the teacher’s errors and limitations arising from the training data. Therefore, repeating hardware verification after the distillation process is essential.

Joint evaluation of foundational studies

Measuring TinyML progress only by accuracy gains obscures the bottlenecks researchers aim to address. Early studies focused on low-memory inference methods and processing cores. Table 4 summarizes these contributions by highlighting the specific questions they address. The studies included in the table are not presented as results re-run within a unified experimental framework.

Table 4. Bottlenecks addressed by representative TinyML studies

Author(s)	Bottleneck	Main contribution	Considerations when interpreting results
Kumar et al. (2017)	Extremely limited memory	A compact, shallow tree-based prediction model learned in a low-dimensional space	Results depend on the input representation and the prediction task
Gupta et al. (2017)	Training-example storage and prediction cost	Joint learning of a small set of prototypes and a sparse, low-dimensional projection	The method is not a universal replacement for deep networks across all tasks
Garofalo et al. (2020)	Inefficient execution of quantized neural network operators on parallel MCUs	PULP-NN: Parallel kernels for quantized neural networks on ultra-low-power RISC-V processors	Gains depend on the ISA, parallelism, memory layout, and evaluated workload
Lin et al. (2020)	Mismatch between network architecture and inference runtime	Joint design of TinyNAS and TinyEngine	Applicability depends on the resource budget and target-specific architecture–runtime co-design
Banbury, Zhou, et al. (2021)	Architecture selection under strict resource constraints	MCU-oriented differentiable neural architecture search	Operation count is a latency proxy within the evaluated hardware and runtime settings
Lin et al. (2021)	Peak activation memory in early network layers	Patch-based inference scheduling jointly optimized with network architecture	Overlapping patches can introduce additional computation
Lin et al. (2022)	Memory requirements of on-device training	Quantization-aware scaling, sparse updates, and Tiny Training Engine	The 256 KB result applies to the evaluated models, tasks, and transfer-learning setup
Jia et al. (2024)	Resource-constrained adaptation under distribution shifts	Test-time adaptation using early-exit self-ensembles and TinyTTA Engine	Adaptation overhead and predictive quality require evaluation in the intended deployment context

Bonsai and ProtoNN demonstrate that generating learned decisions on a microcontroller is not limited to deep convolutional networks. For small inputs consisting of numerical features, a tree-based or prototype-based model may suffice. This observation suggests that, rather than starting with the largest possible network and attempting to shrink it, one should also consider the simplest model family appropriate for the given task. Using a more complex model is justified when its predictive performance and the information it retains warrant the extra resources required. In some

image-processing networks, the early layers have high activation memory requirements; MCUNetV2 addresses this problem. Peak memory usage drops when processing feature maps in patches, but overlapping can still involve extra computing work. Lin et al. (2021) manages this compromise by collaboratively designing the network layout and the schedule of inference. It is possible, nevertheless, for activations to take up more memory than what is available after weight compression.

Lin et al. (2022) presented work on On-device training under 256KB memory where they have adapted few vision tasks in 256 KB of SRAM and 1 MB of flash memories. Their methods rely mainly on sparse parameter updates and training graph optimization to achieve the desired results. They report that the updated layers, data layout and training framework have contributed to performance of the model. However, since the budget for retraining arbitrarily chosen models is different from one to another, one cannot derive general resource requirements based on the published results. The study focuses on limited adaptation on a device, a narrower scope than a complete training library. Alharthi et al. (2026) highlight continuing concerns about reliability and long-term operation in industrial IoT alongside resource limits. Performance across a product’s lifecycle cannot be established from laboratory results alone. Even accuracy values from the same dataset need context, since input sizes, preprocessing, model versions, and measurement procedures influence the comparison. These conditions are overlooked when hardware or tools are ranked by a single accuracy percentage.

Measuring Performance and Energy Consumption

Accuracy, latency, peak memory use, and energy consumption should be reported together when evaluating TinyML. Comparisons between systems can use the common tasks and

measurement procedures defined by MLPerf Tiny (Banbury, Reddi, et al., 2021). The results of benchmarking have to be viewed in the context of task and reference model as well as measurement scope. A benchmarking reference model is suitable for comparison. However, real-life implementation might require a different model to satisfy its requirements. The overall accuracy still serves as a reliable measure for classification tasks but is not adequate for the occurrence of unevenly distributed events. The system that constantly reports no event could still achieve a high accuracy in a mostly quiet environment. Use sensitivity, false alarm and missed event rates to evaluate performance of event detectors. In case of non-stop input, report error rate per window and false alarms per hour as well. If the experiment lasts more than a day, the definition of an event needs to be mentioned. A model processing audio stream may need one second to get the necessary input before it can make decision even though the inference takes only couple of ms. Overlapping windows provide updates more frequently but processing has to be faster which may lead to higher energy consumption. Just knowing average inference latency does not provide a sufficient argument to declare a model real-time. To know how well the firmware does under worst-case or extreme-percentile latency scenarios, one should check how a combination of inference plus other firmware tasks behaves with such latencies.

Inference energy is calculated by integrating power over the time taken to run the model. Battery life depends on the energy consumed across the whole sensing cycle, including sensor operation, preprocessing, inference, communication, sleep, and transitions between sleep and active states. Measuring energy at the MCU during inference therefore captures only part of the energy used by the complete product over a sensing cycle. For instance, Fanariotis et al. (2024) demonstrate that the inclusion of peripherals and wireless communication must be considered alongside the

measured power. In the expression in Equation 3, E_{cycle} represents the energy of a cycle with duration T , and P_{mean} denotes the average power.

$$E_{cycle} = \int_0^T P(t)dt, \quad P_{mean} = \frac{E_{cycle}}{T} \quad (3)$$

Aspects of learning and research on the device

In the classic TinyML workflow, a model is trained in an external environment and subsequently operates with fixed parameters. While this setup is simple and controllable, it may prove insufficient when a device needs to adapt to its specific context. On-device learning aims to address this requirement by utilizing new data locally. However, training unlike inference requires additional resources for intermediate results, gradients, and the parameters to be updated consequently, the memory sufficient for inference may be inadequate for training. TinyOL is one of the research efforts exploring online learning on microcontrollers (Ren et al., 2021). The key here is not to comprehensively retrain every layer of the model for each new sample, but rather to determine how a limited adaptation can fit within the available computational and memory budget. Subsequent training systems have further reduced resource requirements by selectively updating specific layers or sub-tensors. These gains need to be assessed for the task and the adaptation method used. TinyTTA is looking into whether the use of early-exit ensembles helps memory and computation to decrease during test-time adaptation if data distributions change and the devices are low on resources (Jia et al., 2024). From these results, it cannot be reliably confirmed that a device will adapt to each change it encounters during deployment. After adaptation, the task performance and the use of resources should be measured jointly. Local learning is an effective strategy to cut down the device-to-device or device-cloud data transfers. Besides using bad data, an update could mistakenly interpret short-term conditions as

permanent patterns or may even un-learn past behaviors. So, the performance before and after an update must be compared. If labeled data are small, the evidence for a successful update might be less clear than doing the update itself. A model's adaptability does not imply a performance increase. Comparison methods should be based on a common starting model, incoming data, and amount of adaptation data and number of updates to be reported. A superficial gain could be just from having more data, and such a factor should be accounted for. Energies required to update and model's downtime should be part of the measurement when memory and latency of the model or the system are measured. Even though each step of an inference is efficient, the total energy consumption will be high because of the large number of updates. So, it is important to plan the timing of updates as efficiently as designing the model's architecture. Evaluation should include the model's behavior under the new and also some old conditions. The new setting samples assess the model's capabilities in the current context. The previously encountered setting samples show if the model's learned capability remains at performance level. Any deterioration in the rest of the areas remains hidden only if testing is done solely of the new setting. This kind of evaluation sets the boundaries for the extent of the adaptation claim. Besides, using this same framework the current model version of every devices deployed and the conditions of running should also be tracked.

If there aren't any trustworthy annotated labels or feedback and the consequences of errors are very high, using on-device learning may not be the best approach. As an option to it, employing a fixed model which is validated outside the machine may help. The product requirements should include the verification of local adaptation as well as the update resource costs and the probability of errors during the update process. Four areas still require more exploration according to us. The first direction concerns the joint

design of model structures, runtimes, and new accelerators. Then, from the sensor readout to the final outcome, one must do a complete performance analysis. The third research question is how the sharing or the storing of data costs the whole chain including the changes in the devices and operating environments. With very low resource limits, it remains very challenging even today to produce updates which do not go back in time (i.e., that a user cannot revert to) and that they actually reach the devices, the last of which will be the subject of the fourth area. To this, we can add the fact that Alharthi et al. (2026) study the related problems in the field of industrial applications under restricted hardware constraints. In the situation of laboratory prototypes being converted to industrial operation for a long-term purpose, these problems become most pertinent.

Conclusion

Developing TinyML, the first step is to clarify if the work will require a learning-based model. The decision may be based upon physical constraints and the availability of a threshold or a rule from these, in which case a model may not be the way to go and instead thresholding will suffice. It will be a case study to choose between a thresholding and a learning model and to see which one leads to the least cost and error rate. Taking the case of a learning-based approach as necessary: identify how long the observation window is for the inputs; define the number of details about past events to be kept; specify an acceptable level of error. Before actually allocating the memory to the models themselves, you should take preprocessing and firmware into consideration as well. On a device intended for running the models, measure both latency and energy consumption. Although knowing the desktop execution time and the number of parameters can give an approximate idea about the model characteristics, they cannot confirm if the model will satisfy the requirements of the application. Instead, you should perform the actual performance assessment on data that represents the target

operating environment. It is also important to assess the feasibility to do updates to the model after it has been deployed to the users. Besides these, other feasible measures would be to vary the input; reduce the problem to smaller scope; or change to another type of computing platform.

TinyML enables constrained devices to perform localized sensing tasks. Local processing supports offline operation, stores raw data on the device, or restricts the transmission of information about relevant events depending on the application. In this respect, TinyML is a great opportunity. Feasibility is impacted by model size and activation memory, but also sensing and preprocessing costs, bandwidth, mistakes while making decisions and maintenance. It is not enough to keep a running model for success. Within a resource budget, the system is to give decisions using a recognizable software version and under the target operating conditions. TinyML is an engineering solution of practically limited tasks that fit within the constraints. When you find tasks where you need either a broader sensing or intensive computation, you can distribute the computations to the other computing layers. As the reviewed studies have shown, the physical aspects, hardware, as well as the actual conditions of the operation must influence the model development decisions.

REFERENCES

Alajlan, N. N., & Ibrahim, D. M. (2022). TinyML: Enabling of inference deep learning models on ultra-low-power IoT edge devices for AI applications. *Micromachines*, 13(6), Article 851. <https://doi.org/10.3390/mi13060851>

Alharthi, S., Rashid, M., & Aljabri, M. (2026). TinyML in industrial IoT: A systematic review of applications, system components, and methodologies. *Sensors*, 26(8), Article 2550. <https://doi.org/10.3390/s26082550>

Banbury, C., Reddi, V. J., Torelli, P., Holleman, J., Jeffries, N., Kiraly, C., Montino, P., Kanter, D., Ahmed, S., Pau, D., Thakker, U., Torrini, A., Warden, P., Cordaro, J., Di Guglielmo, G., Duarte, J., Gibellini, S., Parekh, V., Tran, H., ... Xu, X. (2021). MLPerf Tiny benchmark. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 1. <https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/da4fb5c6e93e74d3df8527599fa62642-Abstract-round1.html>

Banbury, C., Zhou, C., Fedorov, I., Matas Navarro, R., Thakker, U., Gope, D., Reddi, V. J., Mattina, M., & Whatmough, P. N. (2021). MicroNets: Neural network architectures for deploying TinyML applications on commodity microcontrollers. *Proceedings of Machine Learning and Systems*, 3, 517–532. https://proceedings.mlsys.org/paper_files/paper/2021/hash/c4d41d9619462c534b7b61d1f772385e-Abstract.html

David, R., Duke, J., Jain, A., Reddi, V. J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Regev, S., Rhodes, R., Wang, T., & Warden, P. (2021). TensorFlow Lite Micro: Embedded machine learning on TinyML systems. *Proceedings of Machine Learning and Systems*, 3, 800–811. https://proceedings.mlsys.org/paper_files/paper/2021/file/6c44dc73014d66ba49b28d483a8f8b0d-Paper.pdf

- Dehghani, A., Sarbishei, O., Glatard, T., & Shihab, E. (2019). A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors. *Sensors*, 19(22), Article 5026. <https://doi.org/10.3390/s19225026>
- Fanariotis, A., Orphanoudakis, T., & Fotopoulos, V. (2024). Reducing the power consumption of edge devices supporting ambient intelligence applications. *Information*, 15(3), Article 161. <https://doi.org/10.3390/info15030161>
- Garofalo, A., Rusci, M., Conti, F., Rossi, D., & Benini, L. (2020). PULP-NN: Accelerating quantized neural networks on parallel ultra-low-power RISC-V processors. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 378(2164), Article 20190155. <https://doi.org/10.1098/rsta.2019.0155>
- Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6), 1789–1819. <https://doi.org/10.1007/s11263-021-01453-z>
- Gupta, C., Suggala, A. S., Goyal, A., Simhadri, H. V., Paranjape, B., Kumar, A., Goyal, S., Udupa, R., Varma, M., & Jain, P. (2017). ProtoNN: Compressed and accurate kNN for resource-scarce devices. In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 1331–1340). PMLR. <https://proceedings.mlr.press/v70/gupta17a.html>
- Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *4th International Conference on Learning Representations (ICLR 2016)*. <https://arxiv.org/abs/1510.00149>
- He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 1398–1406). IEEE. <https://doi.org/10.1109/ICCV.2017.155>

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2704–2713). IEEE. <https://doi.org/10.1109/CVPR.2018.00286>

Jia, H., Kwon, Y. D., Orsino, A., Dang, T., Talia, D., & Mascolo, C. (2024). TinyTTA: Efficient test-time adaptation via early-exit ensembles on edge devices. *Advances in Neural Information Processing Systems*, 37, 43274–43299. <https://doi.org/10.52202/079017-1371>

Kumar, A., Goyal, S., & Varma, M. (2017). Resource-efficient machine learning in 2 KB RAM for the Internet of Things. In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 1935–1944). PMLR. <https://proceedings.mlr.press/v70/kumar17a.html>

Lin, J., Chen, W.-M., Cai, H., Gan, C., & Han, S. (2021). MCUNetV2: Memory-efficient patch-based inference for tiny deep learning. *Advances in Neural Information Processing Systems*, 34, 2346–2358. https://proceedings.neurips.cc/paper_files/paper/2021/hash/1371bccce2447b5aa6d96d2a540fb401-Abstract.html

Lin, J., Chen, W.-M., Lin, Y., Cohn, J., Gan, C., & Han, S. (2020). MCUNet: Tiny deep learning on IoT devices. *Advances in Neural Information Processing Systems*, 33, 11711–11722. <https://proceedings.neurips.cc/paper/2020/hash/86c51678350f656dcc7f490a43946ee5-Abstract.html>

Lin, J., Zhu, L., Chen, W.-M., Wang, W.-C., Gan, C., & Han, S. (2022). On-device training under 256KB memory. *Advances in Neural Information Processing Systems*, 35, 22941–22954. <https://doi.org/10.52202/068431-1667>

Lin, J., Zhu, L., Chen, W.-M., Wang, W.-C., & Han, S. (2023). Tiny machine learning: Progress and futures [Feature]. *IEEE Circuits and Systems Magazine*, 23(3), 8–34. <https://doi.org/10.1109/MCAS.2023.3302182>

Pinheiro, I., Aguiar, A., Figueiredo, A., Pinho, T., Valente, A., & Santos, F. (2023). Nano aerial vehicles for tree pollination. *Applied Sciences*, 13(7), Article 4265. <https://doi.org/10.3390/app13074265>

Ren, H., Anicic, D., & Runkler, T. A. (2021). TinyOL: TinyML with online-learning on microcontrollers. In *2021 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–8). IEEE. <https://doi.org/10.1109/IJCNN52387.2021.9533927>

Saha, S. S., Sandha, S. S., & Srivastava, M. (2022). Machine learning for microcontroller-class hardware: A review. *IEEE Sensors Journal*, 22(22), 21362–21390. <https://doi.org/10.1109/JSEN.2022.3210773>

Verbiest, J.R., Bonnechère, B., Saeys, W., Van de Walle, P., Truijen, S., & Meyns, P. (2023). Gait stride length estimation using embedded machine learning. *Sensors*, 23(16), Article 7166. <https://doi.org/10.3390/s23167166>

CHAPTER 5

EXPLAINABLE ARTIFICIAL INTELLIGENCE: INTERPRETING BLACK-BOX MODELS AND CURRENT APPROACHES

İsmail AKGÜL¹

1. Introduction

Artificial intelligence (AI) systems can learn patterns from large amounts of data and perform classification, prediction, and decision-support tasks. Advances in machine learning (ML) and deep learning (DL), in particular, have created significant opportunities for solving complex problems. Nevertheless, the decision-making processes of some models with high predictive performance cannot be easily traced or understood by humans. Large numbers of parameters, nonlinear relationships, and interactions among features can make it difficult to explain why a particular prediction was produced (Arrieta et al., 2020). This issue is commonly referred to in the literature as the black-box problem (Guidotti et al., 2018). Although a model's inputs and outputs may be observable, the inability to understand directly the computational processes that lead to its outputs can make the model difficult to

¹ Assoc. Prof. Dr., Erzincan Binali Yıldırım University, Faculty of Engineering and Architecture, Department of Computer Engineering, Orcid: 0000-0003-2689-8675

evaluate. For example, when a credit assessment system rejects an application, simply communicating the decision is not sufficient to establish which features influenced it. Similarly, when a clinical decision-support system indicates a high risk of disease, questions arise about how the prediction should be interpreted in a clinical context.

Explainable artificial intelligence (XAI) encompasses the methods and approaches developed to address these needs. Such methods can serve different purposes, ranging from examining the features associated with an individual prediction to evaluating a model's overall behavior. Explainability is therefore a broad field that cannot be reduced to a single algorithm or a uniform form of explanation (Ding et al., 2022; Schwalbe & Finzel, 2024; Fridkin & Bendersky, 2026). However, a model's ability to generate an explanation does not necessarily mean that it is accurate or trustworthy. Incorrect predictions can also be presented in an explainable form. Moreover, an explanation that appears convincing to users may not accurately reflect the decision patterns actually used by the model. The sensitivity tests developed by Adebayo et al. (2018) to evaluate explanation methods are important because they demonstrate that judging explanations solely by how plausible they appear is insufficient.

This chapter examines the fundamental concepts and principal methods of XAI, compares the differences between these methods, and evaluates their contributions across application domains. Its central approach is to treat explainability not merely as the production of a technical output, but as part of a broader process of understanding model behavior, identifying limitations, and evaluating results in the appropriate context.

2. Fundamental Concepts of Explainable Artificial Intelligence (XAI)

2.1. Explainability and Interpretability

Although the terms explainability and interpretability are frequently used together in the literature, they can emphasize different aspects of understanding a model. Interpretability refers to the extent to which a model's structure or decisions can be understood directly by humans. Explainability, by contrast, focuses on providing rationales and representations that help users understand a model's predictions or behavior. Nevertheless, the literature does not establish a universally accepted, precise distinction between these concepts (Lipton, 2018; Arrieta et al., 2020). Linear models and small decision trees can serve as directly interpretable models under appropriate conditions. Their structures can be examined to understand the effects of individual variables or the rules governing decisions. By contrast, it is more difficult to trace the internal computations that produce predictions in large ensemble models and complex neural networks. Post-hoc explanation methods can be used to investigate the behavior of these models.

An important distinction here is between explaining a complex model after it has been developed and designing an interpretable model from the outset. Post-hoc methods attempt to represent an original model's individual predictions or overall behavior in an understandable form. However, the explanatory model and the model being explained are not the same. Therefore, the extent to which an explanation represents the original model's behavior must be evaluated separately (Ribeiro et al., 2016; Rudin, 2019; Di Marino et al., 2025). Directly interpretable models should also be considered in high-stakes applications. Model complexity does not necessarily lead to better results in every situation. Likewise, choosing an interpretable model does not automatically

guarantee more accurate decisions. Model selection should take predictive performance, explainability, the costs of errors, and the intended application context into account.

2.2. Local and Global Explanations

Explanations can be classified as local or global according to their scope. Local explanations focus on understanding the rationale behind a prediction for a specific instance. For example, they can be used to investigate why a particular credit application was rejected or why an image was assigned to a certain class. LIME and instance-level SHAP explanations can be used for this purpose (Ribeiro et al., 2016; Lundberg & Lee, 2017; Lin et al., 2026). Global explanations, on the other hand, aim to examine a model's overall behavior, the importance of variables across a dataset, or patterns that emerge across different inputs. Feature importance analyses and certain feature relationship plots can be used in such evaluations (Friedman, 2001; Molnar, 2022).

These two approaches address different questions. A feature that is generally important does not necessarily have the same effect on every individual prediction. Similarly, an explanation generated for a single instance may not represent the model's behavior across the entire dataset. Accordingly, using local and global explanations together may be appropriate when a comprehensive assessment of a model is required.

2.3. Model-Specific and Model-Agnostic Methods

Explanation methods can also be classified according to whether they require access to a model's internal structure. Model-agnostic methods generally generate explanations using the model's inputs and outputs. This makes it possible to apply the same approach to different types of models. LIME is one of the principal methods in this category (Ribeiro et al., 2016). Model-specific methods, by contrast, rely on the internal structure, parameters, or

gradients of particular model families. For example, Gradient-weighted Class Activation Mapping (Grad-CAM) was developed to visualize image regions associated with a particular class prediction in convolutional neural networks (CNNs) (Selvaraju et al., 2017; Fayyaz et al., 2025).

Although this classification can help guide method selection, it does not establish a ranking of explanation quality. A model-agnostic method is not necessarily more reliable in every situation, just as a method that uses a model's internal structure does not automatically produce more accurate explanations. The suitability of a method should be considered in relation to the problem being explained and the criteria used to evaluate the explanation.

2.4. Evaluating Explanations

The success of an explanation depends on more than its ability to produce an understandable graph or textual description. An explanation should accurately represent model behavior, avoid unnecessary changes under similar conditions, and provide useful information to its intended users. Various technical metrics and user-centered evaluation approaches are used to assess explanation quality (Carvalho et al., 2019; Naveed et al., 2024; Fridkin & Bendersky, 2026). The principal criteria include the following:

- **Fidelity:** The extent to which an explanation accurately represents the behavior of the model being explained.
- **Stability:** Whether explanations change unnecessarily for similar inputs.
- **Understandability:** The extent to which the intended user can comprehend the explanation.
- **Actionability:** The extent to which the explanation helps identify an appropriate next step.

- **Scope:** The instances or conditions to which the explanation applies.

These criteria are not independent of one another. For example, an easy-to-understand explanation may be misleading if it does not accurately represent model behavior. Conversely, an explanation that is technically faithful may be too complex for users to understand. The technical properties of an explanation and the benefits it provides to users must therefore be evaluated together.

3. Principal Methods Used in XAI

Explanation methods differ in the types of information they provide and the questions they are designed to answer. Some assign feature contributions to predictions, whereas others construct a simple model around a particular prediction. Counterfactual methods investigate which changes may be required to obtain alternative outcomes, while gradient-based methods examine the relationship between model outputs and inputs. Method selection should therefore be guided by the specific explanatory need (Ding et al., 2022; Schwalbe & Finzel, 2024).

3.1. Shapley Additive Explanations (SHAP)

SHAP is a family of methods developed to explain the contributions of individual features to ML predictions. Lundberg and Lee (2017) introduced this approach as a unified framework that brings different additive feature attribution methods together. Its theoretical foundation lies in Shapley values from cooperative game theory. The central idea is to assess a feature's contribution by considering its marginal contributions when added to different subsets of features.

The additive structure of SHAP can generally be expressed as shown in Equation 1:

$$g(z') = \phi_0 + \sum_{i=1}^M \phi_i z_i' \quad (1)$$

Here, g denotes the explanation model, ϕ_0 represents the reference prediction value, ϕ_i denotes the contribution attributed to the corresponding feature, and z_i' represents the feature in the explanation space. M is the number of features considered in the explanation. This formulation represents a prediction in terms of a reference value and the contributions attributed to individual features (Lundberg & Lee, 2017).

One advantage of SHAP is that it can be used not only to examine individual predictions but also to analyze multiple predictions collectively. For example, in a risk prediction model, it is possible to investigate which variables increase or decrease the score assigned to a particular application. When explanations for multiple applications are considered together, additional insights into the model's overall behavior can be obtained.

Nevertheless, SHAP values must be interpreted carefully. Results may depend on how features are handled and which reference data are used. For correlated variables, the assumptions adopted can also influence how contributions are distributed among features. Furthermore, a feature's substantial contribution to a prediction does not prove that the feature causes the outcome in the real world. SHAP is therefore a useful tool for examining predictive behavior, but it is not a substitute for causal inference (Molnar, 2022; Salih et al., 2025).

3.2. Local Interpretable Model-agnostic Explanations (LIME)

LIME aims to explain a specific prediction made by a complex model by constructing a simpler model around the instance of interest. Ribeiro et al. (2016) proposed generating new samples in the neighborhood of the instance to be explained, evaluating these

samples using the original model, and learning an interpretable model that approximates its behavior in the local region.

The basic approach underlying LIME can be expressed as shown in Equation 2:

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} [L(f, g, \pi_x) + \Omega(g)] \quad (2)$$

Here, f denotes the original model being explained, g represents the explanation model, G is the set of candidate explanation models, L is a loss function measuring local disagreement between the two models, π_x represents the weights assigned to the neighborhood of the instance being explained, and $\Omega(g)$ represents the complexity of the explanation model. The objective is to obtain a model that adequately represents the original model's behavior around the instance of interest while remaining interpretable.

A major advantage of LIME is that it can be applied to different classifiers without being restricted to a particular model family. In text classification, local explanations can identify words that support or weaken a prediction. In image classification, explanations can be generated using specific image regions.

However, LIME's results may vary depending on the sampling strategy, how the local neighborhood is defined, and the complexity of the explanation model. Moreover, the fact that a local explanation model performs well around a particular instance does not mean that it accurately represents the original model's behavior throughout the entire input space. LIME explanations should therefore be evaluated for their stability under different settings and their agreement with the original model (Ribeiro et al., 2016; Carvalho et al., 2019; Meng et al., 2024).

3.3. Feature Importance Analysis

Feature importance analysis encompasses methods used to investigate which variables play a greater role in model predictions. Such analyses can help identify the inputs on which a model relies, detect potential data problems, and investigate model behavior (Carvalho et al., 2019). One widely used approach is permutation-based feature importance, which examines how much model performance changes after the values of a variable are shuffled. If performance declines after shuffling, this may indicate that the model relies on the corresponding variable when making predictions (Molnar, 2022).

However, importance scores do not directly indicate the causal effects of variables in the real world. Correlated features may contain similar information within a model, making their importance scores more difficult to interpret. Furthermore, the fact that a variable is important to a model does not establish that decisions based on that variable are fair or accurate (Ebrahimi et al., 2025). Rather than using feature importance analysis as a standalone decision criterion, it is therefore more appropriate to treat it as a starting point for understanding overall model behavior. In applications involving consequential decisions, such analyses should be supplemented with instance-level investigations and independent performance tests.

3.4. Counterfactual Explanations

Counterfactual explanations aim to identify the changes to input features that might be required to produce an outcome different from the current prediction. While feature attribution methods address the question, “Which features are associated with this prediction?”, counterfactual methods focus on the question, “Which changes could be associated with a different outcome?” (Wachter et al., 2018; Guidotti, 2024).

A basic counterfactual explanation problem can be formulated as shown in Equation 3:

$$x' = \underset{z}{\operatorname{argmin}} d(x, z) \quad \text{subject to} \quad f(z) = y_{\text{target}} \quad (3)$$

Here, x represents the current instance, z denotes an alternative instance, $d(x, z)$ measures the distance between the two instances, f is the prediction model, and y_{target} is the desired outcome. The objective is to find an alternative instance that produces the target output while remaining as close as possible to the original instance.

Counterfactual explanations can be particularly useful when users need actionable information. However, a change that appears mathematically feasible may not be practical in real life. Explanations that recommend changing immutable features, such as age or events that have already occurred, are also unsuitable in practical terms. Changeability, realism, and relevance to the user should therefore be considered when evaluating counterfactual explanations (Guidotti, 2024).

It is also important not to confuse counterfactual explanations with causal relationships in the real world. The fact that a model produces a different output for a different input does not, by itself, prove that making the same change in real life would produce the same outcome. These explanations should therefore primarily be regarded as tools for examining model behavior.

3.5. Gradient-Based and Visual Explanation Methods

Some explanation methods used with DL models exploit the sensitivity of model outputs to changes in their inputs. Gradient-based methods examine the relationship between changes in the output and components of the input to identify the features to which the model is sensitive (Wang et al., 2024). Integrated Gradients evaluates changes in a model's output along a path between an input

and a reference input by using gradients. The method provides an axiomatic approach to addressing situations in which ordinary gradients may be insufficient (Sundararajan et al., 2017). Gradient-weighted Class Activation Mapping (Grad-CAM), in turn, aims to display image regions associated with a particular class prediction in CNNs as a heatmap. This method makes it possible to visually examine which regions of an image may contribute to a classification result (Selvaraju et al., 2017).

These visualizations can make it easier to investigate the image regions to which a model is sensitive. However, highlighting a particular region in a heatmap does not prove that the model understands that region in the same way humans do or that its decision necessarily depends on that region. Whether the explanation reflects the model's learned behavior must be tested separately (Adebayo et al., 2018). The reliability of gradient-based and visual explanation methods should therefore be assessed not merely by inspecting the visualizations they produce, but by systematically testing whether the explanations are consistent with the behavior learned by the model.

3.6. Attention Mechanisms and Relationship to Explanation

Attention mechanisms, particularly in natural language processing and transformer architectures, involve computations that indicate how a model distributes weights across different input components. These weights can be used to examine which components receive greater weight during processing. However, interpreting attention weights directly as explanations of model decisions remains controversial (Jain & Wallace, 2019; Lyu et al., 2024).

Jain and Wallace (2019) presented evidence showing that attention weights cannot always be regarded as reliable explanations for model predictions. A high attention weight assigned to a word or

text segment does not, by itself, establish its influence on the final prediction.

Attention weights should therefore be treated as indicators that can help investigate a model's internal computations. Their validity as explanations must be independently established for the particular model and task.

3.7. Rule-Based and Example-Based Explanations

Rule-based explanations aim to describe the conditions under which a decision is made in an understandable form. Example-based methods, on the other hand, help investigate model decisions by relating a prediction to similar instances. These approaches can be useful, particularly when explanations need to be understood by non-technical users (Guidotti et al., 2018; Hong & Park, 2025).

The Anchors approach developed by Ribeiro et al. (2018) focuses on identifying sets of high-precision conditions under which a particular prediction remains valid. However, a rule that achieves high precision for certain instances is not necessarily valid across a broader set of examples. Similarly, two instances that a model considers mathematically similar may not be meaningfully similar from a human perspective.

Accordingly, when using rule-based and example-based explanations, it is important to consider the conditions under which an explanation remains valid, its scope, and the suitability of the similarity measure being used.

4. Comparison of Explanation Methods

Because explanation methods address different questions, it would be inappropriate to claim that a single method is the best choice for every application. SHAP and LIME help examine feature contributions, whereas counterfactual explanations focus on the changes required to obtain alternative outcomes. Methods such as

Integrated Gradients and Grad-CAM can be used to investigate the relationship between inputs and the outputs of DL models. The main characteristics of these explanation methods are summarized comparatively in Table 1 (Ribeiro et al., 2016; Lundberg & Lee, 2017; Selvaraju et al., 2017; Sundararajan et al., 2017; Adebayo et al., 2018; Guidotti et al., 2018; Ribeiro et al., 2018; Jain & Wallace, 2019; Molnar, 2022; Guidotti, 2024).

Table 1 Comparison of the main characteristics of explanation methods

Method	Main Purpose	Advantage	Limitation
SHAP	Attribute feature contributions to predictions	Can be used for individual and aggregate analyses	Depends on assumptions and computational cost
LIME	Explain a specific prediction using a local model	Applicable to different model types	Results may vary with sampling and parameter settings
Feature importance analysis	Examine variables considered important by the model	Provides insights into overall behavior	Does not establish causal effects
Counterfactual explanations	Identify changes associated with an alternative outcome	Can provide actionable information	Suggested changes may be unrealistic or impractical
Integrated Gradients	Examine input contributions using gradients	Can be used with deep neural networks	May be sensitive to the reference input
Grad-CAM	Visualize regions associated with an image prediction	Facilitates visual inspection	Does not, by itself, establish the reason for a decision
Attention weights	Examine the distribution of attention	Provides an indication of internal computations	Not always a reliable explanation
Rule-based and example-based methods	Explain decisions through conditions or examples	Can be understandable to humans	May involve limitations in scope and similarity

In practice, method selection should begin by identifying the question that needs to be explained. Feature importance analysis may be appropriate when the objective is to investigate which variables the model generally relies on. SHAP or LIME may be considered when the rationale behind a specific prediction is of interest. Counterfactual explanations may be more suitable when the aim is to inform users about how an alternative outcome might be achieved.

It is also possible to combine multiple methods. For example, general feature importance can be examined first, followed by SHAP explanations for selected instances. Nevertheless, different methods should not be expected to produce identical results, as they rely on different assumptions and explanatory objectives. Differences between explanations can therefore serve as an indication that the model and the methods themselves warrant further investigation.

5. Application Domains and Current Challenges

5.1. Healthcare and Clinical Decision Support

AI can be used in healthcare for tasks such as medical image analysis, disease risk prediction, and clinical decision support. In these applications, it is important not only for a model to produce accurate predictions but also for users to understand how those predictions should be assessed in a clinical context. Explanation methods can help healthcare professionals examine model outputs and identify potential errors (Arrieta et al., 2020; Saw et al., 2025).

For example, the image regions to which an image classification model is sensitive can be examined using methods such as Grad-CAM. However, the fact that a visualization highlights a clinically meaningful region does not, by itself, prove that the model identified the disease for the correct reasons. The model may also rely on image characteristics that are not directly related to the disease. Explanations should therefore not replace clinical validation or independent performance evaluations (Selvaraju et al., 2017; Adebayo et al., 2018).

The intended audience is also important in this domain. The technical details required by a researcher may differ from the information a clinician needs when making a decision. Explanations intended for patients may require a different language and level of detail. The target user and decision-making context should therefore be taken into account when designing explanations.

5.2. Finance and Credit Assessment Systems

In the financial sector, ML can be used for tasks such as credit risk prediction, fraud detection, and risk management. In these applications, explanations can support the examination of features associated with a particular prediction and the investigation of potential model errors. For example, SHAP or LIME can be used to examine the contributions of specific variables to a credit application assessment (Ribeiro et al., 2016; Lundberg & Lee, 2017).

However, the importance of a variable in a prediction does not establish that a decision is fair or legally compliant. If the training data reflect historical inequalities, explanation methods may help investigate these patterns, but they do not eliminate them automatically. Fairness assessments must also consider model performance across different groups, data quality, and compliance with the rules governing the decision-making process.

Counterfactual explanations can also be used to provide information to users in financial applications. For example, they can indicate the conditions under which a model might produce a different decision. Nevertheless, the suggested changes must be realistic and feasible. A mathematical scenario that changes a model's output may not translate into a recommendation that a user can implement in real life (Wachter et al., 2018; Guidotti, 2024).

5.3. Cybersecurity and Anomaly Detection

In cybersecurity, ML is used for tasks such as network traffic classification, malware detection, and the identification of unusual behavior (Mohale & Obagbuwa, 2025). Explanation methods can help investigate why an event was classified as suspicious or which features increased its risk score (Guidotti et al., 2018; Calzarossa et al., 2025). Security analysts can use these explanations to compare model outputs with independent indicators and investigate potential false positives.

However, explainability has a particular dimension in cybersecurity: detailed information about a model's decision logic can support defensive operations, but it can also help malicious actors understand the system's limitations. It is therefore important to consider who receives an explanation and the level of detail shared. Explainability should form part of security assessment rather than replace robustness testing, error analysis, or operational monitoring.

5.4. Public Services and High-Stakes Decisions

Automated decision systems used in public services, education, employment, and social assistance can affect individuals' rights and opportunities. Understanding the reasons behind decisions, investigating errors, and auditing decision-making processes are therefore important. In high-stakes applications, implementing a system that generates post-hoc explanations may not be sufficient; the use of directly interpretable models should also be considered whenever feasible (Rudin, 2019).

The European Union's Artificial Intelligence Act establishes a regulatory framework that includes risk-based obligations and transparency-related requirements for certain AI systems. However, the regulation should not be interpreted as imposing the same unrestricted explanation requirements on every AI system. The applicable requirements depend on the system's classification, intended purpose, and relevant provisions (European Parliament & Council of the European Union, 2024).

The Artificial Intelligence Risk Management Framework published by the US National Institute of Standards and Technology (NIST) also links explainability and interpretability to the management of risks associated with trustworthy AI. This approach supports the view that explainability should not be reduced to a single technical method but should instead be incorporated into the

design, development, evaluation, and use of AI systems (NIST, 2023).

5.5. Explainability, Trust, and User Evaluation

One of the major challenges in XAI is the assumption that providing more explanations will always increase trust. In practice, however, the goal should not simply be to increase trust but to ensure that it is appropriately calibrated. Excessive trust in an unreliable model, as well as the unjustified rejection of a reliable one, can lead to undesirable outcomes (Miller, 2019; NIST, 2023).

The benefits explanations provide to users should not be measured solely by whether users like them or find them easy to understand. It is also necessary to examine whether explanations improve decision quality, help users understand model limitations, and prevent misplaced confidence. The literature on XAI evaluation emphasizes that user-centered evaluation constitutes a distinct dimension that complements technical criteria (Miller, 2019; Naveed et al., 2024).

Accordingly, two fundamental questions should be considered together when evaluating explanations: Does the explanation accurately reflect the model's behavior? Can the intended user understand and use the explanation appropriately? Answering only one of these questions is insufficient to assess the overall quality of an explanation.

5.6. Current Challenges and Research Directions

One of the major challenges in XAI is establishing context-appropriate criteria for evaluating the quality of different explanation methods. Because these methods have different objectives, comparing all of them using a single metric is difficult. For example, evaluating a feature attribution method and a method that identifies the changes required to obtain alternative outcomes using the same

criterion may overlook their fundamental differences (Ding et al., 2022; Schwalbe & Finzel, 2024).

Another challenge concerns the stability and reliability of explanations. Producing substantially different explanations for similar inputs can make it difficult for users to understand model behavior. Nevertheless, consistency alone is not sufficient; an explanation must also be sensitive to the model's actual behavior. Explanation methods should therefore be tested both for stability and for their ability to represent model behavior accurately (Adebayo et al., 2018; Carvalho et al., 2019; Calzarossa et al., 2025; Fridkin & Bendersky, 2026).

The relationship between explainability and causality also requires careful consideration. Feature contributions, gradients, and changes in model predictions can provide information about how a model behaves, but they do not independently establish real-world cause-and-effect relationships. Explaining model behavior must therefore be distinguished from identifying causal mechanisms in the real world (Wachter et al., 2018; Baron, 2023).

The growing adoption of generative AI and large language models (LLMs) has introduced new questions for explainability research. The complex internal structures of these systems can make it difficult to fully explain the processes through which a particular response is generated. Although attention weights and feature attribution methods can be useful for specific investigations, they should not be assumed to provide a complete explanation of the model's internal workings or the correctness of its responses (Jain & Wallace, 2019; Fridkin & Bendersky, 2026).

Finally, explainability should not be addressed only at the final stage of model development. Different explanatory needs may arise during data preparation, model development, testing, deployment, and post-deployment monitoring. The selection of

explanation methods should therefore be planned alongside model documentation, risk management, and user evaluation (NIST, 2023).

6. Conclusion and Future Research

XAI is an important research field that aims to make the predictions of complex ML models more understandable to humans. Feature attribution methods such as SHAP and LIME help investigate individual predictions, while feature importance analyses provide insights into overall model behavior. Counterfactual explanations identify changes that may lead to alternative outcomes, whereas gradient-based and visual methods can be used to investigate the relationship between inputs and DL model outputs.

Each of these methods has its own assumptions, conditions of use, and limitations. An explanation method should therefore not be selected solely because it is widely used or produces visually understandable outputs. The question to be explained must first be identified, the suitability of the method must be tested, and the results must be evaluated in the context in which they will be used.

An understandable explanation does not establish that a model is accurate or fair. Explanation methods can account for incorrect decisions and may, in some cases, represent model behavior only approximately. The technical reliability of explanations and the benefits they provide to users should therefore be evaluated separately.

Future research will continue to focus on evaluating explanation methods more consistently, developing forms of explanation that meet user needs, and clarifying the boundaries between causal explanations and explanations of model behavior. In addition, explainability should be considered alongside fairness, robustness, accountability, and risk management.

In conclusion, XAI provides tools that contribute to understanding black-box models, but it is not sufficient on its own to ensure trustworthy AI. An effective approach requires selecting an appropriate explanation method, testing the resulting explanations, and evaluating the findings in the relevant decision-making context. Particularly in high-stakes applications, explainability should be treated as part of a broader assessment and governance process that extends throughout the entire model lifecycle.

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018). Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems* (Vol. 31, pp. 9505–9515).

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115. <https://doi.org/10.1016/j.inffus.2019.12.012>

Baron, S. (2023). Explainable AI and causal understanding: Counterfactual approaches considered. *Minds and Machines*, 33, 347–377. <https://doi.org/10.1007/s11023-023-09637-x>

Calzarossa, M. C., Giudici, P., & Zieni, R. (2025). An assessment framework for explainable AI with applications to cybersecurity. *Artificial Intelligence Review*, 58, Article 150. <https://doi.org/10.1007/s10462-025-11141-w>

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), Article 832. <https://doi.org/10.3390/electronics8080832>

Di Marino, A., Bevilacqua, V., Ciaramella, A., De Falco, I., & Sannino, G. (2025). Ante-hoc methods for interpretable deep models: A survey. *ACM Computing Surveys*, 57(10), Article 262, 1–36. <https://doi.org/10.1145/3728637>

Ding, W., Abdel-Basset, M., Hawash, H., & Ali, A. M. (2022). Explainability of artificial intelligence methods, applications and challenges: A comprehensive survey. *Information Sciences*, 615, 238–292. <https://doi.org/10.1016/j.ins.2022.10.013>

Ebrahimi, S., Abdelhalim, E., Hassanein, K., & Head, M. (2025). Reducing the incidence of biased algorithmic decisions through feature importance transparency: An empirical study.

European Journal of Information Systems, 34(4), 636–664.
<https://doi.org/10.1080/0960085X.2024.2395531>

European Parliament & Council of the European Union. (2024). *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)*. Official Journal of the European Union.

Fayyaz, A. M., Abdulkadir, S. J., Talpur, N., Al-Selwi, S. M., Hassan, S. U., & Sumiea, E. H. (2025). Grad-CAM (Gradient-weighted Class Activation Mapping): A systematic literature review. *Computers in Biology and Medicine*, 198, Article 111200. <https://doi.org/10.1016/j.compbiomed.2025.111200>

Fridkin, S., & Bendersky, M. (2026). Interpretable machine learning: A comprehensive review of foundations, methods, and the path forward. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 16(1), Article e70075. <https://doi.org/10.1002/widm.70075>

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 1189–1232. <https://doi.org/10.1214/aos/1013203451>

Guidotti, R. (2024). Counterfactual explanations and how to find them: Literature review and benchmarking. *Data Mining and Knowledge Discovery*, 38, 2770–2824. <https://doi.org/10.1007/s10618-022-00831-6>

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5), Article 93. <https://doi.org/10.1145/3236009>

Hong, S., & Park, W. (2025). Developing user-centered system design guidelines for explainable AI: A systematic literature review. *Artificial Intelligence Review*, 58, Article 386. <https://doi.org/10.1007/s10462-025-11363-y>

Jain, S., & Wallace, B. C. (2019). Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 3543–3556). Association for Computational Linguistics. <https://doi.org/10.18653/v1/N19-1357>

Lin, T., Lau, R. Y. K., & Hu, S. (2026). A critical review of state-of-the-art explainable artificial intelligence (XAI) methods and their business applications. *Artificial Intelligence Review*, 59, Article 161. <https://doi.org/10.1007/s10462-026-11565-y>

Lipton, Z. C. (2018). The mythos of model interpretability. *ACM Queue*, 16(3), 31–57. <https://doi.org/10.1145/3236386.3241340>

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems* (Vol. 30, pp. 4765–4774).

Lyu, Q., Apidianaki, M., & Callison-Burch, C. (2024). Towards faithful model explanation in NLP: A survey. *Computational Linguistics*, 50(2), 657–723. https://doi.org/10.1162/coli_a_00511

Meng, H., Wagner, C., & Triguero, I. (2024). SEGAL time series classification—Stable explanations using a generative model and an adaptive weighting method for LIME. *Neural Networks*, 176, Article 106345. <https://doi.org/10.1016/j.neunet.2024.106345>

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267, 1–38. <https://doi.org/10.1016/j.artint.2018.07.007>

Mohale, V. Z., & Obagbuwa, I. C. (2025). A systematic review on the integration of explainable artificial intelligence in intrusion detection systems to enhancing transparency and interpretability in cybersecurity. *Frontiers in Artificial Intelligence*, 8, Article 1526221. <https://doi.org/10.3389/frai.2025.1526221>

Molnar, C. (2022). *Interpretable machine learning: A guide for making black box models explainable* (2nd ed.).

National Institute of Standards and Technology. (2023). *Artificial intelligence risk management framework (AI RMF 1.0)* (NIST AI 100-1). U.S. Department of Commerce. <https://doi.org/10.6028/NIST.AI.100-1>

Naveed, S., Stevens, G., & Robin-Kern, D. (2024). An overview of the empirical evaluation of explainable AI (XAI): A comprehensive guideline for user-centered evaluation in XAI. *Applied Sciences*, 14(23), Article 11288. <https://doi.org/10.3390/app142311288>

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why should I trust you?”: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1135–1144). Association for Computing Machinery. <https://doi.org/10.1145/2939672.2939778>

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 1527–1535. <https://doi.org/10.1609/aaai.v32i1.11491>

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1, 206–215. <https://doi.org/10.1038/s42256-019-0048-x>

Salih, A. M., Raisi-Estabragh, Z., Galazzo, I. B., Radeva, P., Petersen, S. E., Lekadir, K., & Menegaz, G. (2025). A perspective on explainable artificial intelligence methods: SHAP and LIME. *Advanced Intelligent Systems*, 7(1), Article 2400304. <https://doi.org/10.1002/aisy.202400304>

Saw, S. N., Yan, Y. Y., & Ng, K. H. (2025). Current status and future directions of explainable artificial intelligence in medical imaging. *European Journal of Radiology*, 183, Article 111884. <https://doi.org/10.1016/j.ejrad.2024.111884>

Schwalbe, G., & Finzel, B. (2024). A comprehensive taxonomy for explainable artificial intelligence: A systematic survey of surveys on methods and concepts. *Data Mining and Knowledge Discovery*, 38, 3043–3101. <https://doi.org/10.1007/s10618-022-00867-8>

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 618–626). <https://doi.org/10.1109/ICCV.2017.74>

Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 3319–3328). PMLR.

Wachter, S., Mittelstadt, B., & Russell, C. (2018). Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31(2), 841–887.

Wang, Y., Zhang, T., Guo, X., & Shen, Z. (2024). Gradient based feature attribution in explainable AI: A technical review. *arXiv*. <https://doi.org/10.48550/arXiv.2403.10415>

CHAPTER 6

IDENTIFYING SIMILARITIES BETWEEN PROPOSED AND EXISTING COURSES IN GRADUATE PROGRAMMES USING NATURAL LANGUAGE PROCESSING

EBRU YILMAZ İNCE¹

Murat İNCE²

Introduction

Universities have become heavy producers of text, not just teachers of it. Course descriptions, programme structures, syllabi, learning outcomes and a thick layer of institutional paperwork accumulate year on year. Once that material is read as data rather than stored and forgotten, the academic processes behind it can be managed more deliberately and more sustainably (Jurafsky & Martin, 2020; Romero & Ventura, 2020; Aldowah et al., 2019).

¹ Doç.Dr., Isparta University of Applied Sciences, Computer Technologies, Orcid: 0000-0001-9462-0363

² Doç.Dr., Isparta University of Applied Sciences, Computer Technologies, Orcid: 0000-0001-5566-5008

Graduate programmes sit at the sharp end of this. Their content has to track a moving research frontier, so academics put forward new courses each term and the relevant boards weigh them up. A proposal that repeats too much of an existing course threatens the diversity a programme is supposed to protect, which is why the overlap is held below a set ceiling. In the institutional context examined here that ceiling is 70%; a proposal whose content overlaps an existing course at or above this level is sent back for review rather than added as it stands.

In practice that check still falls to board members reading side by side. In a large graduate institute, with well over a hundred courses already on the books, the comparison is slow, tiring and open to the quirks of individual judgement. An automatic first pass would relieve all three pressures, and natural language processing is well suited to supply one.

Measuring how alike two texts are a staple of the field. The literature usually splits the methods in two; lexical measures, which count shared words and their forms, and semantic measures, which weigh the contextual meaning of words and so read content more deeply (Gomaa & Fahmy, 2013; Wang & Dong, 2020; Chandrasekaran & Mago, 2021). The present study works at the lexical and subword end of that spectrum, where methods are fast, transparent and easy to reproduce.

This chapter applies that toolkit to a concrete institutional question. We compared 132 analysed existing courses with 31 proposed in 2024 at a university graduate institute, running each proposal through three measures, Jaccard overlap, TF-IDF cosine similarity and a character n-gram cosine measure, and reading the results against the institutional 70% threshold. The aim is not to make the final call, which stays with curriculum committees, but to show that a modest multi-measure screen can give a fast, repeatable

first pass that flags proposals for expert review. Three questions guide the work;

RQ1. How similar are newly proposed graduate courses to existing courses under lexical and subword similarity measures?

RQ2. How consistently do Jaccard, TF-IDF cosine and character n-gram cosine identify potentially overlapping courses?

RQ3. Can a multi-measure similarity screen support the preliminary review of proposals against an institutional similarity threshold?

The remainder of the chapter reviews the relevant background, sets out the methods and dataset, presents and discusses the findings, and closes with the limitations and the implications for programme development.

Text Similarity and Its Role in Education

Lexical similarity grew out of the vector space model, where a document becomes a vector of weighted terms (Salton et al., 1975; Salton & McGill, 1983). TF-IDF supplies the weights, rewarding terms that are frequent in one document but rare across the collection (Spärck Jones, 1972; Ramos, 2003), and the cosine of two such vectors gives a tidy overlap score. The Jaccard index takes an even simpler view, dividing the words two texts share by the words they use between them (Arts et al., 2018; Tan et al., 2006). Character n-gram representations sit alongside these; by breaking text into overlapping character sequences, they pick up subword, orthographic and morphological overlap, which is helpful in a language where a single stem spawns many surface forms. Such measures are fast and transparent, and they remain a sensible first pass for near-duplicate detection (Manning et al., 2008).

Semantic measures try to see past the exact words. Distributional embeddings such as Word2Vec and GloVe place related terms close together in a dense space (Mikolov et al., 2013; Pennington et al., 2014), latent semantic analysis recovers meaning from co-occurrence structure (Landauer et al., 1998), and corpus- and knowledge-based measures combine several signals at once (Mihalcea et al., 2006). The Transformer, and BERT in particular, pushed this further by reading context from both directions (Vaswani et al., 2017; Devlin et al., 2019); sentence-level encoders then made whole-text comparison practical (Reimers & Gurevych, 2019; Cer et al., 2018), while a growing body of analysis has unpicked what these models learn (Rogers et al., 2020; Sitikhu et al., 2019). These contextual methods are powerful but heavier to deploy, and the present study deliberately stays with lighter lexical and subword measures, returning to contextual models among the directions for future work.

In education, text similarity already underpins academic plagiarism detection, where lexical signals are combined to catch reworded passages (Foltýnek et al., 2019; Alzahrani et al., 2012; Potthast et al., 2010). The same machinery extends naturally to comparing course content for curriculum management. What is far less common is a transparent, multi-measure screen tied to an institutional overlap threshold, and work grounded in Turkish graduate course content is rarer still. This chapter sets out to fill that gap.

Methods

Text Preprocessing

Every course was represented by the concatenation of its title, description and topic list, and this text passed through a conventional cleaning pipeline built with NLTK (Chai, 2023; Bird et al., 2009).

For the word-level measures, Jaccard and TF-IDF cosine, the text was tokenised, lower-cased and stripped of punctuation, Turkish function words were removed with a Turkish stop-word list, and the remaining words were lemmatised so that inflected forms collapse onto a single term. The character n-gram measure, by contrast, was computed on the normalised but non-lemmatised text, so that the subword and morphological cues it is meant to capture were preserved rather than removed. We note in the limitations that NLTK offers only limited morphological support for Turkish, and that a dedicated analyser such as Zemberek would be a natural upgrade.

Similarity Measures

Three measures ran in parallel. The Jaccard index, Eq. (1), treats each course as a set of words and divides the size of their intersection by the size of their union, giving a quick lexical overlap for sets A and B . Because it works on sets, it ignores how often a term repeats, which makes it sensitive to shared vocabulary but blind to emphasis.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

TF-IDF weighting, Eq. (2), scores a term t in document d by its local frequency $tf_{t,d}$ against its rarity across the N documents, where df_t is the document frequency. We used word unigrams and bigrams with sublinear term frequency, smoothed IDF and L2 normalisation.

$$w_{t,d} = tf_{t,d} \cdot \log\left(\frac{N}{df_t}\right) \quad (2)$$

Two courses were then compared by the cosine of their TF-IDF vectors, Eq. (3), which measures the angle θ between vectors A and B and so stays robust to differences in length.

$$\cos \theta = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \cdot \sqrt{\sum_{i=1}^n B_i^2}} \quad (3)$$

The third measure is lexical rather than semantic. Each course was turned into character n-gram TF-IDF vectors using within-word n-grams of length two to five (scikit-learn’s `char_wb` analyser) with sublinear term frequency and L2 normalisation, and two courses were compared by the cosine of these vectors. Character n-grams capture subword and orthographic overlap, which matters in an agglutinative language such as Turkish, where a shared stem surfaces in many inflected forms; they do not, however, model meaning in the way a contextual encoder does, and we are careful not to present them as a semantic measure.

Decision Rule

For each proposal we took the highest of its three similarity scores against any existing course as a single screening value and compared that value with the 0.70 threshold. A proposal reaching the threshold on any measure was flagged for expert review; the rest were marked as below the threshold. Two points follow from this design and are important to state plainly. First, the rule is deliberately conservative; because any of the three measures can raise a flag, it favours recall over precision, which suits a first-pass screen whose errors are corrected by human reviewers. Second, the three measures live on different scales and distributions, so applying one threshold across all of them is a screening heuristic rather than a validated cross-metric cut-off; the value of a normalised or weighted combination, and of an empirically validated threshold, is taken up in the limitations.

Dataset

The data come from the course catalogue of a university graduate institute. The catalogue held 136 existing courses; once

four specialisation courses were set aside, 132 entered the analysis, and these 132 were the set each proposal was compared against. Thirty-one courses proposed in 2024 were screened. Each course carried about thirteen topics, for a total of 1,716. Two of the proposals, both in robotics, returned a score of zero on every measure, which indicates that their catalogue records held no extractable content at the time of analysis; these two were set aside as a data-quality exclusion, leaving 29 proposals with usable content for interpretation. Table 1 summarises the dataset.

Table 1. Characteristics of the dataset.

Characteristic	Value
Existing courses in catalogue	136
Existing courses analysed (4 specialisation excluded)	132
Proposed courses (2024)	31
Proposed courses with usable content	29
Average topics per course	13
Total number of topics	1,716

Tools and Environment

The analysis ran on a desktop with a 12th-generation Intel Core i7-12650H CPU at 3.5 GHz and 16 GB of RAM, under Python 3.9. Preprocessing used NLTK (Bird et al., 2009), the similarity computations used scikit-learn (Pedregosa et al., 2011), and the data were handled with pandas (McKinney, 2010).

Results and Discussion

Overall Similarity Distribution

Each of the 29 proposals with usable content was compared against all 132 analysed courses under the three measures, and the maximum similarity per proposal was kept as the screening value. Figure 1 plots the twelve proposals with the highest maximum similarity, and Table 2 lists the full set. One pattern stands out at once; the character n-gram measure tends to return the largest scores,

the TF-IDF cosine sits in the middle, and Jaccard is usually the most conservative, except where two courses share a great deal of literal wording.

Figure 1. Similarity scores of the twelve proposals with the highest maximum similarity, under three measures, against the 0.70 threshold.

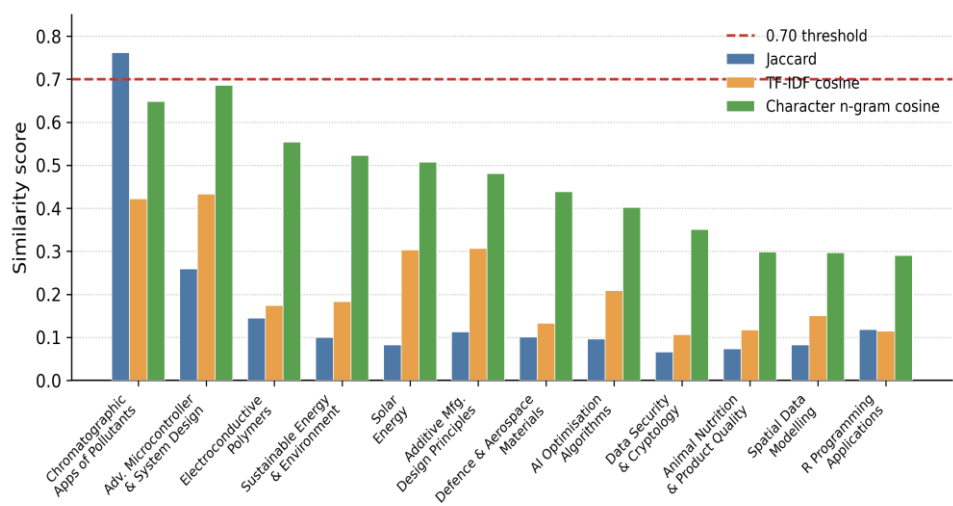


Table 2. Similarity screening of the courses proposed in 2024 (maximum similarity and screening outcome by the 0.70 threshold).

Proposed course	Jaccard	TF-IDF cos.	Char n-gram	Max.	Screening outcome
Chromatographic Applications of Pollutants	0.762	0.422	0.648	0.762	Flag for review
Advanced Microcontroller and System Design Applications	0.259	0.433	0.686	0.686	Below threshold
Electroconductive Polymers	0.145	0.174	0.554	0.554	Below threshold
Sustainable Energy and Environment	0.100	0.183	0.523	0.523	Below threshold
Solar Energy	0.083	0.303	0.507	0.507	Below threshold
Design Principles for Additive Manufacturing	0.113	0.307	0.481	0.481	Below threshold
Defence and Aerospace Materials Technologies and Applications	0.101	0.133	0.439	0.439	Below threshold

Artificial Intelligence Optimisation Algorithms	0.096	0.209	0.402	0.402	Below threshold
Data Security and Cryptology Applications	0.066	0.106	0.351	0.351	Below threshold
Animal Nutrition and Its Effects on Product Quality	0.073	0.117	0.299	0.299	Below threshold
Modelling and Mapping Techniques with Spatial Data Analysis	0.083	0.150	0.297	0.297	Below threshold
R Programming Applications	0.118	0.115	0.290	0.290	Below threshold
Artificial Intelligence Applications with R	0.083	0.114	0.287	0.287	Below threshold
Waste Reduction, Recovery and Energy from Biomass under Zero Waste	0.087	0.116	0.257	0.257	Below threshold
Biofuel Use in Sustainable Development	0.067	0.072	0.253	0.253	Below threshold
Folkloric Use of Turkey's Wild Plants	0.064	0.062	0.253	0.253	Below threshold
Sustainable Urban Area Applications and Urban Transformation	0.079	0.055	0.249	0.249	Below threshold
Spectroscopic Methods in Organic Chemistry	0.076	0.059	0.242	0.242	Below threshold
Methods for Monitoring and Researching Forest Ecosystems	0.051	0.080	0.222	0.222	Below threshold
Fundamentals of Ecological Research in Forest Ecosystems	0.061	0.050	0.198	0.198	Below threshold
Instrumental Analysis Methods	0.123	0.104	0.194	0.194	Below threshold
Advanced Deep Learning Techniques	0.052	0.071	0.192	0.192	Below threshold
Developments in Vegetable Oil Production Technology and Quality Control	0.096	0.032	0.188	0.188	Below threshold
Important Plant Areas of Turkey	0.071	0.089	0.184	0.184	Below threshold
University-Industry Cooperation and Technoparks	0.049	0.057	0.170	0.170	Below threshold
Essential Oil Technology	0.065	0.025	0.158	0.158	Below threshold
Real Estate Valuation	0.095	0.029	0.151	0.151	Below threshold
Microbiology Applications in Biogas Technologies	0.063	0.037	0.148	0.148	Below threshold
Above-ground and Below-ground Interactions	0.063	0.046	0.106	0.106	Below threshold
Robotic Motor Applications	—	—	—	—	No content (excluded)
Robot Motors and Drives Applications	—	—	—	—	No content (excluded)

Threshold Screening

Read against the 0.70 ceiling, the picture is clear. Of the 29 proposals with usable content, exactly one reached the threshold; Chromatographic Applications of Pollutants, which overlapped the existing course Analysis Methods of Pollutants in Water and Agricultural Products at a Jaccard score of 0.762. It was therefore flagged for expert review. The other 28 stayed below 0.70 on every measure. Table 3 lists the five proposals with the highest maximum similarity and the existing course each is closest to.

Table 3. Threshold screening against the 0.70 ceiling (the five proposals with the highest maximum similarity).

Proposed course	Closest existing course	Max.	Screening outcome
Chromatographic Applications of Pollutants	Analysis Methods of Pollutants in Water and Agricultural Products	0.762	Flag for review
Advanced Microcontroller and System Design Applications	Wireless System Design Applications	0.686	Below threshold
Electroconductive Polymers	Polymer Chemistry	0.554	Below threshold
Sustainable Energy and Environment	Alternative Energy Sources	0.523	Below threshold
Solar Energy	Plant Use in Solar Cell Production	0.507	Below threshold

Discussion

The single flagged course is instructive. Its overlap was caught by Jaccard, at 0.762, even though the TF-IDF cosine put it at only 0.422 and the character n-gram measure at 0.648, both below the threshold. The gap between Jaccard and TF-IDF here is itself informative; Jaccard rewards the raw set of shared words, so heavy reuse of the same specialised vocabulary pushes it up, whereas TF-IDF down-weights terms that recur across many course descriptions and so reads the pair as less alike. In practice this means the proposal leaned on much of the existing course's wording without being its statistical twin, and only the measure most sensitive to literal overlap

raised the flag. The practical lesson is that running several measures together guards against the blind spots of any one of them, which is consistent with the mixed-signal approach favoured in plagiarism research (Foltýnek et al., 2019).

More broadly, the 28 proposals that fell below the threshold are better described as below the textual-similarity ceiling than as original; the measures compare wording, not learning outcomes or pedagogical intent, so two courses can read as distinct yet still share substantial academic ground, and the reverse is also possible. Read this way, the screen says which proposals need a closer human look, not which are genuinely novel. The two robotics proposals that returned zero on every measure are a reminder of the same point from the data side; their records held no extractable content, so the zeros reflect missing input rather than true distinctiveness, and they were set aside rather than interpreted.

Limitations and Future Work

Several limitations frame these results, and each point to a clear next step. The study covers one graduate institute in a single proposal cycle, so it is best read as a practical case study rather than a general model. No independent expert ground truth was collected, so the screen is validated against the institutional threshold rather than against human judgements of overlap; a small annotation study, in which two or three academics rate the top candidate pairs and agreement is measured with Cohen's or Fleiss' kappa, would let precision, recall and F1 be reported and would test whether 0.70 is the right line to draw. The decision rule takes the maximum across three measures that sit on different scales, so a normalised or weighted combination, or a learned ensemble, would rest on firmer ground than a shared threshold. The character n-gram measure captures subword and orthographic overlap but not meaning; adding a genuine semantic measure, a Turkish or multilingual sentence

transformer such as Sentence-BERT or a BERTurk-based encoder (Reimers & Gurevych, 2019; Devlin et al., 2019; Schweter, 2020), is the most promising extension and would let lexical, subword and semantic signals be compared directly. Finally, the Turkish preprocessing relied on NLTK, whose morphological support for Turkish is limited; a dedicated analyser such as Zemberek would sharpen lemmatisation, and widening the corpus across institutions and cycles would test how far the approach travels.

Conclusion

This chapter set out to screen graduate course proposals for content overlap, using natural language processing in place of a purely manual first pass. Across 132 analysed existing courses and 31 proposed in 2024, three measures, Jaccard overlap, TF-IDF cosine similarity and a character n-gram cosine measure, were combined and read against the institutional 70% threshold. Of the proposals with usable content, only one reached the threshold, overlapping an existing course at a Jaccard score of 0.762 and so warranting expert review; the rest fell below it.

The wider value lies in the framing as much as the finding. A multi-measure similarity screen gives academic boards a fast, consistent and reproducible first pass that prioritises proposals for human review, rather than a verdict that replaces it. The right picture is human-in-the-loop: the screen flags candidates, and the committee decides. Strengthened with expert validation, a principled way of combining measures and a genuine semantic model, and extended beyond a single institution and language, the approach could become a standing aid in curriculum development, turning a slow manual chore into a routine, data-informed step.

References

- Aldowah, H., Al-Samarraie, H., & Fauzy, W. M. (2019). Educational data mining and learning analytics for 21st century higher education: A review and synthesis. *Telematics and Informatics*, 37, 13–49.
- Alzahrani, S. M., Salim, N., & Abraham, A. (2012). Understanding plagiarism linguistic patterns, textual features, and detection methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 42(2), 133–149.
- Arts, S., Cassiman, B., & Gomez, J. C. (2018). Text matching to measure patent similarity. *Strategic Management Journal*, 39(1), 62-84.
- Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media.
- Cer, D., Yang, Y., Kong, S.-Y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.-H., Strope, B., & Kurzweil, R. (2018). Universal sentence encoder. *arXiv:1803.11175*.
- Chai, C. P. (2023). Comparison of text preprocessing methods. *Natural Language Engineering*, 29(3), 509–553.
- Chandrasekaran, D., & Mago, V. (2021). Evolution of semantic similarity: A survey. *ACM Computing Surveys*, 54(2), 1–37.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019* (pp. 4171–4186). Association for Computational Linguistics.
- Foltýnek, T., Meuschke, N., & Gipp, B. (2019). Academic plagiarism detection: A systematic literature review. *ACM Computing Surveys*, 52(6), 1–42.

- Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13), 13–18.
- Jurafsky, D., & Martin, J. H. (2014). *Speech and language processing* (Vol. 3, pp. 2-5). London: Pearson.
- Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25(2–3), 259–284.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.
- McKinney, W. (2010). Data structures for statistical computing in Python. In *Proceedings of the 9th Python in Science Conference* (pp. 56–61).
- Mihalcea, R., Corley, C., & Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of AAAI 2006* (pp. 775–780). AAAI Press.
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv:1301.3781*.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proceedings of EMNLP 2014* (pp. 1532–1543). Association for Computational Linguistics.
- Potthast, M., Stein, B., Barrón-Cedeño, A., & Rosso, P. (2010). An evaluation framework for plagiarism detection. In *Proceedings*

- of COLING 2010 (pp. 997–1005). Association for Computational Linguistics.
- Ramos, J. (2003). Using tf-idf to determine word relevance in document queries. In Proceedings of the first instructional conference on machine learning (Vol. 242, No. 1, pp. 29-48).
- Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Proceedings of EMNLP-IJCNLP 2019 (pp. 3982–3992). Association for Computational Linguistics.
- Rogers, A., Kovaleva, O., & Rumshisky, A. (2020). A primer in BERTology: What we know about how BERT works. Transactions of the Association for Computational Linguistics, 8, 842–866.
- Romero, C., & Ventura, S. (2020). Educational data mining and learning analytics: An updated survey. WIREs Data Mining and Knowledge Discovery, 10(3), e1355.
- Salton, G., & McGill, M. J. (1983). Introduction to modern information retrieval. McGraw-Hill.
- Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613–620.
- Schweter, S. (2020). BERTurk: BERT models for Turkish. Zenodo. <https://doi.org/10.5281/zenodo.3770924>
- Sitikhu, P., Pahi, K., Thapa, P., & Shakya, S. (2019). A comparison of semantic similarity methods for maximum human interpretability. In 2019 Artificial Intelligence for Transforming Business and Society (AITB) (pp. 1–4). IEEE.

- Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1), 11–21.
- Tan, P.-N., Steinbach, M., & Kumar, V. (2006). *Introduction to data mining*. Pearson Education.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* 30 (pp. 5998–6008).
- Wang, J., & Dong, Y. (2020). Measurement of text similarity: A survey. *Information*, 11(9), 421.

CHAPTER 7

DIGITAL PHENOTYPING OF MIGRAINE: PERSONALIZED MONITORING, RISK MODELING, AND DECISION SUPPORT WITH ARTIFICIAL INTELLIGENCE AND ENGINEERING APPROACHES

Şule YÜCELBAŞ¹
Cüneyt YÜCELBAŞ²

INTRODUCTION

The Clinical and Societal Significance of Migraine

Migraine is a primary headache disorder whose symptoms vary considerably from one person to the next. Besides headache, an attack may bring nausea, sensitivity to light and sound, worsening with physical activity and, in some people, transient neurological aura symptoms. The International Classification of Headache Disorders defines migraine as migraine without aura, migraine with aura, and related subtypes. Assessment depends on the features of the pain, but also on attack frequency, duration, and accompanying

¹ Assoc.Prof.Dr., Tarsus University, Department of Computer Engineering, Orcid: 0000-0002-6758-8502

² Assoc.Prof.Dr., Tarsus University, Department of Electronics and Automation, Orcid: 0000-0002-4005-6557

symptoms (HCCIHs, 2018). This classification is the basic reference for defining clinical labels in AI-based analyses. Operational prediction targets, such as an attack onset derived from a diary, still have to be defined separately (HCCIHs, 2018; Ferrari et al., 2022).

The Global Burden of Disease 2023 analysis estimated that headache disorders affect billions of people worldwide, and that migraine accounts for a considerable share of the years lived with disability (GBD, 2025a). An analysis based on GBD 2021 data reported that migraine prevalence rose from roughly 733 million cases in 1990 to 1.16 billion in 2021 (Lu et al., 2025). Figures like these come from models, and “cases” should not be read as the number of people with a confirmed individual diagnosis. Even so, the findings point to a real need for monitoring methods that are accessible and sustainable.

A defining feature of migraine is that symptoms shift over days and even hours. In a clinical interview, retrospective recall can distort when an attack began, what may have triggered it, which medication was taken, and how much daily functioning was lost. A regular headache diary can help a person and their healthcare professional see patterns and treatment response more clearly (American, 2025). Digital diaries can make recording easier, but using an app is not in itself evidence of clinical benefit or diagnostic accuracy (Ferrari et al., 2022; American, 2025).

Defining the Problem From an AI and Engineering Perspective

The role of engineering and Artificial Intelligence (AI) in migraine research goes beyond training a classifier. The real problem is to bring together data from different sources, with different timestamps and degrees of reliability, under a meaningful representation; to identify clinically useful patterns; and to turn the output into a safe decision process. It therefore helps to keep the following target tasks separate:

- distinguishing symptom patterns that support a migraine diagnosis,
- predicting a person's attack probability over a defined time horizon,
- estimating attack duration or functional loss,
- tracking change after a treatment or lifestyle modification,
- producing a summary that a clinician can readily understand.

These tasks differ in their labels, time horizons, and error costs, so evaluating them against a single “success rate” would be inappropriate.

The perspective offered in this chapter is to treat migraine not as a fixed disease label but as a “digital phenotype” that changes over time within a person's context. A migraine-based digital phenotype can be thought of as a representation built by bringing together, over time, self-reported symptoms, sleep and activity patterns, environmental context, and, where appropriate, wearable measurements. It does not replace a clinical diagnosis. The aim is not to reduce a person to data but to look more carefully at temporal relationships that a clinical interview may miss. From this standpoint, the chapter examines how migraine can be monitored digitally using mobile diaries, wearables, and environmental data. It discusses what time-series analysis, machine learning, and explainable AI might contribute, along with data processing and evaluation procedures. Three cautions stay in view throughout: missing or biased data can produce misleading predictions, the relationships identified cannot be read as direct cause and effect, and digital systems cannot stand in for clinical diagnosis. No experiment with new patient data and no clinical performance result is presented

here. Instead, drawing on the literature, the chapter proposes a conceptual system architecture that accounts for person-specific change and prediction uncertainty, respects privacy, and supports clinician judgment, together with an evaluation framework for future research.

MATERIALS AND METHODS

Scope and Methodological Approach

This chapter should not be taken for a systematic review or an experimental study run on new patient data. It is a narrative work that proposes a research design, drawing on clinical classification, global burden studies (GBD, 2025a; Lu et al., 2025; GBD, 2018; GBD, 2025), and sources on digital headache diaries. The architecture, tables, and evaluation plan below are a proposed methodological framework, not experimental performance results. A systematic assessment of the evidence would call for a separate systematic review, with a predefined search strategy, inclusion and exclusion criteria, independent assessment, and proper PRISMA reporting (Page et al, 2021). The proposed reporting and evaluation approach could later be elaborated in line with the TRIPOD+AI and PROBAST+AI principles (Collins et al., 2024; Moons et al., 2025).

Proposed Research Questions

A study proposed on this topic could reasonably be built around questions such as the following:

- Does a person's own diary data calibrate individual attack risk better than a model trained only at the population level?
- Does adding context on sleep, stress, the menstrual cycle, weather, and medication use improve prediction meaningfully compared with pain records alone?

- When missing data and logging behavior are added to the model, are predictions more reliable, or does logging frequency simply become a proxy for disease severity?
- Does an interface that is explainable and reports uncertainty make it easier for clinicians to assess the output?

These questions can be expanded or revised according to the aim, scope, and intended audience of the research. It would be best to settle that together with experts in the field.

Data Sources and the Data Dictionary

The minimum dataset is a headache diary containing date and time, headache onset and end, pain intensity, accompanying symptoms, daily functional loss, and the acute medication used. Optional context fields may cover sleep duration and quality, self-reported stress level, skipped meals, fluid intake, physical activity, menstrual information, and environmental conditions. Sleep or activity data from smartwatches should be used only when the device's measurement limits are clearly documented. Physiological signals such as heart rate are not specific to migraine, and they should not be presented as a diagnostic biomarker on their own. Table 1 summarizes the main data domains that can be used in digital phenotyping of migraine, example variables, and what to watch for when interpreting them. This matters most when self-report and sensor measurements are considered together: neither should be read as a migraine-specific biomarker.

Table 1 Data sources and variables that can be used in digital phenotyping of migraine

Data domain	Example variables	Measurement format	Main risk / point of caution
Symptom diary	Onset and end, intensity, nausea, light sensitivity	Timestamped self-report	Recall error; missing entries
Functional loss	Attendance at work or school, reduced daily activity	Likert scale or standardized instrument	Comparability across individuals
Medication use	Drug class, time, dose (where appropriate)	Event log	Clinical supervision and protection of sensitive data
Sleep/activity	Sleep duration, awakenings, step and activity summary	Wearable device or self-report	Differences between devices; measurement error
Context	Stress, meals, menstruation, weather	Self-report / external data	Confounding variables; causal fallacy
Logging behavior	Skipped days, delayed entries, app usage	System metadata	Digital access and participation bias

The data dictionary should define each variable’s unit, permitted values, source, temporal resolution, missingness code, and clinical meaning. For “attack onset,” for example, it must be decided in advance whether the reference point is the moment the person first notices a symptom or the moment the headache begins. If information gathered after the time of prediction is used as model input, data leakage results and the model’s performance is overestimated. Model inputs should therefore be limited to what is actually available when the prediction is made. Only then can performance under real-world conditions be assessed reliably and without bias.

Data Processing Pipeline

The proposed pipeline has six main stages: data collection and informed consent, quality control and time alignment, feature extraction, training/validation/test splitting, modeling and calibration, and clinical evaluation and reporting. If records from the same person are scattered at random across training and test sets, the model can pick up person-specific traits and its apparent accuracy will be inflated. To evaluate how well a model performs on other

people, each participant's data should therefore sit in only one set. When the goal is instead to predict the same person's future attacks, earlier records should be used for training and later ones for testing. The two approaches answer different questions, so their results should be evaluated separately. Where possible, validation on independent data is also advisable, to see whether the model holds up across different patient groups and devices. Splits that prevent leakage, along with transparent reporting, matter for judging how a clinical prediction model will perform in practice (Collins et al., 2024; Wolff et al., 2019; Collins et al., 2015; Stubberud et al., 2023). One small prospective study used mobile diary entries and physiological measurements to predict next-day headache. With data from 18 participants, its best model reached a test ROC-AUC of 0.62. That modest performance shows the approach cannot yet be considered a dependable clinical forecasting system, and that it needs independent validation (Stubberud et al., 2023).

Missing or faulty columns in a person's data rows should not be filled automatically with zeros. In this setting, a person reporting no pain and a person making no entry for that day are different situations. Treating days without records as pain-free puts false information into the dataset and can hurt predictive performance. Missingness indicators should be kept separate. Where appropriate, multiple imputation or imputation methods learned on the training set alone should be used, and sensitivity analyses should be run under different missingness scenarios. Scaling, feature selection, balancing, and hyperparameter tuning must likewise be done within the training stage only. Otherwise, information from the test data leaks into the model and produces misleading output.

Feature Extraction and Model Families

For tabular data, logistic regression and regularized linear models are strong, interpretable starting points. Tree-based methods

such as random forests and gradient boosting can capture nonlinear relationships. Recurrent neural networks or Transformer-based time-series models may be worth exploring when there is enough good-quality timestamped data, but complex models should not be assumed to do better on small datasets. When there is little data per person, hierarchical models or adaptation to a personal baseline deserve consideration in place of added complexity.

Features can include not only raw values but also change relative to the person. Examples are the change in sleep over the previous 24–72 hours, deviation from the person’s own average, attack frequency in the past week, regularity of logging over time, and lagged values of environmental variables. This approach accepts that “little sleep” in the general population and sleeping less than usual for one particular person may not mean the same thing. Even so, such features should be reported with a clear line between those justified clinically in advance and those that came out of exploratory analysis. Table 2 compares, for different prediction tasks, example targets, the model families that can be tried first, and suitable performance measures. As the type of prediction task changes, so should the evaluation approach.

Table 2 Possible prediction tasks, model families, and evaluation measures

Task	Example target	Suitable starting models	Recommended evaluation
Attack probability	Attack onset in the next 24 hours	Logistic regression, gradient boosting	PR-AUC, Brier score, calibration
Attack duration	Duration in hours	Regression, survival analysis	MAE; survival metrics if data are censored
Functional loss	Functional impact the next day	Ordinal logistic model, tree-based model	Class-wise sensitivity, macro-F1
Personal change	Trend before and after treatment	Interrupted time series, mixed-effects model	Effect size and confidence interval
Clinical summary	Weekly symptom and medication pattern	Rule-based summary and visualization	Clinician usability and error analysis

Evaluation and Validation

When classes are imbalanced, accuracy alone misleads. In a dataset where attacks are rare, a model that always says “no attack” can reach high accuracy. Sensitivity, specificity, precision, F1, and ROC-AUC should therefore be examined, and, especially when the positive class is rare, PR-AUC and calibration too. For clinical use, what matters is whether a predicted 30% risk actually materializes about 30% of the time in similar cases. The Brier score and calibration plots help here. Decision curve analysis can ask whether a model offers net benefit at various thresholds, compared with the options of “doing nothing” or “intervening for everyone.” Calibration captures a dimension distinct from discrimination, and it needs separate examination if predicted probabilities are to be interpretable (Van Calster et al., 2019).

Confidence intervals should be computed with suitable methods such as person-level resampling; the many days belonging to one person must not be treated as independent samples. Subgroup analyses may cover age, sex, device type, aura status, and digital literacy, but unstable results in small subgroups should not harden into firm conclusions. A model failing in one subgroup is not merely a statistical problem; it is a problem of access and equity. Systematic review of model development and evaluation for bias and applicability is also recommended (Moons et al., 2025; Wolff et al., 2019).

PROPOSED SYSTEM ARCHITECTURE AND EVALUATION FRAMEWORK

End-to-End Architecture

In the proposed architecture, a mobile app or web interface forms the data-entry layer; validation and timestamp services form the data-quality layer; encrypted storage and access control form the security layer; feature generation and model serving form the

analytics layer; and visualization and clinician approval form the decision-support layer. Data processing can be done on the device as far as possible. If transfer to a central server is needed, only the fields required for the purpose should be sent, identifiers should be separated from analytic records, and a retention period should be set. Figure 1 shows the proposed system architecture for digital phenotyping of migraine. The architecture offers an integrated workflow that starts with data collection and preprocessing and extends to personalized prediction, model validation, and clinical decision support. Data security and ethical governance are handled across the system’s whole life cycle, not only at the final stage.

Figure 1 Proposed conceptual system architecture for digital phenotyping of migraine and personalized clinical decision support



Table 3 presents the main architectural layers of the proposed system, each layer’s responsibilities, and the engineering controls that can be applied if something fails. Security and privacy measures should not be confined to a single layer; they need to run through the whole system.

Table 3 Architectural layers and controls of the proposed digital phenotyping system

Layer	Responsibility	Engineering control	In case of failure
User interface	Recording symptoms and context	Accessible design, low logging burden	Short form / offline entry
Data validation	Unit, time, and range checks	Schema validation, consistency rules	Prompt the user to correct
Storage	Protecting records	Encryption, role-based access, audit trail	Restrict access, log the incident
Analytics	Feature generation and risk prediction	Versioning, leakage tests	Withhold the prediction and give the reason
Decision support	Summary and uncertainty display	Calibration, explanation, clinician check	Halt the automated decision
Monitoring	Model and data drift	Performance and subgroup tracking	Revalidate or roll back

Predictions the system produces should not be treated as a definitive diagnosis or as showing what caused a migraine attack. Instead of saying “lack of sleep caused the migraine attack,” for example, the system can note that attacks were more frequent in periods when the person’s records showed shorter sleep. That does not prove that lack of sleep directly causes attacks. Too many alerts can also cause needless anxiety, so their frequency and content need careful tuning. Every prediction should state which time window was assessed, how reliable the result is, and which decisions must not be made on the basis of the prediction. In short, the system’s purpose should not be to make a definitive diagnosis or steer treatment decisions. It should support clinical decision-making by giving users clear and dependable information for tracking attacks and assessing possible risks.

Reporting and Interpretation Limits for Future Research

No patient data were collected and no machine learning algorithm was trained for this chapter. Accordingly, no numerical results on accuracy, AUC, or clinical impact are given. What the chapter covers instead are the key considerations, grounded in the

existing literature, for developing a system that predicts migraine attacks: defining attacks accurately from a clinical standpoint, evaluating data with person and time characteristics in mind, handling missing records properly, examining how reliable the predictions are, and protecting personal data. Whether the proposed approach is useful in real-world conditions can be established only by testing it on independent data under a protocol prepared in advance.

In the results section of a study, the number of participants, the days over which records were collected, and the data collection process should be shown in a flow diagram. Missing-data rates and the distribution of migraine attacks should also be described. The performance of the models developed should be compared, and results should be assessed across both different participants and different time periods. It is also necessary to examine how closely the model's predicted risk levels match actual outcomes, in what situations wrong predictions occur, and how the model performs in different participant groups. Clinicians' and users' assessments of the system should be included as well. A model's success should be judged not only by overall accuracy but by its errors: attacks it fails to predict, and warnings it gives when no attack will occur. That gives a fuller picture of the system's strengths and limits. Current reporting guidelines for clinical prediction models should be followed when reporting results (Collins et al., 2024; Liu et al., 2020; Cruz Rivera et al., 2020).

DISCUSSION

The Contribution of the Digital Phenotype Approach

Digital monitoring is a valuable tool that helps clinicians assess patients' symptoms and changes in health status more fully during visits. When attack onset, accompanying symptoms, medication timing, and functional loss are recorded regularly, the

clinician is likely to see a more concrete pattern. The purpose of a headache diary is not to generate as much data as possible but to collect, in a sustainable way, reliable information that can influence treatment decisions (American, 2025). The best application may therefore be not the one that uses the most sensors but the one that gets meaningful data with the least burden on the person.

Another point in digital migraine tracking is that missing records should not be seen only as lost data; their possible causes matter too. Someone may be unable to log on days when the attack is severe. Another person may forget to open the app on symptom-free days. Missingness, in other words, may not be random at all but tied to disease severity. Filling such gaps with zeros, or evaluating only the days with complete records, can create systematic bias. Data collection should accordingly be designed with short entries that reduce user burden and an interface that does not penalize skipped entries. Users should also be told, briefly and plainly, about situations like these.

Causality, Triggers, and Personalization

Variables linked to migraine, such as sleep, stress, meals, hormones, or weather, can change together in observational records. Observing a variable before an attack does not by itself show that it caused the attack. Early prodromal symptoms, for instance, may have changed the person's sleep, appetite, or activity. In that case the relationship we see may reflect an early sign of an approaching attack more than a trigger. This distinction is critical if a prediction model is not to be read as a causal explanation.

Personalization raises some points that need care. A model built for one person can predict well from that individual's past data. Over time, though, the person's living conditions, medications, data-collection devices, and logging habits can change, and these changes can affect the model's performance. Performance should therefore

be monitored regularly, and retraining should be considered once enough reliable new data has accumulated. A retrained model should be tested on independent data before it goes into use. The cases where the model fails to predict migraine attacks, or warns although no attack will occur, should also be examined. Such errors may cause needless anxiety, create false reassurance, or affect decisions about seeking healthcare, and the likelihood of each should be assessed. Once the necessary checks are complete, changes to the model should be documented and the updated version released. Newly collected data should likewise not be assumed to be always correct; its accuracy and reliability should be checked before it is used in retraining.

Ethics, Privacy, and Security

Migraine diaries can contain personal and sensitive health information, such as patients' symptoms, medications, and reproductive health. Data privacy and security should therefore be considered from the design stage of digital monitoring systems onward. To begin with, only the data needed for the purpose of the research should be collected, participants should be informed clearly and understandably about how their data will be used, and the necessary consent should be obtained. How long data will be kept should be determined, security measures such as encryption and access control applied, and deletion made possible when needed. Anonymization may also fall short of removing the risk of re-identification entirely. The timing of records and rare data patterns in particular can lead to indirect identification. To reduce these risks, approaches such as secure analysis environments, aggregated results, and, where appropriate, federated learning can be considered when sharing data. Federated learning, however, does not guarantee data privacy on its own. Data security in digital migraine monitoring systems should therefore not be limited to technical measures; it should be addressed together with ethical principles such as

individual privacy, autonomy in decision-making, transparency, and accountability. International ethics guidance on AI in health also stresses the importance of these principles (WHO, 2021).

Developers of digital migraine monitoring systems also have to consider users' differing access to technology. It matters that people who do not use smartwatches, lack regular internet access, or find digital record-keeping difficult can still benefit from the system. Apps should therefore be designed, as far as possible, to work without extra devices and to be easy to use in their basic features. Nor should the use of AI-supported prediction systems become a precondition for receiving healthcare. These systems cannot replace clinical assessment, and that should not be forgotten. Headaches that begin suddenly, are very severe, or differ from a person's usual migraine symptoms can signal other health problems. Because digital systems cannot reliably identify the cause in such cases, users should be told clearly to seek medical assessment and emergency care when needed.

Limitations

The chapter's main limitation is that it rests neither on a systematic literature search nor on an experimental application supported by clinical data. The sources used establish the societal impact and clinical features of migraine, but they do not show how useful the proposed system would be under real-world conditions. The data that digital monitoring systems can draw on have their own limits as well. Self-reported records, for example, can be affected by misremembered past symptoms or by personal expectations. Measurements from smartwatches and similar wearables vary with the device's characteristics and the data-processing methods. Models built on data from a single healthcare institution or from volunteer participants alone cannot be expected to perform equally well across age groups, populations, and health systems. The reliability of the

proposed approach should therefore be evaluated in different participant groups and on independent data. Finally, a model reaching high predictive performance does not mean that patients' health or quality of life has improved. Showing such benefits in a meaningful way requires separate clinical evaluation.

A Roadmap for Future Research

In the first stage, a diary containing a small number of clinically meaningful variables should be developed through co-design with patients and clinicians. In the second, a prospective pilot study should examine usability, logging adherence, and data quality. In the third, simple baseline models and more complex ones should be compared under a preregistered analysis plan, with person-wise and time-wise testing. The next stage should carry out external validation and decision curve analysis at different centers. The final stage should be an impact study asking whether the system improves clinical decisions, quality of life, or the care process. Until these stages are complete, the application should not be described as one that “diagnoses migraine” or “prevents attacks.” If a clinical AI intervention is later to be evaluated prospectively, the protocol could be prepared following the SPIRIT-AI recommendations and the results report following CONSORT-AI (Liu et al., 2020; Cruz Rivera et al., 2020).

CONCLUSION

AI and engineering approaches can contribute to migraine management through data integration, time-series analysis, personalized risk prediction, and clinical decision support. But a dependable solution is more than a high-performing algorithm. Clinically meaningful labels, correct handling of missing records, person-wise and time-wise independent validation, calibration, explainability, privacy, and equitable access have to be designed together. The central proposal of this chapter is to build a digital

phenotyping system that states its uncertainty openly and supports the conversation between clinician and patient, rather than an AI that collapses migraine into a single class. Such an approach uses sensor data not as an end in itself but to understand the individual’s experience better and to make care decisions better informed. The clinical value of the proposed framework can be shown only through transparent, pre-planned, and independently validated research.

Table 4 brings together the key elements that a future study should report, from cohort and label definitions to validation, performance, fairness, usability, and safety, as a checklist. The list can be used alongside reporting and bias-assessment guidelines for clinical prediction models (Collins et al., 2024; Moons et al., 2025; Wolff et al., 2019; Collins et al., 2015; WHO, 2021).

Table 4 Minimum reporting checklist for the proposed research

Domain	Information to be reported
Cohort	Inclusion criteria, centers, demographic distribution, participants lost
Label	Attack definition, prediction horizon, clinical assessment method
Data quality	Missingness, timestamp consistency, sensor type and version
Modeling	Baseline model, feature selection, hyperparameter search, leakage safeguards
Validation	Person-wise, time-wise, and, where possible, external-center testing
Performance	Sensitivity, specificity, PR-AUC, calibration, confidence intervals
Fairness	Subgroup results and possible differences in data access
Usability	Logging burden, user dropout rate, clinician feedback
Safety	Consent, access control, retention period, incident response
Interpretation	Error analysis, uncertainty, clinical limits and intended use

Acknowledgements

The authors used ChatGPT (OpenAI) to assist with translation, language editing, improving the clarity and readability of the manuscript, and the generation and editing of Figure 1. All scientific content was reviewed, edited, and approved by the authors.

REFERENCES

- American Migraine Foundation. (2025, April 8). Headache journals: Tracking your migraine. <https://americanmigrainefoundation.org/resource-library/headache-journals/>
- Collins, G. S., Reitsma, J. B., Altman, D. G., & Moons, K. G. M. (2015). Transparent reporting of a multivariable prediction model for individual prognosis or diagnosis (TRIPOD): The TRIPOD statement. *British Journal of Surgery*, 102(3), 148–158. <https://doi.org/10.1002/bjs.9736>
- Collins, G. S., Moons, K. G. M., Dhiman, P., Riley, R. D., Beam, A. L., Van Calster, B., et al. (2024). TRIPOD+AI statement: Updated guidance for reporting clinical prediction models that use regression or machine learning methods. *BMJ*, 385, Article e078378. <https://doi.org/10.1136/bmj-2023-078378>
- Cruz Rivera, S., Liu, X., Chan, A.-W., Denniston, A. K., & Calvert, M. J. (2020). Guidelines for clinical trial protocols for interventions involving artificial intelligence: The SPIRIT-AI extension. *The Lancet Digital Health*, 2(10), e549–e560. [https://doi.org/10.1016/S2589-7500\(20\)30219-3](https://doi.org/10.1016/S2589-7500(20)30219-3)
- Ferrari, M. D., Goadsby, P. J., Burstein, R., Kurth, T., Ayata, C., Charles, A., Ashina, M., van den Maagdenberg, A. M. J. M., & Dodick, D. W. (2022). Migraine. *Nature Reviews Disease Primers*, 8, Article 2. <https://doi.org/10.1038/s41572-021-00328-4>
- GBD 2016 Headache Collaborators. (2018). Global, regional, and national burden of migraine and tension-type headache, 1990–2016: A systematic analysis for the Global Burden of Disease Study 2016. *The Lancet Neurology*, 17(11), 954–976. [https://doi.org/10.1016/S1474-4422\(18\)30322-3](https://doi.org/10.1016/S1474-4422(18)30322-3)

- GBD 2021 Headache Collaborators. (2025). Global, regional, and national burden of headache disorders, 1990–2021, with forecasts to 2050: A Global Burden of Disease study 2021. *Cell Reports Medicine*, 6(10), Article 102348. <https://doi.org/10.1016/j.xcrm.2025.102348>
- GBD 2023 Headache Collaborators. (2025a). Global, regional, and national burden of headache disorders, 1990–2023: A systematic analysis for the Global Burden of Disease Study 2023. *The Lancet Neurology*, 24(12), 1005–1015. [https://doi.org/10.1016/S1474-4422\(25\)00402-8](https://doi.org/10.1016/S1474-4422(25)00402-8)
- Headache Classification Committee of the International Headache Society (HCCIHs). (2018). The International Classification of Headache Disorders, 3rd edition. *Cephalalgia*, 38(1), 1–211. <https://doi.org/10.1177/0333102417738202>
- Liu, X., Cruz Rivera, S., Moher, D., Calvert, M. J., & Denniston, A. K. (2020). Reporting guidelines for clinical trial reports for interventions involving artificial intelligence: The CONSORT-AI extension. *The Lancet Digital Health*, 2(10), e537–e548. [https://doi.org/10.1016/S2589-7500\(20\)30218-1](https://doi.org/10.1016/S2589-7500(20)30218-1)
- Lu, Y., Li, Q.-Y., Gan, L., You, Y., Wang, C.-D., Guo, Z.-W., Shi, J., & Liu, X.-Y. (2025). The global and regional burden and trends of migraine from 1990 to 2021: Global Burden of Disease Study 2021. *Frontiers in Neurology*, 16, Article 1686288. <https://doi.org/10.3389/fneur.2025.1686288>
- Moons, K. G. M., Damen, J. A. A., Kaul, T., Hooft, L., Andaur Navarro, C., Dhiman, P., et al. (2025). PROBAST+AI: An updated quality, risk of bias, and applicability assessment tool for prediction models using regression or artificial intelligence methods. *BMJ*, 388, Article e082505. <https://doi.org/10.1136/bmj-2024-082505>

- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., et al. (2021). The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, 372(71). <https://doi.org/10.1136/bmj.n71>
- Stubberud, A., Ingvaldsen, S. H., Brenner, E., Winnberg, I., Olsen, A., Gravdahl, G. B., Matharu, M. S., Nachev, P., & Tronvik, E. (2023). Forecasting migraine with machine learning based on mobile phone diary and wearable data. *Cephalalgia*, 43(5). <https://doi.org/10.1177/03331024231169244>
- Wolff, R. F., Moons, K. G. M., Riley, R. D., Whiting, P. F., Westwood, M., Collins, G. S., et al. (2019). PROBAST: A tool to assess the risk of bias and applicability of prediction model studies. *Annals of Internal Medicine*, 170(1), 51–58. <https://doi.org/10.7326/M18-1376>
- World Health Organization. (2021). Ethics and governance of artificial intelligence for health: WHO guidance. <https://www.who.int/publications/i/item/9789240029200>
- Van Calster, B., McLernon, D. J., van Smeden, M., Wynants, L., & Steyerberg, E. W. (2019). Calibration: The Achilles heel of predictive analytics. *BMC Medicine*, 17, Article 230. <https://doi.org/10.1186/s12916-019-1466-7>

CHAPTER 8

CONTINUAL LEARNING FOR GEARBOX FAULT DIAGNOSIS UNDER VARYING LOAD CONDITIONS: A COMPARATIVE STUDY OF FINE-TUNING, EXPERIENCE REPLAY, AND KNOWLEDGE DISTILLATION

Canan TASTIMUR¹

Introduction

Gearbox failure diagnosis in industrial production systems arises due to variable operating conditions, high loads, and prolonged use. Early gearbox failure diagnosis offers several benefits. Chief among these is preventing sudden production system shutdowns, thus avoiding disruptions to many activities and increased costs. Since production continuity is ensured, maintenance costs are also reduced. In real industrial production lines, gearbox failure situations occur under variable operating conditions, and due to different load, speed, and time characteristics, the addition of new failure data types to the dataset can be observed. Recently, deep learning approaches have been developed as solutions to various

¹ Asst. Prof. Dr., Erzincan Binali Yildirim University, Faculty of Engineering Architecture, Department of Computer Engineering, Erzincan/Türkiye, Orcid: 0000-0002-3714-6826

failure diagnosis problems. However, the vast majority of these approaches assume that the existing dataset is static and that all training data is accessible simultaneously. However, in real-time production lines, due to variable conditions, it is likely that new problematic failure examples will be added to the dataset. In this case, the catastrophic forgetting problem occurs, where the model loses previously learned information while learning new tasks to address these newly added failures. In light of all these circumstances, the main motivation of this study is to design an architecture for gearbox fault diagnosis that goes beyond simply achieving a high instantaneous accuracy rate, ensuring the preservation of previously learned information by the model and enabling it to adapt to new tasks it will learn later.

The traditional fine-tuning approach, which adapts quickly to new operating conditions, leads to the corruption of information learned from previous tasks. In the Experience Replay method, even with a limited number of past examples, it has not been possible to completely preserve the high-level representations of the model to solve this problem. Knowledge Distillation (KD), on the other hand, preserves the learned behavior of the previous model, transferring this information to the new model and ensuring the preservation of past knowledge. In this section, three different learning strategies—Fine-Tuning, Experience Replay, and Experience Replay-based Knowledge Distillation (Replay+KD)—are compared to ensure that the previously learned information is not forgotten by adding new operating conditions and tasks sequentially to the gearbox system. Under the same experimental conditions, Fine-Tuning, Replay, and Replay+KD strategies were implemented in a total of 15 paired experiments using five folds and three different seeds. Thus, the analysis of information conservation between tasks and the stability under different experimental designs were examined, rather than just the average classification success of the models. Continual learning

performance was evaluated using traditional classification metrics such as Accuracy, Macro-F1, Precision, and Recall, as well as forgetting and backward transfer (BWT). Furthermore, the results were evaluated using the Wilcoxon signed-rank test and rank-biserial effect size analysis tests to examine the statistical reliability of the proposed models.

Methodology

To perform automatic diagnosis from different signal representations under different operating conditions in gearbox fault diagnosis, an architecture has been designed that allows simultaneous access to the training dataset and continues learning under variable conditions, thus being used in real industrial applications. The representations of healthy and faulty signals in the gearbox dataset used are shown in Figure 1. The general structure of the gearbox testing setup and the systematic approach to the fault diagnosis process are presented in Figure 2. In this section, in a continual learning (CL) problem consisting of sequential tasks, each task is defined as $\mathcal{T} = \{T_1, T_2, \dots, T_K\}$, and the task sequence in (1) is reached.

$$T_K = (D_K, \mathcal{Y}_K) \quad (1)$$

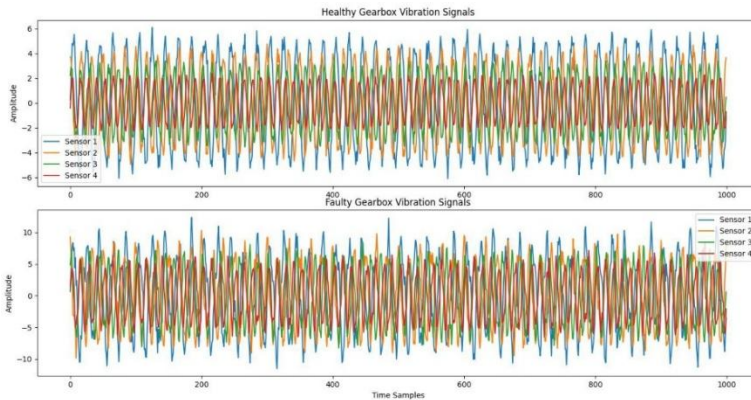


Figure 1 Healthy and faulty gearbox vibration signals

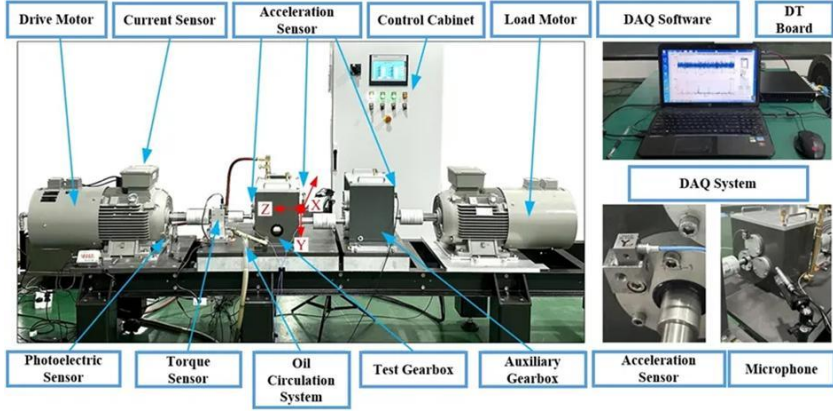


Figure 2 General structure of the gearbox testing setup (Liu et al. 2025).

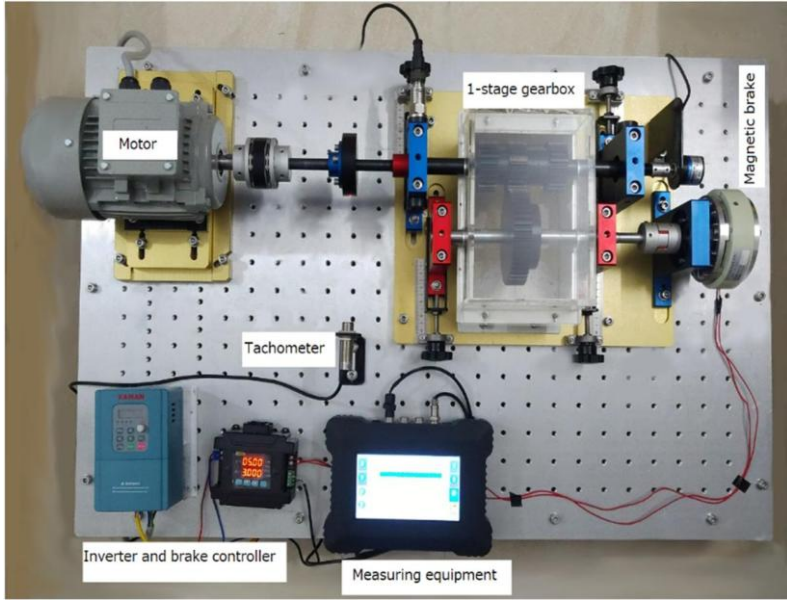


Figure 3 Gearbox setup hardware (Du et al. 2022).

The dataset obtained from the relevant working conditions is D_K , while the label for the data used in each task is $\mathcal{Y}_K$. The model completes its training on the first task D_1 and continues with the tasks sequentially as $D_1 \rightarrow D_2 \rightarrow \dots \rightarrow D_K$. During CL training, the goal is not

only to improve the performance of the new task but also to preserve the knowledge gained from previously learned tasks as much as possible (Zhang et al. 2025; Risca et al. 2025; Lin and Zhao 2026; Lin et al. 2025). In the literature, the dilemma encountered in the use of CL is the stability–plasticity expression, where plasticity refers to learning new tasks and stability refers to preserving previously learned information (Lalit Vidyasagar et al. 2026; Chen et al. 2022; Ding et al. 2024; Wei et al. 2025; Chen et al. 2023). In this section's CL study, three different learning Fine-Tuning (FT), Experience Replay (Replay), and Experience Replay with Knowledge Distillation (Replay+KD) strategies were adopted, and the results from the datasets obtained from each were expressed comparatively.

If the parameter of the model used for each task T_K is θ_K , then the first task model is as in (2). Then the model can be updated using the information it learned before. The main objective carried out in the CL process is in (3). Here, $\mathcal{P}_{new}$ represents the performance in the new task, $\mathcal{P}_{old}$ represents the level of preservation of previous tasks, and λ represents the balance between old and new information.

$$\theta_1 = \arg \min_{\theta} \mathcal{L}(D_1; \theta) \quad (2)$$

$$\max(\mathcal{P}_{new} + \lambda \mathcal{P}_{old}) \quad (3)$$

The first method, Fine-tuning CL, is considered a reference method in comparison studies. In this approach, after the model is trained on the first task, the model parameters obtained are used as the starting point for the next task. Experience Replay is based on the principle of reusing selected examples from previously learned tasks along with the new task during continual learning. This mechanism aims to represent the distribution of past tasks in the optimization process, preventing the model from over-adapting to the new task. Knowledge Distillation (KD) is a teacher-student

approach that allows the transfer of knowledge learned by the previous model to the current model. The basic continual learning approach of the study is the Replay+KD method. In this method, real past examples provided by experience replay and previous model knowledge provided by knowledge distillation are used together. Figure 4 shows the general flow of the proposed architecture.

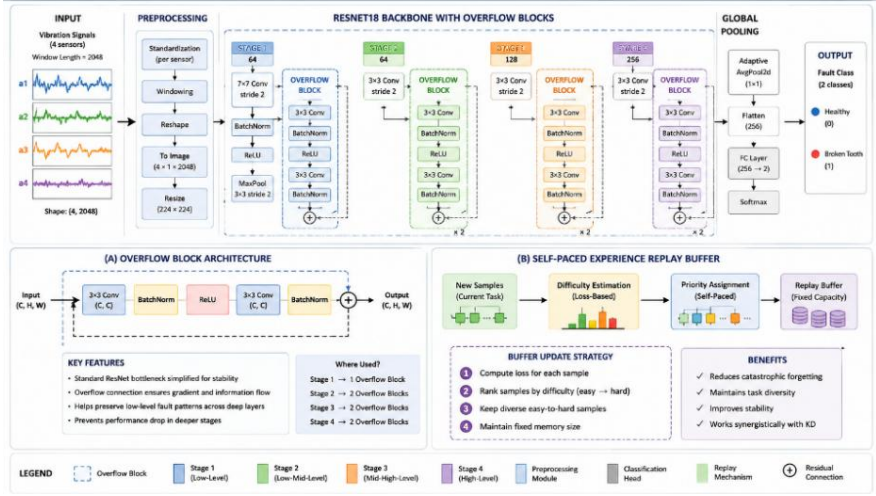


Figure 4 Overflow of the proposed architecture.

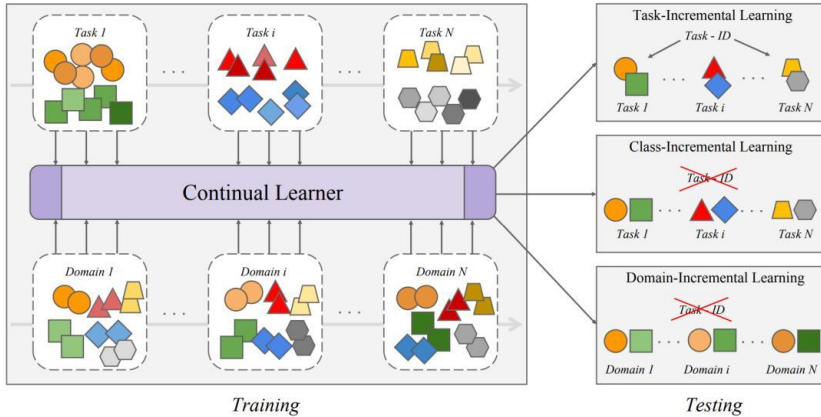


Figure 5 Setting of different offline continual learning tasks (Yang et al. 2025)

Experimental Results

In this study, three different CL approaches were tested on the same dataset to classify two different faulty and healthy states of gearboxes under working and variable operating conditions. The public dataset used (Pandya and Parey, 2018) included 10 healthy samples and 10 broken tooth records, and these records contained load levels between 0%, 10%, 20%, ..., 90%. Furthermore, each record received data from four different vibration channels. All gearbox vibration signals were passed through several preprocessing steps. The training process for all three different CL methods was conducted using the ResNet18 backbone. Additionally, to increase experimental reliability, five load level groups and three different random seedings were used, resulting in a total of 15 paired experiments for each method. Table 1 presents the general classification performance results for the three different methods.

Table 1 Main results of three different CL methods

Metric	Fine-Tuning	Replay	Replay + KD
Accuracy (%)	98.23 ± 1.32	97.52 ± 2.09	98.40 ± 1.79
Macro-F1	0.9823 ± 0.0132	0.9752 ± 0.0210	0.9839 ± 0.0179
Precision	0.9829 ± 0.0127	0.9767 ± 0.0187	0.9848 ± 0.0161
Recall	0.9823 ± 0.0132	0.9752 ± 0.0209	0.9840 ± 0.0179

When the results in Table 1 are examined, significant differences are found between the methods. While the Fine-Tuning approach forms a strong basis among the methods, the fact that the standard deviation of the Replay method is higher compared to other methods indicates increased performance variability between experiments. The highest performance was obtained with the Replay+KD approach. This shows that Replay+KD can adapt to new working conditions while preserving past knowledge in a continual learning scenario. In contrast, the fact that the simple Replay method achieved approximately 0.71 percentage points lower accuracy compared to Fine-Tuning suggests that directly reusing past examples alone may not be sufficient.

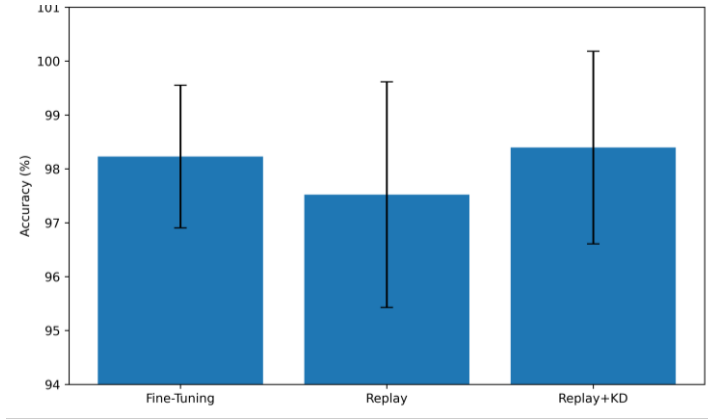


Figure 6 Overall accuracy comparison across the 15 paired runs.

The accuracy values of three different CL models under different load conditions and on five load groups are presented in Figure 7. Consecutive groups in the CL experiments can be considered as different load groups. The behavior of the model under variable operating conditions is also tested. In general, all three methods are seen to maintain high accuracy levels. Table 2 shows the average accuracy rate, average standard deviation, and average macro f1 score for each model under these different load conditions. While Fine-Tuning and Replay+KD methods generally exhibit similar performance, higher inter-experimental variability was observed in the Replay method.

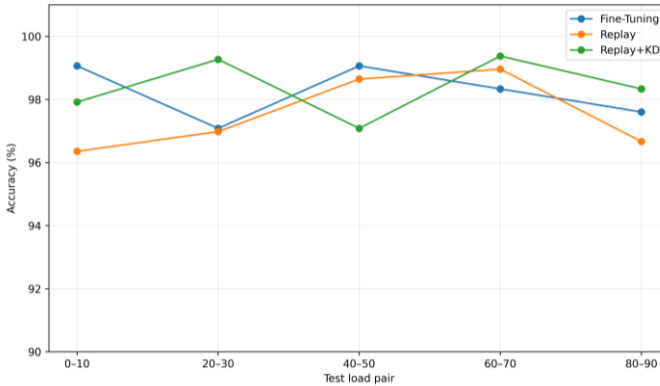


Figure 7 Accuracy across the five load-level test groups.

Table 2 Load-pair analysis of three different CL methods

Load pair	Method	Accuracy mean	Accuracy SD	Macro-F1 mean
0–10	Fine-Tuning	99.06%	1.13%	0.9906
0–10	Replay	96.35%	3.21%	0.9634
0–10	Replay+KD	97.92%	2.03%	0.9791
20–30	Fine-Tuning	97.08%	1.78%	0.9708
20–30	Replay	96.98%	2.37%	0.9698
20–30	Replay+KD	99.27%	0.79%	0.9927
40–50	Fine-Tuning	99.06%	1.13%	0.9906
40–50	Replay	98.65%	1.26%	0.9865
40–50	Replay+KD	97.08%	3.16%	0.9707
60–70	Fine-Tuning	98.33%	0.48%	0.9833
60–70	Replay	98.96%	0.36%	0.9896
60–70	Replay+KD	99.38%	0.62%	0.9937
80–90	Fine-Tuning	97.60%	1.30%	0.9760
80–90	Replay	96.67%	2.13%	0.9666
80–90	Replay+KD	98.33%	1.30%	0.9833

Three different CL methods were tested on different folds (5-fold) and different random seeds (42, 123, 2026) to prove that the models could achieve the same high performance on different experimental designs. Figure 8 shows the changes in accuracy of the models under different experimental running conditions, and Figure 9 shows the box-plot visualization of the range in which the accuracy values of the models changed despite different fold and seed settings.

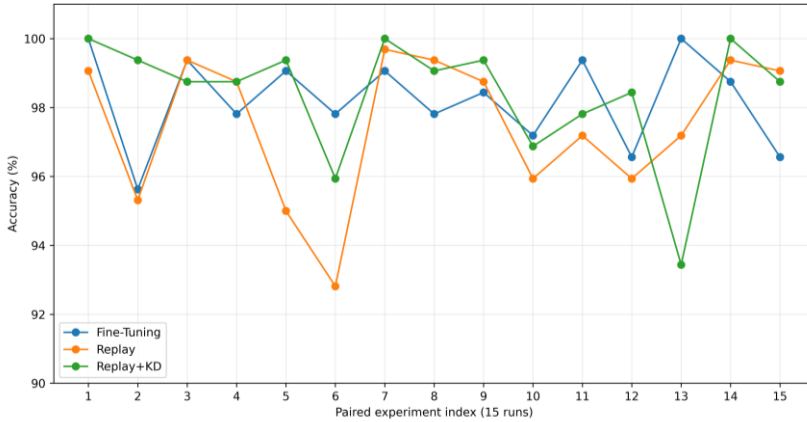


Figure 8 Accuracy across 5 folds $\times$ 3 seeds.

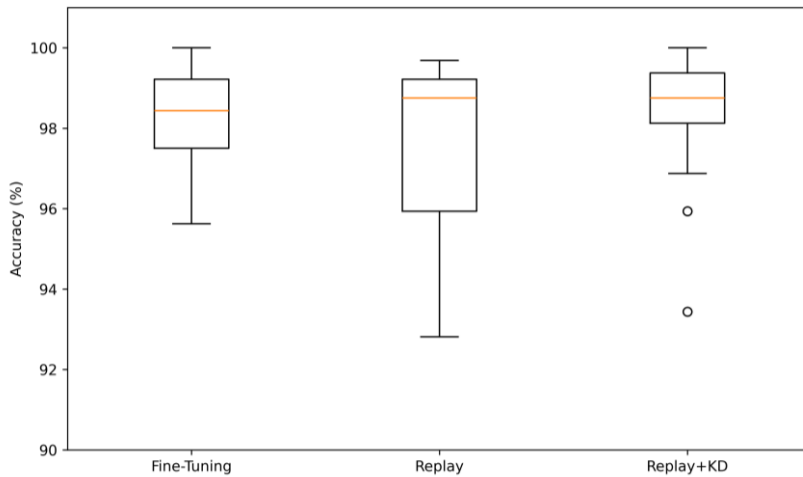


Figure 9 Distribution of accuracy across 15 paired runs.

To examine whether the results obtained from these three methods were statistically significant, the Wilcoxon signed-rank test and the Rank-biserial test were applied. In the comparison between Replay and Replay+KD, the mean difference was -0.88 percentile, and the Wilcoxon test result was $p = 0.0473$. Therefore, the performance superiority of Replay+KD over the basic Replay method is statistically significant. A p-value higher than 0.5 indicates that the result is significant. The Rank-biserial effect size of -0.600 also supports the finding of a significant effect in favor of the method in this comparison. In addition, no statistically significant difference was found between the Replay and Fine-tuning approaches. Figure 10 highlights the results of statistical significance.

Table 3 Comparison of CL methods based on statistical significance

Comparison	N	Mean diff.	Median diff.	Wilcoxon p	Rank-biserial
Fine-Tuning vs Replay	15	0.71 pp	0.31 pp	0.2996	0.314
Fine-Tuning vs Replay+KD	5	-0.17 pp	-0.94 pp	0.3619	-0.276

Replay vs Replay+KD	5	-0.88 pp	-0.62 pp	0.0473	-0.600
------------------------	---	----------	----------	--------	---------------

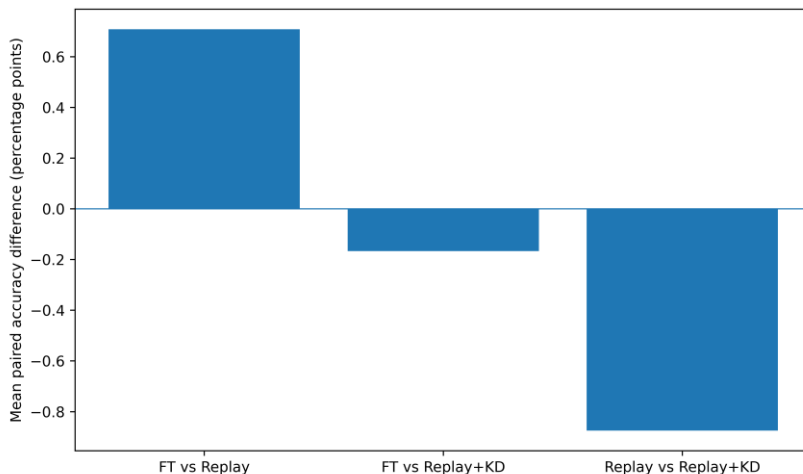


Figure 10 Paired model differences based on statistical significance

Another important criterion for CL methods is forgetting and backward transfer behavior. A negative value indicates that the final accuracy is higher than the initial accuracy, and therefore the model improves its performance rather than losing past task performance, in other words, it exhibits positive backward transfer. The average forgetting was found to be $-5.89 \pm 11.07\%$ for Fine-Tuning, $-4.74 \pm 9.59\%$ for Replay, and $-6.95 \pm 10.81\%$ for Replay+KD. The negative average forgetting values in all three methods indicate that the performance in previous tasks generally increased at the end of the continual learning process. Replay+KD having the most negative average value at -6.95% shows that the information distillation mechanism can contribute to the improvement of previous task performance in addition to preserving past task information.

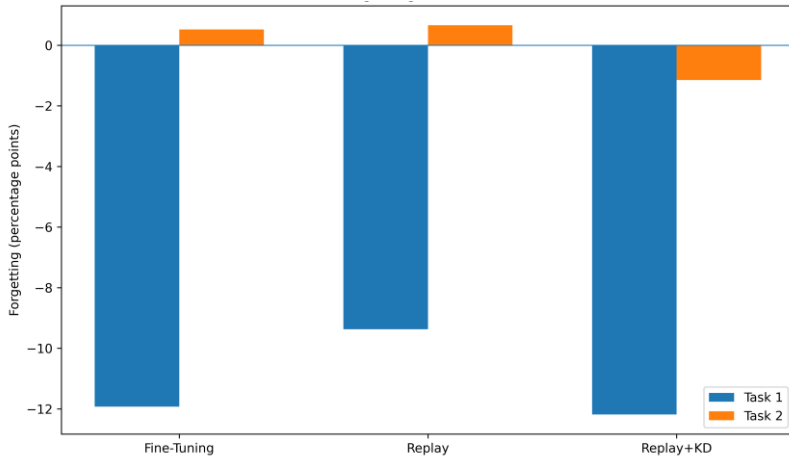


Figure 11 Task-wise forgetting

Table 4 Forgetting and backward performance across CL tasks

Metric	Fine-Tuning	Replay	Replay + KD
T1-mean	-11.93%	-9.37%	-12.19%
T1-SD	13.19%	12.01%	12.95%
T2-mean	0.52%	0.66%	-1.15%
T2-SD	2.87%	2.74%	4.01%
Avg mean	-5.89%	-4.74%	-6.95%
Avg SD	11.07%	9.59%	10.81%

Conclusions

This study evaluated three continual learning strategies—Fine-Tuning, Experience Replay, and Replay+KD—for two-class gearbox fault diagnosis under changing operating-load conditions. Using the same ResNet18 backbone, all approaches achieved high diagnostic performance, demonstrating their effectiveness in distinguishing healthy and broken-tooth conditions. Replay+KD achieved the highest overall accuracy ($98.40 \pm 1.79\%$) and Macro-F1 score (0.9839 ± 0.0179), slightly outperforming Fine-Tuning and clearly improving over standard Replay. Statistical analysis confirmed that Replay+KD significantly outperformed Replay ($p = 0.0473$), while its difference from Fine-Tuning was not statistically

significant. Furthermore, Replay+KD obtained the most favorable average forgetting value ($-6.95 \pm 10.81\%$), indicating beneficial backward transfer during sequential learning. These findings suggest that knowledge distillation effectively complements experience replay by improving continual learning performance without compromising diagnostic accuracy. Overall, Replay+KD provides a promising balance between adaptation, knowledge retention, and classification performance for gearbox fault diagnosis under varying operating conditions.

References

- Bojian, C., Changqing, S., Juanjuan, S. H. I., Lin, K. O. N. G., Luyang, T. A. N., Dong, W. A. N. G., & Zhongkui, Z. H. U. (2023). Continual learning fault diagnosis: A dual-branch adaptive aggregation residual network for fault diagnosis with machine increments. *Chinese Journal of Aeronautics*, 36(6), 361–377.
- Chen, B., Shen, C., Wang, D., Kong, L., Chen, L., & Zhu, Z. (2022). A lifelong learning method for gearbox diagnosis with incremental fault types. *IEEE Transactions on Instrumentation and Measurement*, 71, 1–10, Article 3514010. <https://doi.org/10.1109/TIM.2022.3177138>
- Ding, A., Yi, X., Qin, Y., & Wang, B. (2024). Self-driven continual learning for class-added motor fault diagnosis based on unseen fault detector and propensity distillation. *Engineering Applications of Artificial Intelligence*, 127, 107382.
- Du, N. T., Dien, N. P., & Nga, N. T. T. (2022). Gear fault detection in gearboxes operated in non-stationary conditions based on variable sideband analysis without a tachometer. *Frontiers in Mechanical Engineering*, 8, 1021222.
- Lin, S., & Zhao, H. (2026). A new fault diagnosis and incremental learning method for wind turbine gearbox operation stream data. *Engineering Applications of Artificial Intelligence*, 181, 115455.
- Lin, T., Song, L., Cui, L., & Wang, H. (2025). Continual learning for unknown domain fault diagnosis in rotating machinery via diffusion-integrated dynamic mixture experts. *Engineering Applications of Artificial Intelligence*, 156, 111056.
- Liu, X., Lei, Y., Li, N., Yang, B., Feng, K., & Shu, Y. (2025). Domain weighted distribution adaptation network: A novel remaining useful life prediction framework for machinery targeting time-varying operation conditions. *Advanced Engineering Informatics*, 68, 103794.

Pandya, Y., & Parey, A. (2018). *Gearbox fault diagnosis data* [Data set]. Prestige Institute of Engineering Management and Research (PIEMR), Open Energy Data Initiative (OEDI). <https://data.openei.org/submissions/623>

Risca, D., Lourenço, A., & Marreiros, G. (2025). Continual learning for rotating machinery fault diagnosis with cross-domain environmental and operational variations. *arXiv preprint arXiv:2504.10151*.

T., L. V., P. S., B., & Murugan, S. (2026). Robust vibration-based gearbox fault diagnosis: A review of transfer learning, domain adaptation, continual learning, and self-supervised methods. *Journal of Vibration Engineering & Technologies*, 14, 248. <https://doi.org/10.1007/s42417-026-02458-z>

Wei, D., Zuo, M. J., & Tian, Z. (2025). Continual learning for fault diagnosis considering variable working conditions. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 239(3), 584–595. <https://doi.org/10.1177/1748006X241252469>

Yang, Y., Zhou, J., Ding, X., Huai, T., Liu, S., Chen, Q., ... He, L. (2025). Recent advances of foundation language models-based continual learning: A survey. *ACM Computing Surveys*, 57(5), 1–38.

CHAPTER 9

AUTOMATED DETECTION OF PREDATORY JOURNALS: A SURVEY OF COMPUTATIONAL APPROACHES AND RELIABILITY CHALLENGES

Didem Yangeç¹

Abstract

Predatory journals are a major publishing problem that threatens the reliability of scholarly communication. They have become more visible with the growth of open access publishing. Recent approaches to their automatic detection draw on machine learning, natural language processing, web and visual analysis, explainable artificial intelligence, and large language models. However, differences in datasets, ground truth labels, feature representations, and validation strategies make it difficult to compare reported performance directly. This structured narrative review compares studies on automatic predatory journal detection in terms of their datasets, labeling strategies, computational methods, validation approaches, and reliability concerns. The synthesis identifies several key methodological challenges. These include inconsistent ground truth labels, class imbalance, false positives and false negatives, changes over time, external validation, and

¹ Öğr. Gör. Dr., Muğla Sıtkı Koçman Üniversitesi, Bilgi İşlem Daire Başkanlığı,
Orcid: 0009-0003-4882-9934

reproducibility. The generalizability of these systems across disciplines, geographical regions, and linguistic contexts also requires further assessment. The findings suggest that future research should look beyond classification performance. Greater attention should be given to datasets that combine multiple sources and include timestamps, evidence fusion, explainable models, and decision support with human oversight. Large language models are therefore recommended for extracting evidence, structuring information, and generating explanations, alongside independent validation and human oversight.

Keywords: Predatory journals, automatic detection, machine learning, explainable artificial intelligence, large language models

Introduction

Background and Motivation

The digitalization of scholarly communication and the growth of open access publishing have made scientific knowledge more accessible to researchers. These changes have also made the reliability of publishing practices an important concern. Predatory publishing is associated with misleading or opaque practices, inadequate editorial and peer review processes, and financial interests taking priority over scientific quality assurance. However, the literature offers no universally accepted operational definition of predatory publishing or agreement on the criteria that define it (Kakamad et al., 2024, 2026; Perera, 2025). Common indicators include opaque or inadequate peer review, inconsistencies in editorial board information, unverifiable indexing and impact metric claims, unclear or misleading article processing charges, aggressive publication invitations, and difficulties in verifying publisher information (Fung et al., 2026; Kakamad et al., 2024; Thallinger et al., 2026). Some of these features can be extracted automatically from journal websites, bibliographic sources, and publisher

communications. This has made predatory journal detection a subject of computational research using natural language processing, machine learning, and web mining.

Researchers' decisions to publish in predatory or questionable journals cannot be explained solely by a lack of individual awareness. In Türkiye, research has examined the relationship between performance evaluation systems based on metrics and questionable publishing choices (Öztürk & Taşkın, 2024). In Ghana, the role of academic promotion criteria in discouraging publication in questionable journals has also been assessed (Frandsen et al., 2024). A scoping review of health researchers in South Asia identified publication pressure, inadequate research support, and perceived immediate benefits as major factors (Kashyap et al., 2024). Awareness remains another concern. A rapid scoping review of medical students found limited awareness (Tomlinson, 2024), while a study involving 558 researchers in Jordan identified a need for education (Khabour et al., 2024). A survey in Brazil that received 3,067 responses also reported a significant association between publication pressure and experience with predatory publishing (do Carmo et al., 2026). The potential inclusion of studies published in predatory journals in evidence syntheses extends the problem beyond individual publication decisions (Martinino et al., 2025).

Labeling a journal as “predatory” is more than a technical classification task. Lists of predatory journals differ in their identification criteria, update frequency, transparency, and removal procedures (Kakamad et al., 2024, 2026). Ground truth labels are therefore a key methodological consideration in automatic detection studies. They directly affect how model results should be interpreted. Blacklists such as Beall's List are used to define the positive class. Sources such as DOAJ, Web of Science, Scopus, and PubMed may serve as comparison sources or help define the negative class in

some studies (Richtig et al., 2023). However, overlaps between blacklists and bibliographic indexes have been documented (Richtig et al., 2023). The “predatory” label may also unfairly discredit emerging journals (Perera, 2025). These concerns suggest that such sources should not be treated as fixed or universal ground truth. The reliability of automatic detection systems thus depends on both the algorithms used and the sources and criteria used to assign class labels.

Need for Automated Detection

Traditional journal assessment relies largely on manual review by researchers, librarians, or subject specialists. However, this approach is difficult to scale across large numbers of journals whose websites, editorial teams, indexing claims, and publishing policies change over time. The lack of standardized lists and criteria for identifying predatory journals can also make assessments sensitive to the criteria applied and the publishing context (Kakamad et al., 2024; Perera, 2025). Researchers, editors, libraries, academic institutions, funders, and publishers all continue to play a role in addressing predatory publishing (Laine et al., 2025). Automatic systems should therefore serve as scalable decision support tools that assist human assessment. They can screen large numbers of journals, identify risk indicators, and prioritize cases that require closer review.

One computational example is the Academic Journal Predatory Checking (AJPC) system developed by Chen et al. (2023). It combines web data collection, feature extraction, and classification using machine learning. More recently, Zhuang et al. (2025) assessed website content and design alongside publication metadata. Their study showed how automatic screening could draw on multiple data sources. Automatic detection is thus moving toward

computational screening that combines several sources of evidence and supports expert review.

Advancement of Computational Approaches

Computational methods for automatic predatory journal detection now include machine learning using metadata and text, natural language processing, website and visual analysis, explainable machine learning, large language models, and behavioral evidence analysis. Chen et al. (2023) evaluated textual features from journal websites using several classification algorithms. Zhuang et al. (2025) combined website content and design with bibliometric indicators to develop an approach drawing on multiple data sources. Another area of development concerns the explainability of model outputs and the use of large language models in journal assessment. Wu et al. (2024) used interpretable machine learning and SHAP feature contributions to identify indicators associated with model predictions. Al-Moghrabi et al. (2024) empirically examined ChatGPT's performance in assessing biomedical and dental journals. These studies show growing interest in the evidence behind predictions and the reliability of new artificial intelligence tools, alongside classification performance.

The evidence available for computational assessment extends beyond static web content. Patterns in publication invitations and indexing claims (Fung et al., 2026; Thallinger et al., 2026), unusual editorial board structures (Downes, 2024), and active tests of peer review, fees, communication practices, and observable publisher behavior (Burgiss et al., 2024) provide additional sources of evidence. The field is therefore moving beyond binary classification based on a single data type toward a broader computational framework that can assess evidence from multiple sources.

Scope and Contributions of This Review

Studies on automatic predatory journal detection vary considerably in their datasets, labeling strategies, feature representations, algorithms, and evaluation protocols. The positive class may be defined using blacklists, delisting status, or expert assessment (Chen et al., 2023; Kakamad et al., 2024; Zhuang et al., 2025). The negative class is also defined using different sources. These differences make reported accuracy, precision, recall, and F1 scores difficult to compare directly. Strong performance on internal tests does not guarantee the same reliability in practice. Independent evaluations of AJPC have shown that classification results can vary across journal groups and assessment times (Teixeira da Silva & Kendall, 2023; Teixeira da Silva & Scelles, 2024; Teixeira da Silva et al., 2025). Using different web addresses for the same journal can also produce different outputs (Teixeira da Silva, 2025). These findings show that automatic detection systems require assessment beyond algorithmic performance. Ground truth labels, data sources, external validation, stability over time, input sensitivity, and reproducibility must also be considered.

Some studies of automatic journal assessment focus on specific disciplines. For example, Al-Moghrabi et al. (2024) examined assessment using a large language model in biomedical and dental journals. Other approaches use broader journal collections (Chen et al., 2023; Zhuang et al., 2025). However, overall performance on a broad dataset does not establish comparable reliability across health sciences, engineering, social sciences, and other disciplines. An important research need is therefore to develop automatic assessment systems that can be used across disciplines and validate their performance within each discipline using independent datasets.

This review examines these methodological differences and the need for generalizability across disciplines within a common framework. Its main contributions are to compare datasets and strategies for establishing ground truth; classify machine learning, natural language processing, web and visual analysis, explainable artificial intelligence, large language models, and behavioral approaches within this framework; and assess reported performance in relation to differences in data and labels. It also identifies key challenges involving external validation, changes over time, reproducibility, and misclassification. Finally, it discusses explainable decision support approaches that draw on evidence and retain human oversight. The review addresses two main research questions:

RQ1: How do ground truth definitions, data sources, feature representations, machine learning methods, and validation strategies affect the reliability of automatic predatory journal detection?

RQ2: What findings and limitations does the existing empirical literature report regarding the use of large language models in automatic journal assessment?

Guided by these questions, the review evaluates methods in relation to both their reported performance and the data, labels, evidence, and validation conditions under which their results were obtained.

Methodology

Review Design and Scope

This study is a structured narrative review of the computational literature on automatic predatory journal detection. It examines datasets, labeling strategies, feature representations, methods, and validation approaches. The aim is to identify, classify, and critically synthesize the main computational approaches in this

methodologically diverse field. The primary scope covers studies that automatically or semi automatically identify predatory or questionable publishing at the journal or publication venue level. Approaches using machine learning, deep learning, natural language processing, explainable artificial intelligence, large language models, web and visual analysis, and multiple data sources were considered.

Predatory journals were treated as conceptually distinct from hijacked or cloned journals because they represent different phenomena. However, studies on the computational detection of hijacked or cloned journals were included as related literature because of their methodological similarities. Studies providing empirical evidence on ground truth labels, external validation, stability over time, input sensitivity, and behavioral or structural indicators were considered supporting literature. The evidence base therefore comprised three groups: (i) core studies that directly develop automatic detection or classification methods, (ii) related studies on the computational detection of hijacked or cloned journals, and (iii) supporting studies that provide evidence on ground truth, external validation, or detection features.

Information Sources and Search Strategy

The literature search was conducted using Scopus, Web of Science, Google Scholar, and IEEE Xplore. It focused on automatic detection of predatory and questionable journals, alongside methodologically related studies on hijacked and cloned journal detection. Topic terms were combined with computational method terms. Topic terms included “predatory journal”, “predatory publishing”, “predatory publishing venue”, “questionable journal”, “hijacked journal”, “clone journal”, and “journal hijacking”. Method terms included “automatic detection”, “classification”, “machine learning”, “deep learning”, “artificial intelligence”, “natural

language processing”, “NLP”, “large language model”, “ChatGPT”, “BERT”, “explainable artificial intelligence”, “web analysis”, “domain analysis”, and “network analysis”. Boolean operators were used to combine these terms. Separate sets of search queries were developed for predatory or questionable journals and for hijacked or cloned journals.

An example of a general search string was: (“predatory journal” OR “questionable journal” OR “predatory publishing venue”) AND (“automatic detection” OR “classification” OR “machine learning” OR “deep learning” OR “natural language processing” OR “artificial intelligence” OR “large language model” OR “ChatGPT” OR “BERT”). Searches for hijacked or cloned journals combined (“hijacked journal” OR “clone journal” OR “journal hijacking”) with method terms such as “detection”, “classification”, “machine learning”, “artificial intelligence”, “large language model”, “web”, “domain”, and “network analysis”. The electronic search was supplemented by backward and forward citation tracking. Reference lists of key included studies and relevant publications citing them were examined. This helped identify studies that used different terminology but remained relevant to the research questions.

Inclusion and Exclusion Criteria

Studies were assessed according to their relevance to the research questions and the type of evidence they provided for automatic journal assessment. Eligibility criteria were applied consistently with the core, related, and supporting literature groups defined in the Review Design and Scope section. A study was included if it met at least one of the following conditions:

1. It presented a computational method for automatically or semi automatically identifying or assessing predatory or questionable journals or publication venues. Eligible methods

included machine learning, deep learning, natural language processing, large language models, explainable artificial intelligence, web or visual analysis, bibliometric analysis, or combinations of these approaches.

2. It developed a computational or systematic digital method for detecting hijacked or cloned journals.

3. It empirically assessed the external validity, stability over time, or input sensitivity of an existing automatic predatory journal detection system.

4. It provided empirical findings that could directly inform ground truth construction, feature extraction, or evidence validation in automatic detection systems.

Studies concerned only with researcher awareness, publication pressure, institutional policy, or the general causes and consequences of predatory publishing were excluded from the comparative methodological synthesis. Where needed, they were used only to explain the motivation for the study or its sociotechnical context. General studies on fabricated articles and paper mills were also excluded from the core review unless they directly addressed detection at the journal or publisher level.

Duplicate records were removed. Records lacking sufficient bibliographic information to assess their methods or findings were excluded from the main comparative synthesis. For studies available only as titles or abstracts, methodological details that could not be verified were omitted from the comparison tables.

Study Selection

Study selection proceeded in stages. Titles and abstracts were first assessed against the research questions and inclusion criteria. Records clearly outside the scope were excluded. Full texts or

detailed publication records of potentially eligible studies were then examined. The final selection followed the inclusion and exclusion criteria defined in the Inclusion and Exclusion Criteria section.

The included studies were organized into the three evidence groups defined in the Review Design and Scope section. Core computational studies directly examined predatory or questionable journal detection. Related studies addressed the computational detection of hijacked or cloned journals. Supporting studies provided evidence on ground truth, external validity, and behavioral, editorial, or bibliometric indicators. However, the review records did not systematically document the total numbers of retrieved records, duplicates, records screened by title and abstract, or publications assessed in full text. These counts were therefore not estimated retrospectively, and no study selection flow diagram was presented.

Data Extraction and Study Classification

For core computational studies, information used in the comparative synthesis was limited to data explicitly reported in the source publications. Where available, the extracted information covered the unit of analysis, dataset size and class distribution, label sources and meanings, data and feature types, algorithms or models, training and validation approaches, and reported performance metrics. Values not explicitly provided in the source were not estimated. Accuracy, precision, recall or sensitivity, F1, specificity, ROC AUC, and PR AUC were recorded only as reported in each study. Because the studies used different datasets and target class definitions, these values were not used to rank methods directly. Label meaning was also treated as a separate comparison dimension. Labels based on blacklist membership, removal from an index, expert assessment, or another protocol were not assumed to represent the same target variable.

For related and supporting studies, extraction focused on the sample or unit of analysis, evidence type, assessment approach, main findings, and methodological contribution to automatic detection. These studies were assessed separately to clarify methodological challenges and additional sources of evidence. They were not ranked alongside the performance results of core classifiers.

Synthesis Approach

A quantitative meta analysis was not conducted because the datasets, target class definitions, feature representations, and evaluation protocols could not be directly combined. A structured comparative and narrative synthesis was used instead. Reported performance values were therefore neither pooled as a common effect size nor used to rank algorithms across studies. The synthesis proceeded at two levels. First, studies were described by their data and method types. These categories covered metadata and text analysis using natural language processing, web and visual analysis, conventional or explainable machine learning, large language models, and behavioral or multimodal approaches. Second, shared methodological challenges were compared across studies. These included ground truth heterogeneity, class imbalance, the balance between false positives and false negatives, external validation, changes over time, input sensitivity, reproducibility, explainability, and generalizability across contexts.

Following this distinction, the Datasets and Ground Truth Labels and Automatic Detection Methods sections describe datasets, labeling approaches, and computational methods. Their shared implications for reliability are addressed in the subsequent comparative synthesis section. Findings on large language models were interpreted within the limits of the available empirical evidence. Results from direct assessments of predatory or

questionable journals were kept conceptually separate from those concerning hijacked or cloned journals.

Datasets And Ground Truth Labels

Interpreting model performance in automatic predatory journal detection requires attention to both the algorithm and the data and class definitions used for training. A machine learning model learns the labels assigned to its training data. The criteria used to define classes such as “predatory”, “questionable”, and “legitimate” therefore directly determine what its outputs mean. Dataset size and class distribution must be considered alongside label sources, assignment dates, evidence types, and intended meanings.

Blacklist-Based Labeling

Blacklists are commonly used to define the positive class in automatic predatory journal detection. Journals identified as predatory, potentially predatory, or suspicious by a particular list or assessment process are treated as positive examples. Beall’s List has served this purpose in several computational and empirical studies (Al-Moghrabi et al., 2024; Burgiss et al., 2024; Chen et al., 2023; Richtig et al., 2023; Siess, 2025). Bin Ateeq and Al-Khalifa (2023) constructed their dataset using the DOAJ directory, lists of journals rejected or removed by DOAJ, and Beall’s List. Of the 9,866 journals collected, 5,533 were classified as legitimate and 4,333 as questionable. Classification experiments used the 6,836 journals for which web content was available.

Blacklist membership should not be treated as fixed or undisputed ground truth. Richtig et al. (2023) reported that some journals associated with Beall’s List also appeared in bibliographic and indexing systems. These included 570 journals in Scopus, 224 in DOAJ, and 50 in Web of Science (Richtig et al., 2023). This finding shows that blacklist membership and inclusion in an index

or bibliographic system are not necessarily mutually exclusive. The positive class derived from a blacklist should therefore be understood as journals labeled predatory or suspicious under that list's criteria. It does not establish that every journal is definitively predatory. This distinction matters when comparing models trained on different datasets. Kakamad et al. (2024) also showed that lists of predatory journals and publishers vary in their inclusion and exclusion criteria, transparency, and updating procedures. Blacklists are useful label sources, but they should not be treated as flawless ground truth on their own.

Whitelists and Indexing Databases

Some studies define the negative or “legitimate” comparison class using whitelists and indexing sources. Chen et al. (2023) compiled their list of legitimate journals from the BIH QUEST resource, which draws on DOAJ and PubMed Central data. Al-Moghrabi et al. (2024) selected comparison journals from Web of Science and DOAJ. However, treating inclusion in a particular index or whitelist as a universal indicator of legitimacy is a methodological assumption. Richtig et al. (2023) showed that some journals associated with Beall's List also appeared in Scopus, DOAJ, PubMed, PubMed Central, and Web of Science. Blacklist membership and indexing therefore do not always define mutually exclusive classes (Richtig et al., 2023). Similarly, Chen et al. (2023) noted that the absence of a new journal from sources such as DOAJ or Web of Science should not automatically be interpreted as evidence of predatory publishing.

Datasets derived from whitelists or indexes should therefore clearly identify the source behind each label. Where appropriate, descriptions such as “journals listed in DOAJ” or “journals selected from Web of Science” provide greater clarity than the broad label

“legitimate journals”. Making the source explicit allows differences in labels across studies to be interpreted more accurately.

Hybrid and Expert-Based Labeling

Some studies combine labels derived from lists with independent expert assessments. This allows model results to be examined against a separate reference set. It also makes it possible to assess different evidence sources separately, without assuming that list membership and expert judgment represent the same labeling process. Zhuang et al. (2025) classified 12,869 journals that remained on the DOAJ whitelist as “legitimate”. They classified 2,536 journals removed for guideline violations or “questionable practices” as “questionable”. The study also used a separate expert assessment set of 360 journals from Unpaywall that DOAJ had not previously classified as whitelisted or delisted. These journals were assessed by a librarian specializing in open access publishing and two of the study’s authors (Zhuang et al., 2025).

This design makes the distinction between ground truth derived from lists and expert assessment explicit. Expert assessment is not flawless ground truth either. The criteria, assessment protocol, and agreement between assessors can all affect how results should be interpreted. Hybrid datasets and those incorporating expert assessment should therefore report both the final labels and the evidence and assessment procedures used to assign them.

De-listing-Based Labeling

Delisting is another source of labels in some studies. Zhuang et al. (2025) did not classify every journal removed from DOAJ as questionable. They first excluded journals removed for legitimate reasons, such as publication ending or voluntary withdrawal. The remaining journals were assigned to the “questionable” class. This approach accounts for changes in a journal’s assessment status,

whereas a static blacklist does not. However, delisting and predatory publishing are not synonymous. As illustrated by Zhuang et al. (2025), the source’s “questionable” label should therefore be retained rather than generalized to “predatory”.

This distinction shows that models trained under different labeling protocols may perform technically similar classification tasks while predicting different concepts. Model performance must therefore be interpreted alongside the source and intended meaning of the labels.

Behavioral Evidence-Based Data Sources

Labels derived from lists generally reflect third party assessments. Other studies directly examine observable journal or publisher behavior and offer complementary evidence for automatic detection. Burgiss et al. (2024) used active open source intelligence to examine 256 suspicious journals selected from Beall’s and Kscien lists. They assessed peer review, fees, location, and communication patterns. They also submitted five articles generated with ChatGPT 4 to the target journals. They reported that 55 journals accepted or published these articles, or both (Burgiss et al., 2024). This design made it possible to observe actual publisher behavior alongside claims made on publisher websites.

Fung et al. (2026) examined 422 publication invitation emails sent to a single academic. They checked claims about impact factors, indexing, and ISSNs against independent sources. Thallinger et al. (2026) assessed 188 invitation emails in a prospective analysis over two months. They examined communication features such as the alignment between the invitation topic and the recipient’s expertise. These indicators should not be treated as class labels on their own. Their value lies in adding distinct and independent evidence that can be considered alongside lists and other sources.

Class Imbalance and Changes Over Time

Class distribution affects the interpretation of model performance alongside the labeling process. Chen et al. (2023) used a dataset containing 833 predatory and 1,213 legitimate journals. Zhuang et al. (2025) used a substantially imbalanced dataset of 2,536 questionable and 12,869 legitimate journals. Reported performance must therefore be assessed in relation to each dataset's class distribution. Accuracy alone may be insufficient when the positive class is the minority. Zhuang et al. (2025) explicitly acknowledged class imbalance and used the area under the precision recall curve, or PR AUC, to evaluate their models. Performance metrics should therefore be interpreted with particular attention to the balance between identifying minority class cases and producing false positives.

The temporal characteristics of datasets also require attention. Journal websites, editorial information, indexing status, and list membership can change over time. Features and labels collected on one date cannot be assumed to represent a later period equally well. This review treats such changes as a methodological concern. Reporting the data collection date, the period for which labels are valid, and, where possible, the dataset version is important for assessing reproducibility and validity over time.

Why Is There No Universal Ground Truth?

Taken together, the datasets used in the literature reveal at least six labeling approaches based on different sources and protocols. They do not provide a single universal ground truth for automatic predatory journal detection:

1. Labeling based on blacklists: Journals appearing on Beall's List or similar sources are used as the positive class (Burgiss et al., 2024; Richtig et al., 2023).

2. Labeling based on whitelists or indexes: Journals selected from DOAJ, Web of Science, or similar sources are used as the comparison or negative class (Al-Moghrabi et al., 2024; Chen et al., 2023).

3. Expert assessment: Independent assessments by librarians or researchers are used to establish ground truth (Zhuang et al., 2025).

4. Labeling based on delisting: Removal from a list or index is used to define a journal's class (Zhuang et al., 2025).

5. Assessment based on behavioral evidence: Observable practices, such as article acceptance behavior, verifiable publishing claims, and publisher communications, are used as evidence (Burgiss et al., 2024; Fung et al., 2026; Thallinger et al., 2026).

6. Hybrid labeling: Multiple lists or indexing sources are used together. Bin Ateeq and Al-Khalifa (2023) combined the DOAJ directory, lists of journals rejected or removed by DOAJ, and Beall's List to construct their dataset.

These categories do not necessarily represent the same target variable. Labels identifying journals as "listed on Beall's List", "removed from DOAJ", "judged questionable by experts", or showing behavioral risk indicators can carry different meanings depending on the underlying protocol. Reviews of predatory publishing lists have documented differences in inclusion and exclusion criteria, transparency, updating procedures, and removal processes (Kakamad et al., 2024, 2026). This review therefore treats ground truth as the labeling protocol that defines the target class and the evidence supporting it. Comparisons across studies should jointly consider the data source, unit of analysis, class definitions and label meanings, labeling protocol, data collection date, class

distribution, and independent validation status. This makes the conditions behind reported performance clearer.

Automatic Detection Methods

Studies on automatic predatory journal detection differ considerably in the types of data they use and how they represent those data. Early studies mainly used textual features extracted from journal websites and conventional machine learning algorithms. More recent research incorporates website design, visual features, bibliometric indicators, editorial board information, behavioral signals, explainable artificial intelligence, and large language models (LLMs). This section examines these approaches according to their methods and evidence types.

Conventional Machine Learning Approaches

Conventional machine learning approaches classify journals using features extracted from their websites or journal information. The Academic Journal Predatory Checking system developed by Chen et al. (2023) is a key example. Chen et al. (2023) combined data collection, feature extraction, and model prediction in a single system. Their dataset contained 833 predatory and 1,213 legitimate journals. They allocated 80% of the data to training and 20% to testing. Website texts were preprocessed, and Bag of Words and TF IDF representations were assessed alongside diff scores. Eight classifiers were compared. Random Forest produced the best results, with a reported recall of 0.982 and F1 score of 0.980.

Bin Ateeq and Al-Khalifa (2023) compared conventional machine learning, deep learning, and transformer models within the same framework. They used SVM, KNN, a neural network, LSTM, CNN, BERT, and ALBERT. The CNN model achieved an F1 score of 0.96 for journal classification. For identifying reasons related to predatory publishing criteria, SVM achieved a micro F1 score of

0.67. However, the performance of these models depends on their datasets and labels. Internal test results alone do not establish external validity. Independent evaluations of AJPC have reported problematic classifications across different journal groups (Teixeira da Silva & Kendall, 2023; Teixeira da Silva & Scelles, 2024; Teixeira da Silva et al., 2025). These findings show that automatic detection systems also need testing beyond their training data.

Natural Language Processing and Text-Based Analysis

Journal websites contain textual information about scope, peer review, publication ethics, fees, and indexing claims. Natural language processing can use this content for automatic detection. Chen et al. (2023) classified preprocessed web texts using Bag of Words and TF IDF representations. Bin Ateeq and Al-Khalifa (2023) assessed textual content using conventional methods such as SVM and KNN, alongside LSTM, CNN, BERT, and ALBERT. Text analysis has therefore been explored through methods ranging from word representations to contextual language models.

The presence of a claim in a text does not establish its accuracy. An indexing or impact factor claim remains a website statement unless verified against an independent source. Fung et al. (2026) compared claims about indexing, impact factors, ISSNs, and PubMed in publication invitations with verifiable sources. Their study shows the importance of assessing whether textual claims can be verified, alongside identifying their presence. These findings suggest that checking extracted claims against external sources is a useful research direction for NLP systems. This could extend text analysis beyond word patterns to include the verifiability of claims.

Website and Visual Analysis Approaches

Website structure and visual features can also inform automatic journal assessment. Zhuang et al. (2025) combined

website content, design features, and publication metadata to predict questionable open access journals. They used information on aims and scope, editorial boards, open access, copyright, and ethics policies, alongside homepage screenshots and website source code (Zhuang et al., 2025). This brought textual content and online presentation into a common assessment framework.

In the related literature, Andoohgin Shahri et al. (2018) focused on hijacked journals, which are conceptually distinct from predatory journals. They examined 104 websites representing 59 genuine and 45 hijacked journals and used a decision tree classification approach. Features related to domain names and websites were assessed. The resulting decision tree was also tested on a separate sample of five genuine and five hijacked journals. This study provides an early example of using website features as computational evidence for a related form of journal fraud. Abalkina (2021) examined hijacked journals through repeated content in the archives of cloned websites. Article titles, author names, and affiliations were searched using the Google Custom Search API. The method identified 62 cloned websites associated with 57 genuine journals. The study also reported that two cloned websites were anticipated before they became active. Although it did not directly classify predatory journals, it showed how archive similarities and relationships between websites could support hijacked journal detection.

Siess (2025) approached visual features differently by examining published article PDFs rather than journal websites. The dataset contained 443 legitimate and 555 potentially predatory publications. Features included typography, PDF metadata, page layout, visual complexity, and white space. The study treated these features as complementary indicators for large scale screening rather than definitive classification criteria. Website design and visual quality should therefore be interpreted alongside other evidence,

without assuming that they directly reflect publishing quality. Input sensitivity also matters in website assessment. Teixeira da Silva (2025) reported that entering URLs for different pages of the same journal into AJPC could produce different classifications. This finding highlights the need to report the URLs and page content used in automatic website assessment.

Explainable Machine Learning Approaches

Automatic detection requires attention to both the predicted class and the features behind the prediction. Explainable artificial intelligence (XAI) methods help examine how individual features contribute to model outputs. Wu et al. (2024) developed an interpretable machine learning approach to assess early warning risk for predatory open access journals. They used several machine learning algorithms and interpreted model outputs with SHAP. This allowed them to examine the influence of indicators contributing to predictions. Such explanations can support decision making with human oversight. Users can inspect the indicators influencing a risk assessment and use this information to review unexpected results.

Explainability, however, does not establish accuracy. SHAP can show how a feature contributes to a model prediction. It does not prove that the feature alone is a reliable indicator of predatory publishing. Explanations must therefore be assessed alongside label quality and the validation approach.

Large Language Model-Based Approaches

The ability of large language models to interpret information and generate explanations has introduced a new approach to journal assessment. Al-Moghrabi et al. (2024) examined ChatGPT's performance in classifying biomedical and dental journals as predatory or legitimate. The study assessed 400 journals, comprising 200 predatory and 200 legitimate journals. The predatory group was

selected from Beall's List and unsolicited publication invitations. The legitimate group was selected from Web of Science and DOAJ (Al-Moghrabi et al., 2024). Journal names were queried using the free version of ChatGPT available in May 2023. Additional prompts and journal website information were provided when needed. Responses were coded under ten criteria.

ChatGPT correctly classified 92.5% of journals labeled predatory and 71% of those labeled legitimate. Reported overall accuracy was 0.82, sensitivity was 0.93, and specificity was 0.71. The false positive rate was 29%, while the false negative rate was 7.5% (Al-Moghrabi et al., 2024). Performance therefore differed between the two classes. The findings are also limited to the disciplines, reference sources, and ChatGPT version examined.

Hegedűs et al. (2026) provided complementary evidence on LLM systems through a study of hijacked medical journals. This study did not directly examine predatory journal classification. It assessed 21 previously identified and 10 newly identified hijacked journals alongside their genuine counterparts. ChatGPT, Gemini, Copilot, DeepSeek, Qwen, Perplexity, and Claude were compared. The systems could provide general information but could not reliably distinguish genuine journals from hijacked versions. Copilot performed better but still made errors (Hegedűs et al., 2026). Together, these two studies indicate that LLM assessments require careful testing before they can serve as final verification tools. A use that allows closer oversight would involve extracting evidence and supporting human assessment. This recommendation reflects the methodological synthesis developed in this review from the available findings.

Behavioral and Multimodal Approaches

Static website features reflect the information a journal presents about itself. Behavioral evidence adds information about

how a publisher operates in practice. Relevant sources include publication invitations, editorial board patterns, article acceptance behavior, and practices after publication. Burgiss et al. (2024) actively investigated 256 suspicious journals selected from Beall's and Kscien lists. The researchers submitted five articles generated with ChatGPT 4 to these journals. The articles were accepted or published by 55 journals. Peer review processes, fees, communication patterns, and relationships between publishers were also assessed (Burgiss et al., 2024). The study shows how observed publisher behavior can provide evidence alongside website statements.

Editorial board and publication invitation data offer further complementary evidence. Downes (2024) identified 96 individuals who appeared at least 20 times on the editorial boards of predatory publishers. The study identified different circumstances, including identity theft and voluntary membership. Frequent editorial board membership should therefore not be interpreted as proof of predatory publishing on its own. Fung et al. (2026) checked claims about indexing, impact factors, and ISSNs in publication invitations against independent sources. Thallinger et al. (2026) examined 188 publication invitation emails and reported that 148 did not match the recipient's expertise.

Practices after publication provide another layer of evidence. Xu et al. (2026) examined retraction practices among 1,511 potentially predatory journals on the updated Beall's List. Of these, 46 had retracted a total of 645 publications by the end of 2022. The study assessed the visibility of retraction policies, document accessibility, and the information provided in retraction notices (Xu et al., 2026). Such behavioral evidence can complement other data sources. However, tests involving active article submission can raise ethical and operational concerns. Behavioral indicators should

therefore be assessed alongside other evidence rather than treated as ground truth on their own.

Overall Comparison

Core computational studies differ in their datasets, label definitions, features, methods, and evaluation strategies. Their reported performance values are therefore not directly comparable. Table 1 summarizes studies that directly apply computational methods to predatory or questionable journal detection, alongside label meanings and validation approaches.

Table 1. Comparison of core computational studies on predatory or questionable journal detection

Study	Dataset / N	Labels / ground truth	Data / features	Method	Evaluation	Main finding / limitation
Bin Ateeq and Al-Khalifa (2023)	9,866 journals collected; web content from 6,836 journals used in classification experiments	DOAJ, journals rejected or removed by DOAJ, and Beall's List	Journal website texts; different text representations	SVM, KNN, NN, LSTM, CNN, BERT, ALBERT	Best classification: CNN, F1 = 0.96; violation identification or explanation task: SVM, micro F1 = 0.67	Web and NLP approach; dependent on label and web content sources
Chen et al. (2023)	2,046 journals: 833 predatory, 1,213 legitimate	Beall's List and Stop Predatory Journals; legitimate: DOAJ and PMC	Web texts; BOW, TF IDF, diff scores	Eight classifiers; RF selected	80% training, 20% testing; recall = 0.982, F1 = 0.980	Strong internal performance; dependent on lists and web content
Wu et al. (2024)	434 medical and 194 engineering observations; class ratio 1:1	CAS Early Warning List; negative class: open access or hybrid journals selected using SJR	17 bibliometric or journal indicators	LR, SVM, DT, RF, XGBoost, and SHAP	70%/30% split; cross validation with five folds; XGBoost: accuracy = 0.979, F1 = 0.979, AUC = 0.990	Strong internal performance; labels represent early warning or risk status
Al-Moghrabi et al. (2024)	400 journals: 200 predatory, 200 legitimate	Beall's List and unsolicited emails; legitimate: WoS/SCIE and DOAJ	ChatGPT responses; ten indicator categories	ChatGPT; RF/GB for criteria analysis	Accuracy = 0.818; sensitivity = 0.925; specificity = 0.710	FP = 29%, FN = 7.5%; assessed at one time point and limited to specific disciplines
Zhuang et al. (2025)	15,405 journals: 12,869 legitimate, 2,536 questionable	Journals remaining in DOAJ or removed for questionable practices	Web content, design, bibliometrics	RF; CNN for visual representations	Cross validation with five folds; PRC AUC = 0.79 ± 0.03	Combined feature model achieved the highest PRC AUC; expert review recommended

Source: Author's synthesis of the studies summarized in this table.

Table 1 shows why methodological differences must be considered alongside differences in data and labels. Bin Ateeq and Al-Khalifa (2023) combined several reference sources, while Chen et al. (2023) used labels derived from lists and website texts. Al-Moghrabi et al. (2024) assessed journals using ChatGPT. Zhuang et al. (2025) used a questionable journal label based on DOAJ delisting history. Wu et al. (2024) also examined feature contributions associated with model predictions. These studies used different target variables and validation designs. Their performance values should therefore not serve as a common ranking measure.

The methods also differ in their evidence sources and representations. Table 2 summarizes the main method families considered in this review by data source, typical processing approach, strengths, and limitations.

Table 2. Comparison of main method families in automatic predatory journal detection

Approach	Main data source	Typical features / processing	Strength	Main limitation
Metadata analysis	ISSNs, publishers, indexing information, bibliometric records	Structured journal and publisher features	Scalable and relatively easy to verify	Depends on database coverage and currency
Text analysis / NLP	Journal web pages, policy texts, invitation emails	BoW, TF IDF, textual patterns, claim extraction	Enables automatic processing of large amounts of unstructured content	The presence of a claim does not establish its accuracy
Web / visual analysis	Web pages, screenshots, PDF documents	Website design, visual representations, PDF and typography features	Adds signals beyond text	Design quality does not directly establish publishing quality
Conventional ML / deep learning	Structured and extracted features, web texts	SVM, RF, XGBoost, CNN, LSTM	Enables scalable classification of large datasets	Depends on label and training data quality
Explainable artificial intelligence	Model inputs and prediction outputs	SHAP and feature contribution analysis	Allows examination of features associated with predictions	Feature contributions do not establish causality
Large language models	Web texts and journal information expressed in natural language	Classification, information extraction, explanation, summarization	Can process complex and unstructured information in natural language	Sensitivity to model version, prompts, and context; risk of generating inaccurate information
Behavioral / multimodal approaches	Publication invitations, editorial boards, practices after publication, and external sources	Behavioral patterns, claim verification, evidence integration	Can provide complementary evidence beyond journal statements	Data collection is costly, and equivalent evidence may not be available for every journal

Source: Author's synthesis of the studies reviewed in the Automatic Detection Methods section.

The method families in Table 2 focus on different evidence types. Conventional machine learning and NLP support scalable classification using web texts (Chen et al., 2023). Web and visual analysis incorporate structural and visual features (Siess, 2025; Zhuang et al., 2025). Explainable machine learning makes feature contributions associated with predictions visible (Wu et al., 2024). Assessments using LLMs provide empirical evidence on interpreting journal information through natural language (Al-Moghrabi et al., 2024). Behavioral studies add evidence from publication invitations, editorial board patterns, acceptance behavior, and practices after publication (Burgiss et al., 2024; Downes, 2024; Fung et al., 2026; Thallinger et al., 2026; Xu et al., 2026).

These differences make direct performance rankings inappropriate. Chen et al. (2023), Al-Moghrabi et al. (2024), and Zhuang et al. (2025) used different datasets, class definitions, and evaluation protocols. A higher accuracy or F1 score alone therefore does not establish that one method is generally superior. The comparison in this section considers four dimensions: data and evidence types, the source and meaning of ground truth labels, validation strategies, and explainability and reproducibility. the Comparative Analysis And Open Challenges section examines the methodological implications of these dimensions in greater detail.

Comparative Analysis And Open Challenges

The studies examined in previous sections show that automatic predatory journal detection can draw on diverse data sources and computational methods. However, reliability in practical use depends on more than algorithm selection. Ground truth heterogeneity, class imbalance, error types, changes over time, external validation, explainability, and generalizability must be considered together. Across the literature, the central challenge is to establish reliable and comparable evaluation frameworks.

Label Heterogeneity and Ground Truth Problem

One major obstacle to direct comparison is that identical class names can refer to different labeling protocols. Chen et al. (2023) constructed the positive class from blacklist sources. Zhuang et al. (2025) classified journals removed from DOAJ as “questionable” after excluding certain reasons for removal. Predatory publishing lists also differ in their criteria and updating procedures (Kakamad et al., 2024, 2026). Moreover, blacklist membership and indexing are not always mutually exclusive (Richtig et al., 2023). These differences affect the meaning of the target variable. “Blacklisted”, “removed from an index”, and “judged questionable by an expert” do not represent equivalent conceptual classes. Zhuang et al. (2025) also used an expert assessment sample separate from their main dataset. Accuracy, F1, and similar metrics from different studies should therefore not be interpreted as results obtained for the same target variable.

The lists examined by Kakamad et al. (2024, 2026) also vary in their inclusion criteria, updating practices, and removal procedures. Ground truth should therefore be described through more than the source name. The labeling protocol, assessment date, supporting evidence, and intended meaning of the label should also be reported. Algorithmic performance cannot be interpreted independently of ground truth quality. Strong performance may show that a model has successfully learned the assigned labels. It does not establish that those labels fully represent predatory publishing behavior in practice.

Class Imbalance and Performance Metrics

Class distributions differ across studies. Chen et al. (2023) used 833 predatory and 1,213 legitimate journals. Zhuang et al. (2025) used a more imbalanced dataset containing 2,536 questionable and 12,869 legitimate journals. Such differences affect

how performance metrics should be interpreted. Overall accuracy can mask errors in the minority class when performance on the majority class is strong. Precision, recall, F1, and PR AUC should therefore be considered together. Zhuang et al. (2025) accounted for class imbalance by comparing models using PR AUC.

The importance of each metric depends on the intended use. Recall becomes particularly important when the aim is to identify as many risky journals as possible. Precision becomes more important when the aim is to limit the proportion of legitimate journals among those flagged as risky. Model performance should therefore be interpreted alongside class distribution and error types, rather than through a single value.

Consequences of False Positives and False Negatives

False positives and false negatives have different consequences. A false positive can cause a journal considered reliable to be unnecessarily flagged as suspicious. A false negative can make a questionable journal appear to pose little risk. Evaluation should therefore consider error types alongside overall accuracy. In their assessment of ChatGPT, Al-Moghrabi et al. (2024) reported a false positive rate of 29% and a false negative rate of 7.5%. The system correctly classified 92.5% of journals labeled predatory and 71% of those labeled legitimate. These findings show that the same system can have different error profiles across classes. However, the meaning of these rates depends on the ground truth used. When reference labels are uncertain or disputed, observed false positive and false negative rates should not be treated as direct estimates of errors in practice.

An external evaluation of AJPC illustrates another aspect of this problem. Teixeira da Silva and Kendall (2023) assessed the top 2,500 SJR entries on 25 March 2023. The system labeled 65.64% as “suspected predatory” and 33.28% as “normal”, while 1.08% could

not be classified. This distribution does not itself represent a misclassification rate. It does, however, show the need to independently test results produced outside the training setting. False positives and false negatives should therefore be interpreted in relation to the reference labels and intended use of the system, alongside their numerical values.

Temporal Drift and Input Variability

Time is an important factor in automatic detection systems that use web data. Journal websites, editorial boards, indexing status, and publishing policies can change. Blacklists and whitelists may also be updated. A system should therefore not be assumed to behave identically at different times. Teixeira da Silva and Kendall (2023) used AJPC to assess the top 2,500 journals in the SJR ranking on 25 March and 30 March 2023. In the first assessment, 65.64% were classified as “suspected predatory”. Five days later, this proportion had risen to 89.20%. The proportion classified as “normal” fell from 33.28% to 10.16% (Teixeira da Silva & Kendall, 2023). These differences show that system outputs can change over time.

Input variability presents a related problem. Teixeira da Silva (2025) showed that AJPC outputs could change when different URLs for the same journal were entered. These included homepages, editorial board pages, and author instructions. Among the ten journals examined, 50% showed inconsistent classifications depending on the URL used (Teixeira da Silva, 2025). This finding highlights the need to specify which page was assessed. Studies using web data should therefore clearly report the assessment date, input URL, and system version. Archiving web content where possible could also support later examination of the results. This recommendation is the review’s methodological interpretation of the temporal and input sensitivity reported in (Teixeira da Silva & Kendall, 2023; Teixeira da Silva, 2025).

Reproducibility and External Validation

Performance on a held out test set does not establish that a model will perform equally well across other journal groups. Reproducibility and external validation therefore require separate consideration when assessing the reliability of automatic journal evaluation systems. AJPC, developed by Chen et al. (2023), was subsequently evaluated independently across different journal groups (Teixeira da Silva & Kendall, 2023; Teixeira da Silva & Scelles, 2024; Teixeira da Silva et al., 2025). Teixeira da Silva and Scelles (2024) reported that the system classified 47 of the 124 journals in the SJR Sports Science category, or 37.9%, as “suspected predatory”. Teixeira da Silva et al. (2025) reported that 44 of the 50 journals in the FT50 group, or 88%, received the same classification.

These studies did not establish an independent gold standard. Their reported proportions therefore cannot be interpreted directly as false positive rates. They nevertheless show the need to examine AJPC outputs for journal groups outside its training data (Teixeira da Silva & Kendall, 2023; Teixeira da Silva & Scelles, 2024; Teixeira da Silva et al., 2025). Taken together, the findings suggest that validation based solely on random training and test splits is insufficient. Future studies could clarify reproducibility and generalizability by explicitly reporting evaluations conducted at different times and on independent journal groups. This recommendation reflects the methodological synthesis developed in this review.

Explainability and Human Oversight

Understanding which features are associated with a prediction is important for interpreting automatic detection outputs. Wu et al. (2024) used SHAP to examine feature contributions in a model assessing early warning risk. However, a strong contribution to a model prediction does not establish that a feature is a reliable

indicator of predatory publishing in practice. An explanation primarily reveals the patterns used by the model. Explainability should therefore be assessed separately from label quality and data source reliability.

This distinction supports the use of automatic systems as decision support tools. Showing the evidence contributing to a result can help users assess it more carefully than a class label or risk score alone would allow. Expert review of the final decision can also prevent automatic outputs from being treated as definitive judgments, particularly in uncertain cases. This proposed use with human oversight reflects the review’s methodological synthesis of the available findings.

Fairness, Geographical Generalizability, and Language Bias

You et al. (2025) examined the relationship between questionable publishing, countries, and regional publishing environments. They found that publishing patterns can vary geographically. Their study also indicated that regional journals and questionable journals should not be treated as the same phenomenon. Perera (2025) discussed fairness concerns surrounding the “predatory” label, particularly when applied to new, less familiar, or poorly resourced journals. Together, these studies support the view that geography or publishing environment alone should not be treated as an indicator of quality (Perera, 2025; You et al., 2025). However, the available sources do not provide comparative experiments that directly measure language or geographical bias in automatic detection models. They therefore do not establish that models systematically learn to treat languages other than English, website design, or economic resources as risk indicators.

An important open challenge identified in this review is to validate automatic systems across languages, regions, and publishing environments. Subgroup evaluations could help

determine whether models learn publishing behavior or indirect characteristics such as geography and language. However, the limited evidence does not support strong generalizations about fairness or language bias.

Reliability and Reproducibility of Large Language Models

Al-Moghrabi et al. (2024) reported an overall accuracy of 0.82 for ChatGPT's assessment of biomedical and dental journals. Correct classification reached 92.5% for journals labeled predatory and 71% for those labeled legitimate. This difference highlights the need to verify LLM outputs before using them as final decisions. The findings are limited to the ChatGPT version, assessment period, disciplines, and sample examined (Al-Moghrabi et al., 2024). Additional prompts and journal website information were also used when needed. Reporting the model version, prompts, supplied context, and assessment date is therefore important for reproducibility. This recommendation is the review's methodological interpretation of the study design.

Explanations generated by LLMs are not equivalent to verified evidence. Hegedűs et al. (2026) compared ChatGPT, Gemini, Copilot, DeepSeek, Qwen, Perplexity, and Claude using hijacked medical journals. The systems could provide general information but could not reliably distinguish genuine journals from hijacked versions (Hegedűs et al., 2026). Although the study did not directly examine predatory journal classification, it provides complementary evidence for checking LLM outputs against independent sources. The roles of LLMs in extracting evidence, structuring information, and generating explanations should therefore be distinguished from their role as final classifiers. This distinction allows their use within decision support processes that can be checked and supervised.

Transition from Static Features to Behavioral Evidence

Static features such as website text, design, and metadata provide accessible information about a journal. They do not, on their own, establish how a publisher operates in practice. Behavioral evidence can therefore serve as a separate source of information that complements static features. Fung et al. (2026) checked claims about indexing, impact factors, ISSNs, and PubMed in publication invitations against independent sources. Thallinger et al. (2026) examined invitation content and found that the invitation topic did not match the recipient's expertise in 148 of 188 emails. These studies show how publisher statements can be compared with verifiable information and observed communication behavior.

Practices after publication can provide additional evidence. Xu et al. (2026) examined retraction practices among 1,511 potentially predatory journals on the updated Beall's List. They identified 645 retraction records across 46 journals by the end of 2022. The study assessed the visibility of retraction policies, document accessibility, and the quality of retraction notices (Xu et al., 2026). These findings suggest that automatic assessment can consider the verifiability of claims and observable publishing practices alongside textual and design features. However, equivalent behavioral evidence may not be available for every journal. This review therefore proposes using behavioral indicators as a complementary evidence layer, rather than as ground truth on their own.

Overall Synthesis

The literature shows that the main challenges in automatic predatory journal detection are interconnected. Differences in label sources and criteria affect the meaning of performance values. Class imbalance can alter error distributions. Assessment dates and web inputs can affect output stability. A high performance score alone

therefore does not establish system reliability. Internal validation is also insufficient on its own. Systems require external validation on independent data, separate assessment of false positives and false negatives, and clear identification of the evidence supporting predictions. Explainability can reveal which features influence a model, but it cannot resolve weaknesses in labels or evidence quality. Limited comparative evidence across languages and geographical publishing environments also calls for caution when judging generalizability.

These findings show that the research problem extends beyond identifying the classifier with the highest accuracy. The central need is to combine evidence from different sources in ways that remain verifiable, explainable, and reproducible as that evidence changes over time. Automatic detection should therefore be treated as a decision support problem that incorporates evidence quality, uncertainty, and human review.

Future Research Directions

Future research needs to address more than improvements in classification performance. Lists used to identify predatory publishing differ in their coverage and criteria (Kakamad et al., 2026). Zhuang et al. (2025) accounted for class imbalance and used a separate expert assessment set. External evaluations of AJPC show that results obtained outside the training setting require further testing (Teixeira da Silva & Kendall, 2023). Different page URLs for the same journal can also produce different results, highlighting the importance of clearly defining inputs in systems using web data (Teixeira da Silva, 2025). Drawing on these findings, this review considers future research in terms of data standardization, evidence from multiple sources, external validation, explainability, and decision support with human oversight.

Standardized, Timestamped, and Updatable Datasets

The literature uses different lists and data sources to establish ground truth. Kakamad et al. (2024, 2026) showed that lists of predatory journals and publishers differ in coverage and criteria. Richtig et al. (2023) found that some journals on Beall's List also appeared in Scopus, DOAJ, PubMed, PubMed Central, and Web of Science. Zhuang et al. (2025) supplemented listing and delisting information in their main dataset with a separate expert assessment set. These examples challenge the assumption that labels from different sources have equivalent meanings. Future datasets may therefore need to record more than final labels such as “predatory”, “questionable”, or “legitimate”. Recording the label source, assessment date, criteria, and any label changes would provide useful context. This recommendation reflects the review's methodological synthesis of the observed label heterogeneity.

Standardized reporting of journal identifiers, ISSNs, collection dates, assessed URLs, and indexing status could also improve comparability. Recording URLs is particularly important because different web pages can produce different results (Teixeira da Silva, 2025). These records would clarify the temporal and input conditions represented by a dataset. They would also support temporal validation. Teixeira da Silva and Kendall (2023) showed that repeated AJPC assessments produced different classification distributions five days apart. Teixeira da Silva (2025) reported that different page URLs for the same journal could lead to different classifications. Data records should therefore include collection dates, input URLs, and system versions where available. This review also recommends testing models on data collected during a later period, alongside random training and test splits.

Multi-Source Evidence Fusion and Integration of Behavioral Evidence

Existing studies show that journal assessment can use several types of evidence. Zhuang et al. (2025) combined website content, design features, and publication information. Siess (2025) examined PDF features of published articles, while Downes (2024) investigated editorial board patterns. Fung et al. (2026) and Thallinger et al. (2026) examined publication invitations. Xu et al. (2026) assessed retraction practices as a form of quality control after publication. Although these studies use different data, they show how web content, visual features, editorial information, invitations, and practices after publication can provide distinct evidence layers (Downes, 2024; Fung et al., 2026; Siess, 2025; Thallinger et al., 2026; Xu et al., 2026; Zhuang et al., 2025). Publisher statements should be distinguished from information that can be independently verified. Fung et al. (2026) demonstrated this distinction by checking indexing and impact factor claims against external sources.

Future systems should therefore consider the sources and verifiability of evidence when combining it. Independent datasets should be used to test whether behavioral indicators add information beyond static features. Studies should also assess whether combining these evidence types reduces false positives and false negatives. These recommendations arise from this review's methodological synthesis. They do not describe a common evidence fusion model already tested by the reviewed studies.

Explainable and Human-Supervised Decision Support Systems

Wu et al. (2024) used SHAP to examine feature contributions to model predictions. This approach can reveal which features influence a model. A strong contribution to a prediction, however, does not establish that a feature is reliable evidence in practice. Tang (2025) proposed supplementing quantitative machine assessment

with qualitative community assessment. This provides a direct conceptual basis for using automatic systems to support human judgment.

Future systems should distinguish model explanations from the verifiability of the underlying evidence. Outputs could include the source, currency, and verification status of the evidence used. Intermediate outcomes such as “review required” or “insufficient evidence” could also be considered for uncertain cases. These proposed output formats and the recommendation for human oversight reflect the review’s methodological synthesis.

Reliable and Reproducible Use of Large Language Models

Available findings call for caution when using large language models as sole decision makers in journal assessment. Al-Moghrabi et al. (2024) reported different classification performance for journals labeled predatory and legitimate. Hegedűs et al. (2026) compared several chatbots using hijacked medical journals. The systems could not reliably distinguish genuine journals from hijacked versions. Although this second study did not directly examine predatory journal classification, it provides complementary evidence against treating LLM outputs as conclusive evidence without independent verification.

Future studies should examine the reproducibility of LLM assessments. They should clearly report the model and version, prompts, supplied context, and assessment date. Evaluating the same journals across models, prompts, and time points could also help assess output stability. These recommendations form part of this review’s methodological synthesis. They do not represent a common protocol directly tested in the existing studies.

Within this framework, LLMs could primarily extract evidence, structure information, and provide explanations. Sources

should accompany extracted information, and critical claims should be checked against independent data. This would position LLMs as components of decision support with human oversight.

External Validation, Generalizability, and Fair Assessment

Future automatic detection systems require evaluation beyond the datasets used to develop them. Independent validation should cover different data sources, disciplines, languages, and publishing environments. This would help establish whether learned patterns are specific to a particular dataset. Existing literature indicates that publishing practices can vary across regions and contexts (Perera, 2025; You et al., 2025). However, these studies do not directly establish systematic bias across all linguistic or geographical groups in current detection systems. Language, geography, and website design should therefore not automatically be treated as indicators of poor quality.

Where samples are sufficient, future evaluations should report performance separately across languages, disciplines, regions, and publisher groups. They should also examine whether false positive and false negative rates differ across these groups. The purpose is to assess how well model performance holds across contexts. This review proposes this evaluation framework based on the contextual findings in (Perera, 2025; You et al., 2025).

Conceptual Architecture for Next-Generation Automated Journal Evaluation

Taken together, the reviewed studies support developing automatic journal assessment within a decision support framework that draws on multiple sources. Relevant inputs could include web content, journal metadata, visual features, editorial information, bibliometric data, and behavioral evidence. LLMs could assist with extracting and explaining unstructured information. Combining

evidence would be only one part of this system. Each item's source, date, and verification status should also be recorded. Outputs could present a risk assessment alongside uncertainty and supporting evidence. Cases with insufficient or conflicting evidence could be referred for human review.

This architecture has not been implemented and validated in full by a single study in the reviewed literature. Chen et al. (2023), Zhuang et al. (2025), Wu et al. (2024), Al-Moghrabi et al. (2024), Fung et al. (2026), Thallinger et al. (2026), Downes (2024), Siess (2025), and Xu et al. (2026) provide evidence on different data and assessment components. The integrated framework proposed here is a conceptual synthesis of their findings.

Future research should therefore pursue more than higher accuracy. Priorities include traceable label sources, assessments that can be updated over time, independent verification of evidence, testing across contexts, and human review of uncertain cases. Together, these priorities could advance automatic predatory journal detection as a decision support process grounded in verifiable evidence.

Conclusion

Automatic predatory journal detection has expanded from conventional machine learning and natural language processing to web and visual analysis, explainable artificial intelligence, large language models, and behavioral evidence (Al-Moghrabi et al., 2024; Chen et al., 2023; Wu et al., 2024; Zhuang et al., 2025). These developments offer opportunities to assess large numbers of journals at scale. However, the reviewed studies show that system reliability depends on more than an algorithm's classification performance.

For RQ1, the literature indicates that reliability is directly affected by dataset construction, the sources and meanings of ground

truth labels, feature representations, and validation strategies. Blacklists, whitelists, indexing databases, delisting records, expert assessments, and behavioral evidence produce different class definitions (Kakamad et al., 2024, 2026; Richtig et al., 2023; Zhuang et al., 2025). Labels such as “predatory”, “questionable”, and “legitimate” therefore cannot be assumed to represent the same target variable across studies. Class imbalance, the different consequences of false positives and false negatives, changing web content, URL sensitivity, and limited external validation also affect performance interpretation (Teixeira da Silva & Kendall, 2023; Teixeira da Silva & Scelles, 2024; Teixeira da Silva et al., 2025; Teixeira da Silva, 2025; Zhuang et al., 2025). Accuracy, F1, and AUC should therefore be interpreted alongside data and label construction, ground truth quality, collection conditions, temporal validity, and independent validation.

For RQ2, the empirical evidence is more limited. The study directly examining ChatGPT for predatory journal assessment showed that LLMs could generate classifications and explanations. However, limitations remained, including false positives, explanations that had not been independently verified, and results tied to a particular sample and assessment conditions (Al-Moghrabi et al., 2024). A related study comparing AI chatbots using hijacked medical journals provided complementary evidence of difficulties in distinguishing genuine journals from hijacked versions (Hegedűs et al., 2026). These findings should not be treated as directly equivalent because hijacked journal detection and predatory journal detection address different target problems. Overall, the reliability of LLMs as independent, final journal classifiers has not yet been sufficiently established. Their use requires further investigation through broader datasets, independent validation, repeated assessments over time, and reproducible evaluation protocols.

The literature also suggests that reducing journal assessment to a binary classification can discard important information. Website text and structure, bibliometric information, editorial board patterns, publication invitations, indexing claims, and practices after publication can provide distinct and complementary evidence (Burgiss et al., 2024; Downes, 2024; Fung et al., 2026; Thallinger et al., 2026; Xu et al., 2026). Explainable artificial intelligence can help reveal features associated with model predictions (Wu et al., 2024). However, a feature's importance to a model does not establish its causal relevance or reliability as an indicator of predatory publishing. Similarly, explanations generated by LLMs are not equivalent to independently verified evidence.

One limitation of this review is that the numbers of screened, duplicate, excluded, and fully assessed records were not recorded quantitatively during study selection. These counts were not reconstructed retrospectively, and no selection flow diagram was presented. This limits assessment of the review's comprehensiveness and the reproducibility of its selection process.

More reliable automatic detection requires clear tracking of evidence sources, currency, and verifiability. Datasets and labeling protocols should be reported transparently, and internal test results should be supported by independent and temporal validation. The proposal to complement quantitative machine assessment with qualitative expert and community assessment (Tang, 2025) also supports this approach. The current evidence therefore favors using automatic detection as a process that systematically gathers evidence and supports human judgment through decision support tools whose outputs can be checked and reviewed.

References

- Abalkina, A. (2021). Detecting a network of hijacked journals by its archive. *Scientometrics*, 126(8), 7123–7148. <https://doi.org/10.1007/s11192-021-04056-0>
- Al-Moghrabi, D., Abu Arqub, S., Maroulakos, M. P., Pandis, N., & Fleming, P. S. (2024). Can ChatGPT identify predatory biomedical and dental journals? A cross-sectional content analysis. *Journal of Dentistry*, 142, Article 104840. <https://doi.org/10.1016/j.jdent.2024.104840>
- Andoohgin Shahri, M., Jazi, M. D., Borchardt, G., & Dadkhah, M. (2018). Detecting hijacked journals by using classification algorithms. *Science and Engineering Ethics*, 24(2), 655–668. <https://doi.org/10.1007/s11948-017-9914-2>
- Bin Ateeq, W. M., & Al-Khalifa, H. S. (2023). Intelligent framework for detecting predatory publishing venues. *IEEE Access*, 11, 20582–20618. <https://doi.org/10.1109/ACCESS.2023.3250256>
- Burgiss, L., Tatum, B., Henshaw, C., Boswell, M., & Michaels, A. (2024). Predatory publication of AI-generated research papers. In *Proceedings of the First International Conference on Natural Language Processing and Artificial Intelligence for Cyber Security* (pp. 1–9). International Conference on Natural Language Processing and Artificial Intelligence for Cyber Security. <https://aclanthology.org/2024.nlpaiacs-1.1/>
- Chen, L.X., Su, S.W., Liao, C.H., Wong, K.S., & Yuan, S.M. (2023). An open automation system for predatory journal detection. *Scientific Reports*, 13(1), Article 2976. <https://doi.org/10.1038/s41598-023-30176-z>
- do Carmo, F. A., Rezende, S. O., Paxiúba, C. M. C., & Lobato, F. M. F. (2026). Predatory publishing in the academic landscape: researchers' awareness and vulnerabilities. *Scientometrics*, 131(5), 3029–3054. <https://doi.org/10.1007/s11192-026-05614-0>
- Downes, M. (2024). The stock characters in the editorial boards of journals run by predatory publishers. *Learned Publishing*, 37(3), Article e1608. <https://doi.org/10.1002/leap.1608>
- Frandsen, T. F., Lamptey, R. B., & Borteye, E. M. (2024). Promotion standards to discourage publishing in questionable journals: a follow-up study. *The Journal of Academic Librarianship*, 50(5), Article 102895. <https://doi.org/10.1016/j.acalib.2024.102895>

- Fung, C., Razi, B., Basto, M., Bariol, S., Hossack, T., Baskaranathan, S., Ende, D., & Woo, H. H. (2026). Trust, but Verify: An Exploratory Audit of Predatory Journal and Publisher Solicitations. *ANZ Journal of Surgery*, 96(4), 786–791. <https://doi.org/10.1111/ans.70568>
- Hegedűs, M., Dadkhah, M., & Dávid, L. D. (2026). Benchmarking AI chatbots: assessing their accuracy in identifying hijacked medical journals. *Diagnosis*, 13(1), 82–91. <https://doi.org/10.1515/dx-2025-0043>
- Kakamad, F. H., Abdalla, B. A., Abdullah, H. O., Omar, S. S., Mohammed, S. H., Ahmed, S. M., Mohammed, K. K., Hassan, H. A., Baba, H. O., Ahmed, J. O., Mustafa, M. Q., Omar, D. A., Salih, R. Q., Rahim, H. M., Hussein, D. A., Hassan, M. N., Mikael, T. M., Hassan, H. A., & Najar, K. A. (2024). Lists of predatory journals and publishers: a review for future refinement. *European Science Editing*, 50, Article e118119. <https://doi.org/10.3897/ese.2024.e118119>
- Kakamad, F. H., Mustafa, A. M., Abdalla, B. A., Mohammed, S. H., Ahmed, S. M., Abdullah, H. O., Ahmed, J. O., Abdullah, F., Abdalla, S. S., Mikael, T. M., Hassan, H. A., Najar, K. A., & Omar, D. A. (2026). Predatory publishing lists: a systematic review. *Annals of Medicine & Surgery*, 88(2), 1842–1848. <https://doi.org/10.1097/MS9.0000000000004733>
- Kashyap, K., Islam, A. A., & Gielen, J. (2024). Why do healthcare researchers in South Asia publish in predatory journals? A scoping review. *Developing World Bioethics*, 24(2), 54–65. <https://doi.org/10.1111/dewb.12388>
- Khabour, O. F., Alzoubi, K. H., & Aldarabseh, W. M. (2024). Awareness of Jordanian Researchers About Predatory Journals: A Need for Training. *Science and Engineering Ethics*, 30(6), Article 58. <https://doi.org/10.1007/s11948-024-00519-8>
- Laine, C., Babski, D., Bachelet, V. C., Bärnighausen, T. W., Baethge, C., Bibbins-Domingo, K., Frizelle, F., Gollogly, L., Kleinert, S., Loder, E., Monteiro, J., Rubin, E. J., Sahni, P., Wee, C. C., Yoo, J.H., & Zakhama, L. (2025). Predatory journals: what can we do to protect their prey?. *BMJ*, 388, Article q2850. <https://doi.org/10.1136/bmj.q2850>
- Martinino, A., Smeenk, F. W.J.M., Basile, V., Soto, A., & Pouwels, S. (2025). Understanding the Influence of Predatory Journals Articles Included within Systematic Reviews: A Scoping Review. *The Serials Librarian*, 86(3-4), 213–222. <https://doi.org/10.1080/0361526X.2025.2523389>

- Öztürk, O., & Taşkın, Z. (2024). How metric-based performance evaluation systems fuel the growth of questionable publications?. *Scientometrics*, 129(5), 2729–2748. <https://doi.org/10.1007/s11192-024-04991-8>
- Perera, H. S. (2025). Questioning the Predator of the Predatory Journals: How Fair Are Global Publishing Standards?. *Learned Publishing*, 38(2), Article e1662. <https://doi.org/10.1002/leap.1662>
- Richtig, G., Berger, M., Koeller, M., Richtig, M., Richtig, E., Scheffel, J., Maurer, M., & Siebenhaar, F. (2023). Predatory journals: Perception, impact and use of Beall’s list by the scientific community—A bibliometric big data study. *PLOS ONE*, 18(7), Article e0287547. <https://doi.org/10.1371/journal.pone.0287547>
- Siess, A. (2025). Evaluating the visual design of science publications—a quantitative approach comparing legitimate and predatory journal papers. *Scientometrics*, 130(9), 5043–5066. <https://doi.org/10.1007/s11192-025-05411-1>
- Tang, B. L. (2025). Are there accurate and legitimate ways to machine-quantify predatoriness, or an urgent need for an automated online tool?. *Accountability in Research*, 32(2), 182–187. <https://doi.org/10.1080/08989621.2023.2253425>
- Teixeira da Silva, J. A., & Kendall, G. (2023). (Mis-)Classification of 17,721 Journals by an Artificial Intelligence Predatory Journal Detector. *Publishing Research Quarterly*, 39(3), 263–279. <https://doi.org/10.1007/s12109-023-09956-y>
- Teixeira da Silva, J. A., & Scelles, N. (2024). An artificial intelligence tool misclassifies sport science journals as predatory. *Journal of Science and Medicine in Sport*, 27(4), 266–269. <https://doi.org/10.1016/j.jsams.2023.12.006>
- Teixeira da Silva, J. A., Tsigaris, P., & Moussa, S. (2025). Are most FT50 journals truly “suspected predatory”, or has an AI-driven “predatory” journal detector erred in its classification?. *Journal of Electronic Resources Librarianship*, 37(2), 135–146. <https://doi.org/10.1080/1941126X.2025.2497727>
- Teixeira da Silva, J. A. (2025). Variation in journal website characterization by an AI-driven “predatory” journal detector. *Expert Systems with Applications*, 283, Article 127933. <https://doi.org/10.1016/j.eswa.2025.127933>

- Thallinger, C., Aschbacher, V., & Zielinski, C. C. (2026). Invitations to Publish in a Journal by Email: A Prospective Analysis. *Learned Publishing*, 39(3), Article e2076. <https://doi.org/10.1002/leap.2076>
- Tomlinson, O. W. (2024). Predatory publishing in medical education: a rapid scoping review. *BMC Medical Education*, 24(1), Article 33. <https://doi.org/10.1186/s12909-024-05024-x>
- Wu, J., Liu, T., Mu, K., & Zhou, L. (2024). Identification and causal analysis of predatory open access journals based on interpretable machine learning. *Scientometrics*, 129(4), 2131–2158. <https://doi.org/10.1007/s11192-024-04969-6>
- Xu, S. B., Liu, T., Nejadghanbar, H., & Hu, G. (2026). Retraction handling by potential predatory journals. *Accountability in Research*, 33(1), 1–27. <https://doi.org/10.1080/08989621.2024.2434245>
- You, T., Park, J., Lee, J. Y., & Yun, J. (2025). Regional profile of questionable publishing. *Journal of Informetrics*, 19(3), Article 101670. <https://doi.org/10.1016/j.joi.2025.101670>
- Zhuang, H., Liang, L., & Acuna, D. E. (2025). Estimating the predictability of questionable open-access journals. *Science Advances*, 11(35), Article eadt2792. <https://doi.org/10.1126/sciadv.adt2792>

CHAPTER 10

ZERO TRUST CLOUD SECURITY AND ARTIFICIAL INTELLIGENCE DRIVEN IDENTITY PROTECTION

ÖMER SEVİNÇ¹

Abstract Cloud computing, hybrid work, software as a service, and machine identities have weakened the value of a fixed network perimeter. Attackers increasingly target credentials, session tokens, application programming interfaces, and cloud control planes rather than attempting to penetrate a traditional office network. This chapter examines zero trust as an architectural approach for protecting cloud-native and hybrid systems and analyzes how artificial intelligence can support identity threat detection, access-risk assessment, policy enforcement, and incident response. It distinguishes zero trust from a product category and explains its dependence on accurate asset inventories, strong authentication, workload identity, least privilege, microsegmentation, telemetry, and governance. Particular attention is given to token theft, non-human identities, compromised cloud administration, model limitations, and privacy. A phased implementation model is proposed for

¹ Dr., Ondokuz Mayıs University, Vezirköprü Vocational School, Department of Computer Programming.

organizations seeking measurable security improvement without disrupting legitimate users or critical services.

Keywords zero trust, cloud security, identity threat detection, artificial intelligence, cloud native security, access control

Introduction

Enterprise computing no longer resides within a single controlled network. Employees, contractors, applications, devices, and automated workloads access resources distributed across private data centers, several cloud providers, and software-as-a-service platforms. A user may authenticate through one identity provider, receive a token, and reach data through multiple application programming interfaces without traversing a traditional security gateway. Network location therefore provides weak evidence of trust. A valid session may belong to a legitimate employee, an attacker using stolen credentials, or malware operating through a compromised browser.

Zero trust addresses this environment by removing implicit trust based on location, ownership, or a previous authentication event. Access decisions are made for a specific subject, resource, action, and context, and are reassessed as risk changes. NIST defines zero trust as a shift from static network perimeters toward protection of users, assets, and resources (Rose et al., 2020). Artificial intelligence can strengthen this approach by identifying unusual relationships and behavior at cloud scale, but it cannot replace identity governance, secure configuration, or accountable policy.

Cloud Identity as the Primary Attack Surface

Cloud incidents frequently begin with identity rather than exploitation of a network service. Password phishing, consent phishing, adversary-in-the-middle techniques, help-desk manipulation, leaked access keys, and stolen browser tokens can

give an attacker the same interfaces available to an authorized user. Multifactor authentication reduces risk, but some methods remain vulnerable to push fatigue, social engineering, or token replay. Phishing-resistant authentication based on cryptographic credentials provides stronger protection for administrators and other high-impact roles.

Session tokens and assertions require specific controls because they may remain useful after authentication has completed. Short lifetimes, secure storage, audience restriction, sender constraints where supported, continuous access evaluation, and rapid revocation reduce exposure. Identity providers, federation services, and cloud control planes should receive the highest level of monitoring because their compromise can create valid credentials, alter logging, or grant persistent access across many systems. Recent CISA and NIST guidance treats the protection of cloud tokens and assertions as a distinct implementation problem rather than an ordinary password issue (CISA & NIST, 2026).

Zero Trust Architecture for Hybrid and Cloud Native Systems

A zero trust architecture separates policy decisions from enforcement. A policy engine evaluates identity, device posture, resource sensitivity, requested action, threat information, and environmental context. Enforcement points then allow, restrict, or deny the transaction. This model can operate at an identity proxy, application gateway, service mesh, database, endpoint, or software-defined network. Effective placement is close to the protected resource so that internal traffic and service-to-service calls do not bypass the policy.

Cloud-native systems introduce short-lived containers, serverless functions, managed services, and microservices that communicate through application programming interfaces.

Workloads need identities that are independent of internet addresses and do not rely on embedded secrets. Mutual authentication, workload certificates, signed software artifacts, and narrowly scoped authorization can reduce lateral movement. NIST SP 800-207A describes how application-level policies and service infrastructure can apply zero trust principles across hybrid and multi-cloud environments (Chandramouli & Butcher, 2023).

Artificial Intelligence for Identity Threat Detection

AI-supported identity analytics can establish patterns for sign-in time, device, network, privilege use, resource access, transaction volume, and relationships among users and workloads. A model may identify impossible travel, unusual administrative actions, a service account acting interactively, or a user downloading data that is inconsistent with the person's role. Graph analysis is particularly useful when an attacker performs several individually plausible actions that form a suspicious path toward a sensitive resource.

Risk scoring should support proportionate controls. A low-confidence anomaly might increase logging; moderate risk may trigger step-up authentication; strong evidence of active compromise may suspend a token or isolate a workload. Organizations should preserve the reasons, evidence, and policy used for consequential actions. A model that produces an unexplained score should not have unrestricted authority to disable an executive, clinician, production service, or emergency operator. Human review and tested emergency criteria remain necessary.

Non Human Identities and Machine Access

Modern environments contain more machine identities than human users. Service accounts, application registrations, automation agents, build pipelines, managed identities, certificates, and API keys may hold broad privileges and operate continuously. These

identities are often poorly inventoried because development teams create them through infrastructure code or cloud consoles. Long-lived secrets can be copied into repositories, configuration files, tickets, or logs and remain usable after the original project changes.

A secure program assigns an owner and purpose to every non-human identity, eliminates unused credentials, prefers short-lived tokens, and restricts each identity to defined resources and actions. Workload identity federation can reduce static secrets by exchanging trusted platform evidence for temporary credentials. AI can assist with inventory, privilege analysis, and detection of abnormal use, but automated recommendations must recognize scheduled jobs, failover activity, and deployment patterns to avoid interrupting production.

Data Protection Microsegmentation and Policy Enforcement

Zero trust protects data and services through granular policy rather than assuming that internal communication is safe. Data classification allows policy to reflect sensitivity, legal duties, and business impact. Encryption protects content in transit and at rest, while authorization determines who or what may perform a particular operation. Policy should consider the requested action because reading a public record, exporting a database, changing a security rule, and deleting a backup create different risks.

Microsegmentation limits the paths available after compromise. In cloud-native systems, segmentation may be enforced through security groups, service-mesh policy, application gateways, database permissions, and workload identity. Excessive segmentation can create operational complexity, so organizations should begin with high-value resources and common attack paths. Policy-as-code can improve consistency and review, but policy

changes require testing, version control, approval, and rollback because an incorrect rule can cause a widespread outage.

Implementation and Maturity Model

Implementation should begin with visibility. The organization needs an inventory of users, privileged roles, workloads, devices, applications, data, access paths, identity providers, and enforcement points. The next stage strengthens authentication, removes standing privilege, and protects administrative interfaces. Subsequent stages introduce context-aware decisions, workload identity, microsegmentation, automated response, and continuous policy evaluation. CISA’s Zero Trust Maturity Model organizes progress across identity, devices, networks, applications and workloads, and data, supported by visibility, automation, and governance (CISA, 2023).

A phased roadmap should define measurable outcomes rather than claiming that the organization has purchased zero trust. Useful measures include phishing-resistant authentication coverage, privileged-role activation time, dormant identity removal, number and age of static secrets, token revocation time, policy coverage for sensitive applications, and the proportion of service traffic using authenticated workload identities. Incident exercises should test stolen tokens, compromised administrators, malicious applications, and cloud-provider outages.

Domain	Foundational control	Advanced capability
Identity	Phishing-resistant authentication and role governance	Continuous risk evaluation and just-in-time privilege
Workloads	Unique identities and secret reduction	Short-lived credentials and service-level policy
Networks	Segmentation around critical resources	Dynamic application-aware microsegmentation
Data	Classification and access logging	Transaction-level authorization and loss detection

Privacy Governance and Operational Limitations

Continuous identity analytics can collect detailed information about work patterns, location, devices, communication, and resource use. Monitoring should have a defined security purpose, minimum necessary data, access restrictions, retention limits, and independent review. Models should be evaluated for false positives affecting remote workers, travelers, shared devices, accessibility tools, and unusual job roles. Security monitoring must not silently become a general employee-performance system.

Zero trust also has limits. It does not correct insecure software, prevent every supply-chain compromise, or guarantee that an authorized user will act safely. Policy engines and identity providers become critical dependencies that require resilience, backups, and protected administrative paths. Organizations need break-glass procedures that are strongly controlled and reviewed. Success depends on accurate information, disciplined governance, and usable controls that legitimate users can follow.

Conclusion

Zero trust is a practical response to cloud systems in which users, workloads, and data move across organizational and provider boundaries. Its central contribution is the replacement of broad, persistent trust with explicit and contextual authorization. AI improves visibility and can reveal identity abuse that fixed rules miss, but automated decisions must remain explainable, proportionate, and reversible. Organizations should focus first on high-impact identities and resources, then expand enforcement as inventories, telemetry, and operational maturity improve.

References

- Chandramouli, R., & Butcher, Z. (2023). A zero trust architecture model for access control in cloud-native applications in multi-cloud environments (NIST SP 800-207A). National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.SP.800-207A>
- Cybersecurity and Infrastructure Security Agency. (2023). Zero Trust Maturity Model Version 2.0. <https://www.cisa.gov/resources-tools/resources/zero-trust-maturity-model>
- Cybersecurity and Infrastructure Security Agency. (2024). Implementing secure practices for cloud services (Binding Operational Directive 25-01). <https://www.cisa.gov/news-events/news/cisa-directs-federal-agencies-secure-cloud-environments>
- Cybersecurity and Infrastructure Security Agency, & National Institute of Standards and Technology. (2026). Protecting tokens and assertions from forgery, theft, and misuse (NIST IR 8587). <https://www.cisa.gov/news-events/news/cisa-and-nist-release-guidelines-protect-federal-cloud-identity-systems-token-theft-forgery-and>
- National Institute of Standards and Technology. (2025). Implementing a zero trust architecture (NIST SP 1800-35). <https://doi.org/10.6028/NIST.SP.1800-35>
- Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). Zero trust architecture (NIST SP 800-207). National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.SP.800-207>

CHAPTER 11

GRAPH-BASED CYBER-ATTACK DETECTION: REPRESENTATION STRATEGIES, TOPOLOGICAL ANALYSIS, AND COMPUTATIONAL EFFICIENCY

ALİ MELİH KANCA¹
İLKER TÜRKER²

Abstract

Graph-based approaches provide a flexible framework for cyber-attack detection by representing structural relationships among hosts, flows, events, system entities, and temporally ordered observations. This chapter presents an integrated methodological perspective on graph-based cyber-attack detection, focusing on the interplay among graph construction, representation strategy, topological characterization, interpretability, and computational efficiency. Two principal representation strategies are considered: handcrafted representations based on explicitly defined graph-theoretic descriptors and automated representation learning based on

¹ PhD, Karabük University Department of Computer Engineering, Orcid: 0000-0003-3271-7463

² Prof. Karabük University Department of Computer Engineering, Orcid: 0000-0001-7577-4658

graph embeddings and graph neural networks. Natural Visibility Graphs (NVGs) are examined as a focused example of how sequential network-traffic observations can be transformed into graph structures and characterized through interpretable topological properties. Evidence from published and publicly available studies is synthesized to demonstrate how graph topology can reveal attack-specific structural patterns and how the relevance of graph-derived descriptors can be systematically assessed. Feature-importance analysis, representation reduction, validation, and performance–efficiency trade-offs are subsequently discussed as complementary components of graph-based model design. Overall, the chapter provides an integrated framework that connects graph construction, structural representation, topological interpretation, and computational considerations for the development of predictive, interpretable, and computationally efficient cyber-attack detection systems.

Introduction

The increasing scale, complexity, and heterogeneity of modern networked systems have made the timely detection of malicious activities a fundamental requirement for cybersecurity. Network environments continuously generate large volumes of traffic containing interactions produced by legitimate users, applications, services, and potentially malicious actors. Within these environments, cyber-attacks may manifest through changes in communication behavior, packet and flow characteristics, temporal activity patterns, or interactions among network entities. Consequently, intrusion detection has evolved from predominantly rule- and signature-based mechanisms toward data-driven approaches capable of identifying complex patterns associated with both known and previously unseen malicious behavior.

Machine-learning (ML) and deep-learning (DL) techniques have become important components of this transition. Conventional ML algorithms can learn decision boundaries from engineered traffic features, whereas deep-learning architectures can model more complex nonlinear relationships directly from high-dimensional input representations. The increasing use of convolutional neural networks (CNNs), recurrent architectures, autoencoders, transformers, and related learning models has demonstrated that the predictive capability of an intrusion-detection system depends not only on the selected classification algorithm but also on the way in which the underlying security data are represented. A highly expressive classifier may still be constrained when the input representation does not adequately capture the behavioral or structural characteristics associated with malicious activity.

Traditional flow-based intrusion-detection approaches generally represent each network flow as a vector composed of statistical or protocol-related attributes. These attributes may include packet counts, byte counts, packet-length statistics, inter-arrival times, flow duration, forward and backward traffic characteristics, and other numerical descriptors. Such representations are computationally convenient and have been successfully employed in numerous machine-learning and deep-learning studies. However, a conventional feature vector primarily describes the attributes of an individual observation. Relationships among observations, communicating entities, temporally adjacent events, or interacting processes are not necessarily represented explicitly.

This limitation is particularly relevant because many cyber-attacks are inherently relational or structural. Distributed denial-of-service activity, for example, may involve coordinated communication patterns among multiple sources and a common target. Botnet behavior can be reflected through repeated or coordinated interactions among compromised devices. Advanced

threats may consist of sequences of causally related system events, while malware behavior may be characterized through relationships among functions, system calls, or application programming interface interactions. Similarly, temporal changes in individual network-traffic features may contain structural information that is difficult to express through isolated scalar values. These characteristics motivate representations capable of preserving relationships and structural organization in addition to conventional statistical information.

Graph-based modeling provides a natural framework for this purpose. A graph represents a system through a set of nodes and edges, where nodes correspond to entities or observations and edges describe relationships among them. Depending on the cybersecurity problem, nodes may represent hosts, Internet Protocol addresses, users, processes, functions, packets, flows, events, or temporally ordered observations. Edges can represent communication, dependency, invocation, causality, similarity, temporal association, or mathematically defined structural relationships. Graph construction therefore provides a flexible mechanism for transforming cybersecurity data into representations in which relational information becomes explicitly available for analysis (Caville et al., 2022; Daya et al., 2019; Islam et al., 2020; Protogerou et al., 2021; Zola et al., 2022).

An important characteristic of graph-based cybersecurity analysis is that there is no single universally applicable graph representation. Different graph construction strategies emphasize different properties of the underlying data. Communication graphs can model interactions among hosts or network endpoints, while flow graphs can characterize relationships associated with network flows. Function-call graphs represent behavioral relationships among software functions and can therefore support malware-oriented analysis. Provenance graphs provide a mechanism for

modeling causal relationships among system entities and events and are particularly relevant to multi-stage or advanced threats. Dynamic and temporal graphs extend these representations by explicitly incorporating changes in nodes, edges, or attributes over time.

Visibility graphs provide another perspective by transforming numerical sequences into graph structures (Iacovacci & Lacasa, 2016; Lacasa et al., 2008). Instead of defining nodes as communicating hosts or system entities, visibility-graph approaches treat individual observations within an ordered sequence as graph nodes and establish edges according to geometrically defined visibility relationships. Natural Visibility Graphs (NVGs), in particular, provide a mechanism for preserving aspects of the ordering and structural variation of a numerical sequence while enabling that sequence to be examined using graph-theoretic measures. This characteristic makes NVGs relevant to network-traffic analysis, where numerical flow features can be interpreted as sequential observations and subsequently transformed into graph structures (Kanca & Türker, 2026).

Once cybersecurity data have been represented as graphs, the resulting structures can be analyzed through different representation strategies. One approach is to calculate explicitly defined, or handcrafted, graph descriptors. Measures such as degree, graph density, clustering coefficient, shortest-path characteristics, betweenness centrality, closeness centrality, eigenvector centrality, modularity, graph diameter, and community-related measures summarize different structural properties of a graph. These descriptors can subsequently be supplied to conventional ML or DL classifiers and offer an important advantage in terms of structural interpretability because each feature corresponds to a defined graph-theoretic property.

A second strategy is automated representation learning, in which informative graph representations are learned directly from

the data rather than being completely specified through manually selected graph metrics. Graph embeddings, graph neural networks (GNNs), graph convolutional networks, graph attention mechanisms, and related architectures can learn latent representations by aggregating structural and attribute information. Handcrafted and automated representation strategies involve different trade-offs among predictive flexibility, computational complexity, feature engineering requirements, and interpretability.

The methodological diversity of graph-based cyber-attack detection is also reflected in the existing literature. Previous systematic analysis has shown that graph-based approaches have been investigated across multiple cybersecurity application domains using different graph construction, feature extraction, and learning strategies. These studies include both explicitly defined graph-based features and data-driven representation-learning approaches, highlighting that representation strategy constitutes an important methodological design choice in graph-based cyber-attack detection (Kanca & Türker, 2025).

The diversity of graph representations also introduces an additional methodological problem. Constructing a graph does not by itself determine which structural information is useful for cyber-attack detection. When handcrafted graph representations are employed, numerous topological metrics can potentially be extracted from the same graph. Although these metrics characterize complementary properties of graph structure, their contribution to attack classification may differ considerably. Furthermore, calculating a large number of graph metrics can substantially increase computational cost, particularly when graphs are generated repeatedly across multiple traffic features and temporal windows. Consequently, graph-based intrusion detection requires consideration not only of how the graph is constructed, but also of which structural descriptors should be retained.

Interpretability represents another important dimension of this problem. A graph-based representation may achieve high classification performance without necessarily explaining which structural characteristics distinguish malicious traffic from benign traffic. Examining attack-specific topological patterns can help bridge this gap. Instead of treating graph-derived features exclusively as inputs to a classifier, their distributions can be analyzed to determine whether particular attacks are associated with systematic changes in connectivity, clustering, centrality, path structure, or community organization. Such analysis provides a connection between predictive modeling and structural characterization of network behavior.

Previous NVG-based analysis provides evidence for this perspective. In the attack-specific topological study by Kanca et al. (Kanca and Türker, 2026) 76 numerical network-traffic features were independently transformed into NVGs, and ten graph-theoretic metrics were calculated from each representation, resulting in 760 topological descriptors per traffic frame. Beyond evaluating the discriminative capability of these representations, the study examined attack-versus-benign differences across traffic-feature-topology combinations. Of 10,640 comparisons, 7,777 remained statistically significant following false discovery rate correction, and 4,844 exhibited large effect sizes. The findings indicated that attack-related structural differences were distributed across multiple combinations of traffic characteristics and graph metrics rather than being represented by a single universal topological descriptor.

This observation naturally raises a further methodological question: if a large collection of graph descriptors contains potentially discriminative information, are all of those descriptors required for effective detection? Different topological metrics may capture overlapping structural characteristics, while their computation can increase the overall cost of graph-based analysis.

Feature-importance and feature-selection methods therefore provide a potential mechanism for identifying informative descriptors, reducing redundancy, and constructing more compact graph representations. This consideration establishes a direct connection between structural interpretability and computational efficiency.

Taken together, these observations suggest that graph-based cyber-attack detection should not be formulated solely as a classification problem. It can instead be viewed as a sequence of interrelated representation and design decisions. The raw security data must first be associated with an appropriate graph construction mechanism. The resulting graph must then be represented using handcrafted or automatically learned information. When handcrafted topology is employed, relevant structural characteristics must be identified and interpreted. Redundant or computationally expensive information may subsequently need to be reduced, and the resulting representation must ultimately be evaluated in terms of predictive performance, robustness, interpretability, and computational efficiency.

Accordingly, this chapter conceptualizes graph-based cyber-attack detection as an integrated design pipeline rather than as a single model-selection problem. The central question addressed throughout the chapter is: “How can cybersecurity data be transformed into graph representations, structurally characterized, interpreted, reduced, and evaluated to support predictive, interpretable, and computationally efficient cyber-attack detection?”

To address this question, the chapter connects three complementary levels of analysis. The first is the representation level, which considers graph construction alternatives and handcrafted and automated representation strategies. The second is the structural level, which examines how graph topology can characterize network traffic and reveal attack-specific patterns, with particular emphasis on NVG-based representations. The third is the

metric and efficiency level, which considers the relative importance of topological descriptors, representation reduction, predictive performance, and computational cost.

The contribution of the chapter therefore lies primarily in methodological integration and synthesis. It does not present previously reported experiments as newly generated evidence. Instead, selected findings from the authors' published and publicly available prior work are used to support the relevant methodological discussions. These findings are considered within a broader methodological framework that connects graph construction, representation learning, topological interpretation, feature importance, representation reduction, computational efficiency, and model evaluation. This organization provides a basis for understanding not only whether graph-based methods can classify cyber-attacks, but also what structural information they capture, how that information can be interpreted, which components may be informative, and what methodological considerations are required for practical deployment.

Finally, the chapter identifies several issues that remain central to the development of graph-based intrusion-detection systems, including cross-dataset generalization, class imbalance, graph-construction complexity, computational scalability, explainability, and dynamic graph modeling. By considering these issues within a common design framework, the chapter aims to provide a structured foundation for both interpreting existing graph-based cyber-attack detection research and designing future systems.

Graph-Based Representation of Cyber-Security Data

Why Graphs?

A graph $G=(V,E)$ represents a set of nodes V and relationships E . In cyber-security, a node can represent a host, IP address, user, process, packet, flow, function, event, or time-indexed

observation. An edge can encode communication, similarity, dependency, temporal visibility, invocation, causality, or another domain-specific relation. This flexibility makes graph modeling particularly useful when malicious behavior is expressed through interactions rather than isolated feature values.

Graph construction is therefore not a neutral preprocessing step. The choice of nodes and edges determines which structural properties are available to downstream algorithms. A host-communication graph emphasizes connectivity and interaction patterns, whereas a function-call graph captures program behavior. A visibility graph instead maps a numerical sequence into a network and is therefore suited to analyzing the structural organization of sequential observations.

Major Graph Construction Strategies

Graph construction is one of the fundamental stages of graph-based cyber-security analysis because it determines how the entities and relationships contained in the original data are represented structurally. Cyber-security data can originate from heterogeneous sources, including network flows, packet sequences, communication records, system events, application behavior, and temporal observations. Consequently, the definition of nodes and edges may vary considerably depending on the characteristics of the data and the security problem being investigated. Different graph construction strategies therefore emphasize different aspects of malicious behavior.

In network-oriented security applications, communication graphs provide one of the most intuitive representations. In this approach, nodes typically correspond to communicating entities such as hosts or IP addresses, while edges represent communication between these entities. Such a representation shifts the analysis from individual traffic records toward the interaction structure of the

network. For example, unusual connectivity patterns, repeated interactions, or changes in communication behavior may provide useful structural information for identifying intrusion, botnet activity, or lateral movement (Yao et al., 2019).

A related approach is the flow graph, which focuses more directly on network-flow information. Depending on the particular design, nodes may represent flows, communicating endpoints, or other flow-related entities, while edges describe relationships among them. Flow graphs can incorporate information derived from source and destination addresses, ports, protocols, traffic direction, or other flow characteristics. They are therefore particularly relevant to network intrusion and denial-of-service analysis, where malicious activity may produce characteristic communication and traffic patterns (Yao et al., 2018).

Visibility graphs follow a substantially different construction principle. Rather than directly representing communicating entities, visibility-based methods transform an ordered numerical sequence into a graph. Individual observations are represented as nodes, and edges are established according to predefined visibility criteria. Variants such as the Visibility Graph (VG), Horizontal Visibility Graph (HVG), and Natural Visibility Graph (NVG) provide different mechanisms for determining these relationships. In cyber-security applications, numerical network-traffic features can be considered as sequential observations and transformed into visibility graphs. The resulting graph structure can then be characterized through topological metrics, allowing structural properties of changing traffic behavior to be examined.

Outside conventional network-traffic analysis, function-call graphs provide a useful representation of program behavior. Nodes typically correspond to functions, procedures, or API calls, while edges indicate invocation relationships. Instead of describing network communication, these graphs characterize how software

components interact during execution or within program structure. Consequently, function-call graphs are particularly relevant to malware analysis, where malicious and benign programs may exhibit different patterns of function or API usage (Z. Zhao et al., 2013).

Another important representation is the provenance graph. Provenance-oriented approaches model relationships among system entities and events, with edges representing dependencies, information flows, or causal relationships. Such graphs can provide a broader view of how activities propagate through a system. This is particularly useful when an attack cannot be sufficiently characterized by a single isolated event. Multi-stage attacks and advanced persistent threats may involve sequences of related actions, and provenance graphs can support the reconstruction and analysis of these attack chains (Li et al., 2022).

Cyber-security behavior is also inherently time-dependent. For this reason, dynamic or temporal graphs extend conventional static graph representations by incorporating changes in nodes, edges, attributes, or interactions over time. Instead of representing network behavior through a single fixed structure, these approaches can capture how relationships evolve. Temporal graph representations may therefore be useful for detecting attacks characterized by gradual behavioral changes, recurring interactions, or sequences of malicious activities (J. Zhao et al., 2026).

These graph construction strategies are summarized in Table 1.

Table 1. Major graph construction strategies in cyber-security analysis.

Graph type	Typical unit/relationship	Potential cyber-security use
Communication graph	Hosts/IPs and communication links	Intrusion, botnet, lateral-movement analysis
Flow graph	Flows or communicating endpoints	Network intrusion and DDoS characterization
Visibility graph (VG/HVG/NVG)	Sequential observations linked by visibility	Structural analysis of time-varying traffic features
Function-call graph	Functions/API calls and invocation edges	Malware and program-behavior analysis
Provenance graph	Entities/events connected by causal provenance	Advanced persistent threat and attack-chain analysis
Dynamic/temporal graph	Time-evolving nodes and edges	Changing network behavior and temporal attacks

As shown in Table 1, graph construction strategies differ primarily in what is represented as a node and what type of relationship is represented as an edge. This distinction is methodologically important because different graph constructions preserve different forms of information. A communication graph emphasizes interactions among network entities, whereas a function-call graph emphasizes relationships within software behavior. Provenance and temporal graphs provide information about causal or time-dependent relationships, while visibility graphs reveal structural characteristics embedded in sequential numerical observations.

Therefore, the selection of a graph construction strategy should be aligned with both the characteristics of the available security data and the objective of the detection task. There is no single graph structure that is inherently appropriate for every cyber-

security problem. Instead, graph construction determines which structural characteristics become available for subsequent analysis and learning. After the graph has been constructed, a second methodological decision is required: how should the information contained in that graph be represented for machine-learning or deep-learning models? This question leads to handcrafted representation and automated representation learning, which are discussed in the following section.

Representation and Learning Strategies

Once cyber-security data have been transformed into a graph, the next important question concerns how the information contained in the graph should be represented for subsequent analysis and classification. Graph construction determines the entities and relationships preserved from the original data, whereas the representation strategy determines how this structural information is provided to a machine-learning or deep-learning model.

Graph-based cyber-security approaches can broadly be considered from two representation perspectives: handcrafted representation and automated representation learning. These strategies differ in the extent to which structural information is explicitly defined by the researcher or automatically learned from data. Each strategy offers different advantages in terms of interpretability, flexibility, computational requirements, and dependence on feature engineering.

Handcrafted Representation

Handcrafted graph representation describes a graph through a predefined set of structural or topological measures. Instead of directly providing the complete graph structure to a learning algorithm, relevant characteristics of the graph are calculated and organized as a feature vector. Common examples include degree-based measures, graph density, clustering coefficient, shortest-path

characteristics, graph diameter, centrality measures, modularity, and community-related properties.

These metrics characterize different aspects of graph structure. Degree-related measures describe node connectivity, whereas density provides information about the overall level of connectivity in the graph. Clustering coefficients characterize the tendency of nodes to form locally interconnected structures, while path-based measures describe aspects of network-wide organization(Watts & Strogatz, 1998). Betweenness and closeness centrality characterize different aspects of the structural position of nodes within a network(Freeman, 1978). Eigenvector-based centrality additionally considers the importance of the nodes to which a node is connected(Bonacich, 1972). Modularity and community-related measures characterize the organization of networks into groups or densely connected substructures(Newman & Girvan, 2004).

In cyber-security applications, these descriptors can transform complex relational structures into numerical representations that can subsequently be processed by conventional machine-learning or deep-learning models. One of the principal advantages of this approach is interpretability. Because each feature has a graph-theoretic definition, differences observed between benign and malicious behavior can potentially be associated with specific structural characteristics.

However, handcrafted representations also introduce important design decisions. The researcher must determine which topological measures should be calculated, and a predefined metric set may not capture every task-specific pattern contained in the original graph. Furthermore, extracting a large number of graph metrics can introduce redundant information and increase computational cost. Therefore, the selection and importance of handcrafted topological descriptors become important

considerations in the design of graph-based cyber-attack detection systems.

Automated Representation Learning

Automated representation learning follows a different principle. Rather than describing the graph exclusively through manually selected topological measures, the model learns informative representations directly from graph structure and, where available, node or edge attributes.

Graph embeddings and Graph Neural Networks (GNNs) are representative examples of this approach. Graph-learning architectures can aggregate information from neighboring nodes and learn latent representations that reflect both local and broader structural relationships. Depending on the task, these learned representations can subsequently be used for node classification, edge prediction, graph classification, anomaly detection, or other security-related objectives (Lo et al., 2022).

The principal advantage of automated representation learning is its ability to reduce dependence on manually specified graph descriptors (Caville et al., 2022; Lo et al., 2022). Potentially complex and nonlinear structural patterns can be learned directly from data, which may be particularly valuable when the relevant graph characteristics are difficult to define in advance.

However, automated representation learning also presents challenges. Learned latent features generally do not have the same direct semantic interpretation as explicitly calculated topological measures. Consequently, understanding why a particular graph or observation has been classified as malicious can become more difficult. Model complexity, computational requirements, training-data requirements, and scalability to large graphs must also be considered.

Thus, automated representation learning should not simply be interpreted as a replacement for handcrafted representation. Rather, it represents a different mechanism for extracting useful information from graph-structured cyber-security data.

Evidence from Literature

The diversity of graph representation strategies described above is also reflected in the existing cyber-security literature. Graph-based approaches have been investigated across a range of application domains, including intrusion and network intrusion detection, botnet and distributed denial-of-service detection, IoT security, malware analysis, and advanced threat detection. Across these domains, graph construction and representation are closely related to the type of security data being analyzed and the structural relationships that the detection task seeks to preserve.

Handcrafted approaches characterize graph structure through explicitly defined descriptors such as centrality, connectivity, clustering, path-based measures, and other structural properties. These representations provide a direct connection between the original graph and the numerical features supplied to a learning algorithm, which can support structural interpretation of malicious behavior. In contrast, automated representation-learning approaches, including graph embeddings and GNN-based architectures, learn latent representations directly from graph structure and associated attributes (Caville et al., 2022; Lo et al., 2022). Such approaches reduce dependence on manually specified descriptors and enable task-oriented structural patterns to be learned from data.

The literature also demonstrates that the suitability of a representation strategy depends on factors such as graph construction, data characteristics, detection objective, computational requirements, and the desired level of interpretability. Handcrafted and automated approaches therefore represent different

methodological choices for extracting and utilizing structural information from graph-based cyber-security data. Consequently, representation strategy is better understood as a methodological design choice than as a fixed component of graph-based cyber-attack detection.

From this perspective, an important distinction emerges between constructing a graph and representing the information contained in that graph. Graph construction determines which entities and relationships are preserved, whereas representation determines how this structural information becomes available to subsequent learning and analysis. This distinction provides the basis for the next part of the chapter, which focuses on Natural Visibility Graphs as a specific mechanism for transforming sequential network-traffic observations into graph structures.

Natural Visibility Graphs as a Structural Representation

Among graph construction approaches, visibility graphs provide a distinctive mechanism for converting sequential numerical data into graph structures. Unlike communication, flow, or provenance graphs, where nodes and edges generally correspond to explicitly defined entities and interactions, visibility graphs derive their connectivity directly from the geometric relationships among observations in an ordered sequence. This property makes them particularly relevant when the original data consist of numerical measurements whose temporal or sequential organization may contain discriminative information.

In network-traffic analysis, many flow-based features can be arranged as sequences of consecutive observations. Packet statistics, flow duration, inter-arrival-time characteristics, packet-length measures, and forward or backward traffic properties may exhibit different local patterns under benign and malicious conditions. Rather than analyzing these observations only as conventional

numerical values, visibility-graph transformation provides an alternative structural representation. Once the sequence has been converted into a graph, its organization can be characterized using graph-theoretic measures such as connectivity, clustering, path structure, centrality, and community organization.

From a Numerical Sequence to an NVG

Natural Visibility Graph (NVG) transformation maps an ordered numerical sequence into a graph while preserving information about the relative magnitude and organization of its observations (Lacasa et al., 2008). Consider a numerical sequence

$$X = \{x_1, x_2, \dots, x_n\}$$

where each observation x_i corresponds to a position t_i in the ordered sequence. During NVG construction, each observation is represented as a node. Therefore, a sequence containing n observations produces a graph containing n nodes (Lacasa et al., 2008).

The edges of the graph are determined according to the natural visibility criterion. Two observations x_i and x_j , where $t_i < t_j$, are connected when every intermediate observation x_k satisfies the visibility condition

$$x_k < x_i + (x_j - x_i) \frac{t_k - t_i}{t_j - t_i}, \quad t_i < t_k < t_j.$$

Geometrically, this condition means that a straight line drawn between two observations must remain above all intermediate observations. If no intermediate observation blocks this line of sight, the corresponding nodes are connected by an edge. Adjacent observations are therefore naturally connected, while additional long-range edges are determined by the relative values of the observations located between them (Lacasa et al., 2008).

This construction mechanism converts variations in the original numerical sequence into variations in graph connectivity. Peaks, valleys, local fluctuations, and broader changes in the sequence can influence which observations are mutually visible. Consequently, two numerical sequences containing different structural patterns may generate graphs with different degree distributions, clustering behavior, path characteristics, centrality patterns, or community structures.

The important point is that NVG transformation does not merely calculate another statistical feature from the original sequence. Instead, it changes the representation domain of the data:

Numerical sequence $\rightarrow$ visibility relationships $\rightarrow$ graph structure $\rightarrow$ topological descriptors

This transformation makes it possible to examine a numerical traffic feature from a graph-theoretic perspective. Conventional statistics may summarize characteristics such as mean, variance, minimum, or maximum values, whereas an NVG-based representation enables the structural organization of observations within a local sequence to be characterized.

For network-traffic analysis, the transformation can be applied independently to multiple numerical traffic features. Suppose that a traffic dataset contains m numerical features and that a local frame contains n consecutive flow observations. For each feature, the n values within the frame can be interpreted as an independent numerical sequence and transformed into a separate NVG. A single traffic frame can therefore produce multiple graphs, each representing the local structural behavior of a different traffic characteristic (Kanca & Türker, 2026).

For example, a frame may contain sequential observations for packet-length statistics, flow duration, inter-arrival times, forward packet characteristics, and backward packet characteristics.

Rather than combining all of these values into a single graph, the feature-wise NVG approach constructs a separate graph for each feature. The resulting collection of graphs represents different structural dimensions of the same traffic interval.

This distinction is important because different traffic features may respond differently to malicious behavior. An attack may produce substantial structural changes in packet timing while having a smaller effect on another traffic characteristic. Feature-wise graph construction allows these differences to be retained instead of immediately aggregating them into a single representation.

After graph construction, topological measures can be extracted from each NVG. These measures convert the graph structure into numerical descriptors that can subsequently be used for statistical analysis, machine-learning classification, deep-learning models, or interpretability-oriented analysis. In this sense, NVG provides a bridge between sequential network-traffic data and graph-based structural analysis.

The resulting processing chain can be expressed as:

Network Traffic → Numerical Feature → Local Sequence →
NVG Construction → Topological Metrics → Learning/Statistical
Analysis

This framework is particularly relevant when the research objective extends beyond predictive performance. Because each traffic feature produces an explicit graph and each graph can be characterized through interpretable topological measures, it becomes possible to investigate not only whether malicious traffic can be classified, but also which traffic characteristics and which structural graph properties distinguish attack behavior from benign behavior.

Framing and Labeling Considerations

Before NVGs can be constructed from network-traffic data, the continuous or ordered sequence of flow observations must generally be divided into manageable local segments. A common approach is sliding-window framing, in which consecutive observations are grouped into fixed-length frames.

Two parameters are particularly important in this process: window length (WL) and step size. Window length determines the number of consecutive observations included in each frame and, consequently, the number of nodes in each feature-specific NVG. Step size determines how far the window moves before the next frame is generated.

For example, if a window contains 100 observations and the step size is 50, two consecutive frames overlap by 50 observations, corresponding to 50% overlap. Overlapping windows can increase the number of available frames and allow neighboring segments of traffic behavior to be represented with greater continuity.

Window length represents an important design parameter because it determines the amount of local context represented by each graph. A relatively short window generates smaller graphs and may better preserve highly localized changes in traffic behavior. It can also reduce graph-construction and topological-feature extraction costs. However, a very short window may provide insufficient context for representing more complex structural patterns.

Conversely, larger windows contain more observations and can capture broader patterns of traffic behavior. At the same time, they produce larger graphs, potentially increasing the computational cost of graph construction and topological analysis. Large windows may also combine multiple local behaviors within the same frame. Window selection therefore involves a trade-off among

temporal/local resolution, structural information, sample availability, and computational complexity.

A second issue arises from frame labeling. In flow-based intrusion-detection datasets, individual observations typically have their own benign or attack labels. After multiple observations are grouped into a frame, however, a single label must generally be assigned to the entire frame for supervised classification.

If every observation within the frame belongs to the same class, frame labeling is straightforward. Mixed frames are more challenging because they may contain observations from multiple classes or a combination of benign and malicious traffic. Several labeling strategies can be considered. Majority voting assigns the most frequent class within the frame. An attack-prioritized strategy can assign an attack label whenever malicious observations are present, depending on the detection objective. Other application-specific rules may also be defined according to attack prevalence or temporal position.

There is no universally appropriate labeling strategy for every intrusion-detection task. The selected rule changes the semantic meaning of the resulting frame and may influence class distributions and classification results. Therefore, the labeling mechanism should be explicitly defined rather than treated as an incidental preprocessing detail.

Framing and labeling should consequently be considered integral components of NVG-based representation design. The graph presented to the learning algorithm is determined not only by the NVG visibility criterion, but also by which observations are grouped together, how much consecutive frames overlap, and how the resulting graph is labeled. These choices influence the structural information contained in the graph and the characteristics of the resulting representation.

Topological Characterization

After a numerical network-traffic sequence has been transformed into an NVG, the resulting graph must be characterized in a form suitable for statistical analysis or learning algorithms. Although the graph itself preserves structural relationships among observations, its complete adjacency structure may be difficult to interpret directly. Topological metrics provide a mechanism for summarizing different properties of this structure through numerical descriptors.

A single topological metric is generally insufficient to characterize all aspects of an NVG. Graph structure can be examined from several complementary perspectives, including overall connectivity, local organization, node centrality, path structure, and community organization. For this reason, multiple graph-theoretic measures can be extracted from each NVG to obtain a more comprehensive description of the underlying sequence.

Connectivity measures, such as average degree and graph density, describe how strongly the nodes of an NVG are connected. Average degree summarizes the typical number of connections associated with nodes, whereas density represents the proportion of possible edges that are actually present in the graph. In the context of visibility graphs, changes in these measures indicate that the visibility relationships among observations have changed. Therefore, traffic sequences with different numerical organization may produce different levels of graph connectivity.

Local organization can be characterized using measures such as the clustering coefficient. The clustering coefficient describes the extent to which neighboring nodes are also connected to one another. It therefore provides information about the formation of locally interconnected structures within the graph. In an NVG representation, changes in local variation or repeated structural

patterns within a traffic feature may affect the formation of these local connections.

Centrality measures provide another perspective by characterizing the structural position or importance of nodes within the graph. Degree centrality focuses on the number of direct connections associated with a node. Betweenness centrality reflects the extent to which a node lies on shortest paths between other nodes, while closeness centrality describes how close a node is to the remaining nodes in terms of graph distance. Eigenvector centrality additionally considers the importance of the nodes to which a particular node is connected. When these measures are summarized across an NVG, they can provide information about how structural influence or connectivity is distributed throughout the sequence.

Path-based measures characterize the global organization of the graph. Average shortest-path length represents the typical number of edges required to connect pairs of nodes through shortest paths, whereas graph diameter represents the greatest shortest-path distance within the graph. These measures provide information about the overall extent and reachability of the graph. Changes in the sequential organization of network-traffic observations may alter long-range visibility relationships and consequently influence these path characteristics.

Finally, community-related measures examine whether the graph can be separated into internally cohesive groups. Modularity evaluates the strength of community organization, while the number of detected communities indicates how many such groups are identified under a particular community-detection procedure. These measures provide a mesoscopic perspective between local node-level characteristics and global graph-level properties.

The principal metric families and their structural interpretations are summarized in Table 2.

Table 2. Principal topological metric families and their structural interpretations.

Metric family	Examples	Structural interpretation
Connectivity	Average degree, density	Overall level of graph connectivity
Local organization	Clustering coefficient	Tendency of neighboring nodes to form locally connected structures
Centrality	Betweenness, closeness, eigenvector, degree centrality	Relative structural importance or position of nodes
Path structure	Average shortest path, diameter	Global reachability and graph extent
Community structure	Modularity, number of communities	Mesoscopic organization and separation into groups

Taken together, the metric families summarized in Table 2 provide complementary descriptions of an NVG. Connectivity measures indicate how strongly the graph is connected; clustering describes local organization; centrality measures characterize the structural roles of nodes; path-based measures describe global reachability; and community measures characterize intermediate-scale organization. Consequently, the original numerical sequence is no longer represented only by its individual values but also by a set of descriptors reflecting its structural organization.

For cyber-attack detection, this transformation is particularly relevant because malicious behavior may modify the organization of network-traffic features in different ways. An attack does not necessarily produce a uniform increase or decrease in every traffic variable. Instead, it may alter packet timing, traffic direction, packet-length characteristics, flow rates, or other attributes in different patterns. When these feature sequences are transformed into NVGs, such changes may be reflected through differences in connectivity, clustering, centrality, path structure, or community organization.

Therefore, topological characterization provides two related benefits. First, the extracted metrics can serve as numerical inputs to machine-learning or deep-learning models for attack classification. Second, the same metrics can be analyzed statistically to investigate which structural properties differ between benign and malicious traffic. The latter is especially important for interpretability because it shifts the analysis from simply asking whether a model can distinguish attacks toward examining how attack behavior is reflected in graph topology.

However, extracting a large number of topological metrics also introduces an important methodological question. Different metrics may capture overlapping structural information, and their relevance may vary according to the traffic feature and attack class being examined. Thus, the availability of many graph descriptors does not imply that all of them are equally informative.

This observation motivates two subsequent questions addressed in the following sections: Do different cyber-attack classes exhibit distinguishable topological characteristics, and which topological metrics contribute most strongly to their discrimination?

Topology as an Indicator of Attack Behavior

Graph-based cyber-attack detection becomes more informative when topological representations are used not only for classification but also for understanding how malicious traffic differs structurally from benign behavior. From this perspective, attack-specific topological analysis examines whether changes in network-traffic characteristics are reflected in NVG properties such as connectivity, clustering, centrality, and path structure.

In an attack-specific NVG analysis by Kanca and Türker (Kanca and Türker, 2026), 76 numerical network-traffic features were independently transformed into NVGs, and ten graph-theoretic metrics were extracted from each feature, resulting in 760

topological descriptors per traffic frame. A multi-branch CNN achieved an average accuracy of 96.20% and an MCC of 0.9566 under stratified five-fold cross-validation. Beyond classification performance, the analysis investigated whether individual cyber-attack classes exhibited distinguishable topological patterns relative to benign traffic.

For this purpose, 14 attack classes were compared with benign traffic across 10,640 traffic-feature-topology combinations. After false-discovery-rate correction, 7,777 comparisons (73.1%) remained statistically significant, while 4,844 exhibited large Cliff's delta effect sizes. These results provide empirical evidence that malicious traffic can be associated with substantial structural differences in NVG representations.

Importantly, the strongest differences were not associated with a single universal topological metric. Instead, they emerged from specific combinations of traffic characteristics and graph properties. Features related to backward traffic, packet length, and inter-arrival behavior frequently showed strong associations, while degree, clustering, and eigenvector-centrality measures captured different forms of attack-related structural variation. These findings indicate that topological interpretation should consider the interaction between the underlying traffic feature, the selected graph descriptor, and the attack class rather than relying on a single universal metric. Representative examples are presented in Table 3.

Table 3. Representative attack-specific traffic-feature–topology signatures in NVG representations.

Attack example	Traffic feature	Topological metric	Cliff's δ
Infiltration	Fwd Pkts/s	Average Degree	-0.9910
Bot	Fwd Act Data Pkts	Average Clustering	+0.8474
FTP-BruteForce	Pkt Size Avg	Eigenvector Centrality	+0.9946
Brute Force-Web	Flow IAT Std	Average Degree	-0.9994
SQL Injection	Fwd IAT Tot	Average Degree	-0.9999
SSH-Bruteforce	Tot Bwd Pkts	Eigenvector Centrality	-0.9936

As shown in Table 3, different attacks may be associated with different traffic-feature–topology signatures. However, these findings were obtained from a single benchmark dataset, and some attack classes contained substantially fewer observations than others. Therefore, the identified signatures should be validated across additional datasets and network environments before being considered general attack-specific patterns.

It should also be noted that statistical separation between attack and benign distributions does not necessarily imply predictive importance within a classification model; these represent related but distinct analytical perspectives.

Topological Metric Importance and Explainability

Why Metric Importance Matters

Graph-based representations can produce a large number of topological descriptors, particularly when multiple graph metrics are extracted from several traffic features. Although these descriptors characterize different structural properties, they may not contribute equally to cyber-attack classification. Some metrics may contain

highly discriminative information, whereas others may provide redundant or limited additional information.

Computing a large metric set can also increase feature-extraction time, memory requirements, and model dimensionality. Therefore, metric-importance analysis has two complementary objectives. First, it improves interpretability by identifying which structural graph properties contribute most strongly to the detection task. Second, it supports representation reduction by identifying smaller subsets of potentially informative metrics. This creates a direct connection between explainability and computational efficiency.

Complementary Importance Methods

Feature importance can be evaluated from different methodological perspectives. A metric considered important by one method may receive a different ranking from another because the methods measure importance according to different principles. For this reason, combining complementary approaches can provide a broader assessment than relying on a single importance method.

Table 4. Complementary feature-importance and feature-selection methods for evaluating topological metrics.

Method	Core idea	Interpretive role
SHAP	Attributes model predictions to input features using Shapley-value principles	Model-oriented contribution
Grouped Permutation Feature Importance (GPFI)	Jointly permutes features within a predefined group and measures the resulting change in model performance	Group-level predictive dependency
Boruta	Compares real variables with randomized shadow variables	Relevance screening
Recursive Feature Elimination (RFE)	Iteratively removes less important variables	Relative ranking / subset selection

These methods answer related but different questions. SHAP focuses on the contribution of features to model predictions (Lundberg & Lee, 2017), while Grouped Permutation Feature Importance evaluates how model performance changes when the information associated with a predefined feature group is jointly disrupted (Au et al., 2022). Boruta focuses on determining whether features contain relevant information compared with randomized alternatives (Kursa & Rudnicki, 2010), whereas RFE progressively removes less important features to obtain a relative ranking or reduced subset (Guyon et al., 2002).

Integrating Importance Evidence

Because feature-importance methods evaluate relevance according to different principles, their outputs should not necessarily be expected to produce identical rankings. A topological descriptor may appear important because it strongly influences model predictions, because disrupting its information reduces predictive performance, or because it consistently provides information beyond randomized alternatives. These perspectives are related but not equivalent.

For this reason, importance analysis can benefit from considering evidence from multiple complementary methods rather than relying exclusively on a single ranking. An integrated importance analysis can consequently be used to identify candidate descriptors for subsequent representation reduction. However, importance ranking alone is not sufficient evidence that lower-ranked metrics can be removed without affecting detection capability. Any reduced representation should be evaluated again using the same predictive criteria applied to the original representation, while also considering computational cost and the stability of the selected metrics across validation partitions.

This distinction is important for efficient graph-based cyber-attack detection. Feature importance identifies potentially informative structural properties, whereas feature reduction determines whether a smaller representation can preserve sufficient information for the detection task. The transition from importance analysis to representation reduction therefore requires explicit empirical validation rather than assuming that the highest-ranked descriptors automatically constitute an optimal subset.

Performance–Efficiency Trade-Offs

The extraction of graph-theoretic descriptors introduces an additional computational stage into graph-based cyber-attack detection. As the number and complexity of topological metrics increase, the resulting representation may provide a richer structural description of network traffic, but it may also increase feature-extraction time, memory requirements, and overall processing cost. Efficient model design therefore requires considering predictive capability and computational requirements jointly rather than evaluating classification performance in isolation.

Feature-importance analysis provides a natural starting point for representation reduction. After candidate topological descriptors have been ranked or evaluated, progressively smaller metric subsets can be constructed and compared with the full representation. Such subsets may be defined according to individual importance rankings, consensus across multiple importance methods, redundancy analysis, or combinations of these criteria. The objective is not necessarily to identify the smallest possible representation, but to determine whether redundant or weakly informative descriptors can be removed without introducing an unacceptable loss of predictive information.

The evaluation of reduced representations should consider multiple dimensions. Predictive performance can be assessed using

metrics such as accuracy, weighted or macro-averaged F1-score, and Matthews correlation coefficient, while computational evaluation can include graph-construction time, topological feature-extraction time, model training time, inference time, and memory consumption. Reporting these components separately is particularly useful because the computational bottleneck in a graph-based pipeline may arise before the classification model itself is executed.

A reduced representation should also be evaluated under an experimental design that separates feature selection from final performance estimation. If the same validation data are repeatedly used both to select the most informative descriptors and to report the final performance of the reduced representation, the resulting estimate may be optimistic. Independent holdout evaluation or nested cross-validation can provide a more rigorous assessment when the available data and computational resources permit.

From this perspective, efficiency should be interpreted as a performance–cost trade-off rather than simply as dimensionality reduction. A compact representation is useful only when its computational advantages are accompanied by acceptable predictive capability and sufficient stability across validation partitions, datasets, or network environments. Consequently, the preferred representation may differ according to the operational context: offline forensic analysis may tolerate computationally expensive structural descriptors, whereas near-real-time intrusion detection may place greater emphasis on extraction and inference latency.

An Integrated Design Framework for Graph-Based Cyber-Attack Detection

The previous sections demonstrate that graph-based cyber-attack detection involves several interconnected design decisions. Detection performance depends not only on the selected classifier but also on how security data are transformed into graphs, how

structural information is represented, which topological properties are retained, and how computational efficiency and interpretability are evaluated. Based on these considerations, the main stages can be organized into an integrated design framework.

The proposed framework is not restricted to NVG-based representations. Instead, NVG serves as a concrete example of how graph construction, topological characterization, metric importance, and representation reduction can be connected within a broader graph-based cyber-attack detection methodology.

Define the security observation unit.

The first step is to determine what will be represented in the analysis, such as flows, packets, hosts, processes, system events, function calls, or sequential traffic-feature values. This decision establishes the basic unit from which the graph will be constructed.

Select the graph construction strategy.

Nodes and edges should be defined according to the security problem and available data. Communication, flow, provenance, function-call, temporal, or visibility graphs may capture different aspects of malicious behavior.

Choose the representation strategy.

The constructed graph can be represented through handcrafted topological descriptors or automatically learned representations. The selection should consider the objectives of the analysis, particularly predictive capability and interpretability.

Characterize structural information.

Relevant graph properties should be extracted or learned. Depending on the problem, these may include connectivity, clustering, centrality, path structure, community organization, or temporal characteristics.

Evaluate predictive and structural information.

Evaluation should not be limited to classification metrics. When interpretability is an objective, benign and malicious traffic can also be compared in terms of their structural graph characteristics.

Assess metric importance and redundancy.

Evaluate the contribution of topological descriptors using complementary importance or feature-selection methods, while considering redundancy, feature interactions, and stability across validation partitions.

Reduce the representation where appropriate.

Define progressively reduced metric subsets based on importance and redundancy evidence, and compare them with the full representation in terms of predictive capability and computational requirements.

Validate the resulting model.

Evaluate the selected representation using an appropriate validation strategy and, where feasible, independent holdout data or nested cross-validation to reduce selection bias and assess generalizability.

Evaluate computational requirements.

Graph construction, topological metric extraction, model training, inference time, memory requirements, and scalability should be considered when assessing practical applicability.

Preserve interpretability.

Where possible, detection results should be related back to the original security data and graph structure. This can help explain

not only whether malicious behavior is detected but also which structural characteristics contribute to the decision.

Overall, the framework connects the major stages of graph-based cyber-attack detection as follows:

Security Data → Graph Construction → Graph Representation → Structural Characterization → Classification and Structural Analysis → Importance Assessment → Representation Reduction → Validation → Efficiency and Interpretability

This integrated perspective emphasizes that graph-based cyber-attack detection should be approached as an end-to-end representation and analysis problem, rather than solely as a model-selection problem. The effectiveness of the final system depends on the interaction among graph construction, representation, structural information, learning, validation, and computational requirements.

Challenges and Open Research Directions

Despite the potential of graph-based representations for cyber-attack detection, several methodological and practical challenges remain. High classification performance on a benchmark dataset does not necessarily guarantee that the same graph representation, topological characteristics, or selected metrics will remain effective in different network environments. Future research should therefore consider not only predictive performance but also generalization, evaluation reliability, computational scalability, and interpretability.

Generalization Across Datasets and Environments

Topological patterns identified in a particular dataset may partly reflect its traffic-generation process, feature distributions, attack composition, or network environment. Therefore, attack-specific signatures and important graph metrics should be validated across independent datasets before being considered generalizable.

Future studies should investigate whether similar structural patterns remain stable across enterprise, IoT, cloud, and other operational network environments. Cross-dataset experiments would also help determine whether the importance rankings of topological metrics remain consistent under different traffic conditions.

Class Imbalance and Rare Attacks

Cyber-attack datasets often contain substantial differences in the number of observations available for different traffic classes (Chang & Branco, 2021). Consequently, overall accuracy alone may provide an incomplete description of model behavior, particularly for rare attack categories. Evaluation should therefore include class-sensitive measures such as macro-averaged F1-score, per-class precision and recall, and confusion matrices, together with more global measures such as weighted F1-score and Matthews correlation coefficient.

Computational Scalability

Graph construction and topological characterization can introduce substantial computational overhead, particularly when large numbers of traffic features, frames, or graph descriptors are processed (Kanca & Türker, 2025). Scalability should therefore be evaluated across the complete analysis pipeline rather than only at the classifier level. Future work should investigate more efficient graph-construction procedures, selective computation of topological descriptors, parallel processing, and representation-reduction strategies that balance structural information with computational requirements.

Explainability Beyond Feature Importance

Feature-importance methods can indicate which topological descriptors contribute to a model's predictions, but importance alone

does not fully explain why a particular traffic pattern is classified as malicious (Kanca & Türker, 2025). Future research should connect model-level explanations with graph-level structural changes and the underlying network-traffic characteristics from which those graphs are constructed. Such analysis may help distinguish between predictive importance and security-relevant structural interpretation.

Dynamic Graphs and GNN-Based Models

Static graph representations capture structural relationships within a defined observation interval but may not fully represent how those relationships evolve over time. Dynamic and temporal graph models provide a potential direction for incorporating structural evolution directly into cyber-attack detection (Kanca & Türker, 2025; J. Zhao et al., 2026; Zola et al., 2022). Future studies may also investigate advanced GNN-based architectures that capture evolving structural relationships while addressing the computational and interpretability challenges associated with automated graph representation learning.

Conclusion

Graph-based cyber-attack detection should be approached as an end-to-end representation and analysis problem rather than solely as a model-selection problem. The effectiveness of a graph-based detection system depends not only on the learning algorithm but also on how security data are mapped to nodes and edges, how the resulting structural information is represented, and which graph-derived characteristics are retained for subsequent analysis. Graph construction and representation should therefore be treated as central methodological design decisions that directly influence predictive capability, interpretability, and computational requirements.

Within this broader perspective, Natural Visibility Graphs provide a useful mechanism for transforming sequential network-traffic observations into graph structures by encoding relationships

determined by their temporal ordering and relative magnitudes. Evidence from attack-specific NVG analysis indicates that malicious behavior is not necessarily characterized by a single universal topological property; rather, different cyber-attack classes may exhibit distinct traffic-feature-topology signatures. This observation highlights the value of topological analysis not only as a source of predictive features but also as a means of structurally interpreting how malicious traffic differs from benign behavior.

At the same time, richer graph representations are not necessarily better representations. Extracting large collections of topological descriptors can introduce redundancy and computational overhead, making metric relevance and representation reduction important components of graph-based model design. Feature-importance and feature-selection methods can help identify potentially informative structural properties, but importance rankings should not be interpreted as direct evidence that a reduced representation is sufficient. Reduced metric sets should instead be validated explicitly in terms of predictive performance, stability, and computational cost. The central design objective is therefore not simply to maximize classification performance or minimize the number of descriptors, but to achieve an appropriate balance among predictive capability, structural interpretability, and computational efficiency.

Future research should extend this perspective through cross-dataset and cross-environment validation, improved treatment of class imbalance and rare attacks, and more systematic assessment of end-to-end computational scalability. Further research on dynamic graph models and graph neural networks may also improve the ability of automated representation-learning approaches to capture evolving structural patterns while maintaining computational scalability and interpretability. Progress in these directions could support graph-based cyber-attack detection systems that are not only

accurate, but also explainable, computationally practical, robustly evaluated, and transferable across heterogeneous network environments.

Kaynakça

- Au, Q., Herbringer, J., Stachl, C., Bischl, B., Casalicchio, G., Atzmueller, M., Fürnkranz, J., Kliegr, T., & Schmid, U. (2022). Grouped feature importance and combined features effect plot. *Data Mining and Knowledge Discovery 2022* 36:4, 36(4), 1401–1450. <https://doi.org/10.1007/S10618-022-00840-5>
- Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2(1), 113–120. <https://doi.org/10.1080/0022250X.1972.9989806>
- Caville, E., Lo, W. W., Layeghy, S., & Portmann, M. (2022). Anomal-E: A self-supervised network intrusion detection system based on graph neural networks. *Knowledge-Based Systems*, 258. <https://doi.org/10.1016/j.knosys.2022.110030>
- Chang, L., & Branco, P. (2021). *Graph-based Solutions with Residuals for Intrusion Detection: the Modified E-GraphSAGE and E-ResGAT Algorithms*. <http://arxiv.org/abs/2111.13597>
- Daya, A. A., Salahuddin, M. A., Limam, N., & Boutaba, R. (2019). A Graph-Based Machine Learning Approach for Bot Detection. *2019 IFIP/IEEE Symposium on Integrated Network and Service Management, IM 2019*, 144–152. <https://arxiv.org/abs/1902.08538v1>
- Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, 1(3), 215–239. [https://doi.org/10.1016/0378-8733\(78\)90021-7](https://doi.org/10.1016/0378-8733(78)90021-7)
- Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1–3), 389–422. <https://doi.org/10.1023/A:1012487302797/METRICS>

- Iacovacci, J., & Lacasa, L. (2016). Sequential visibility-graph motifs. *Physical Review E*, 93(4). <https://doi.org/10.1103/PHYSREVE.93.042309>
- Islam, R., Refat, R. U. D., Yerram, S. M., & Malik, H. (2020). Graph-Based Intrusion Detection System for Controller Area Networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(3), 1727–1736. <https://doi.org/10.1109/TITS.2020.3025685>
- Kanca, A. M., & Türker, İ. (2025). A Systematic Review of Graph-Based Representation Techniques for Cyber-Attack Detection Across Application Domains. *Concurrency and Computation: Practice and Experience*, 37(27–28). <https://doi.org/10.1002/CPE.70389>
- Kanca, A. M., & Türker, İ. (2026). GraphDL: A visibility graph-based structural representation framework for flow-based cyber-attack detection. *Scientific Reports* 2026. <https://doi.org/10.1038/S41598-026-55073-Z>
- Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with the Boruta Package. *Journal of Statistical Software*, 36(11), 1–13. <https://doi.org/10.18637/JSS.V036.I11>
- Lacasa, L., Luque, B., Ballesteros, F., Luque, J., & Nuno, J. C. (2008). From time series to complex networks: the visibility graph. *Proceedings of the National Academy of Sciences of the United States of America*, 105(13), 4972–4975. <https://doi.org/10.1073/pnas.0709247105>
- Li, J., Zhang, R., Liu, J., & Liu, G. (2022). LogKernel: A Threat Hunting Approach Based on Behaviour Provenance Graph and Graph Kernel Clustering. *Security and Communication Networks*, 2022(1), 4577141. <https://doi.org/10.1155/2022/4577141>

- Lo, W. W., Layeghy, S., Sarhan, M., Gallagher, M., & Portmann, M. (2022). E-GraphSAGE: A Graph Neural Network based Intrusion Detection System for IoT. *Proceedings of the IEEE/IFIP Network Operations and Management Symposium 2022: Network and Service Management in the Era of Cloudification, Softwarization and Artificial Intelligence, NOMS* 2022. <https://doi.org/10.1109/NOMS54207.2022.9789878>
- Lundberg, S. M., & Lee, S. I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems, 2017-December*, 4766–4775. <https://arxiv.org/pdf/1705.07874>
- Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2), 026113. <https://doi.org/10.1103/PhysRevE.69.026113>
- (PDF) *Topological Signatures of Cyber-Attack Classes in Natural Visibility Graph Representations of Network Traffic*. (n.d.). Retrieved October 1, 2026, from https://www.researchgate.net/publication/414686771_Topological_Signatures_of_Cyber-Attack_Classes_in_Natural_Visibility_Graph_Representations_of_Network_Traffic
- Protogerou, A., Papadopoulos, S., Drosou, A., Tzovaras, D., & Refanidis, I. (2021). A graph neural network method for distributed anomaly detection in IoT. *Evolving Systems*, 12(1), 19–36. <https://doi.org/10.1007/s12530-020-09347-0>
- Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of ‘small-world’ networks. *Nature* 1998 393:6684, 393(6684), 440–442. <https://doi.org/10.1038/30918>

- Yao, Y., Su, L., & Lu, Z. (2018). DeepGFL: Deep Feature Learning via Graph for Attack Detection on Flow-Based Network Traffic. *Proceedings - IEEE Military Communications Conference MILCOM, 2019-October*, 579–584. <https://doi.org/10.1109/MILCOM.2018.8599821>
- Yao, Y., Su, L., Lu, Z., & Liu, B. (2019). STDeepGraph: Spatial-temporal deep learning on communication graphs for long-term network attack detection. *Proceedings - 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE 2019*, 120–127. <https://doi.org/10.1109/TrustCom/BigDataSE.2019.00025>
- Zhao, J., Zhang, D., He, Q., Lin, M., & Yang, Y. (2026). DMSTG-AD: an SDN intrusion detection method based on dynamic multi-scale spatio-temporal graph neural network. *Scientific Reports 2026 16:1*, 16(1), 14528-. <https://doi.org/10.1038/s41598-026-44360-4>
- Zhao, Z., Wang, J., & Wang, C. (2013). An unknown malware detection scheme based on the features of graph. *Security and Communication Networks*, 6(2), 239–246. <https://doi.org/10.1002/SEC.524;SUBPAGE:STRING:FULL>
- Zola, F., Segurolo-Gil, L., Bruse, J. L., Galar, M., & Orduna-Urrutia, R. (2022). Network traffic analysis through node behaviour classification: a graph-based approach with temporal dissection and data-level preprocessing. *Computers and Security*, 115. <https://doi.org/10.1016/j.cose.2022.102632>

CHAPTER 12

ADVERSARIAL ARTIFICIAL INTELLIGENCE AND CROSS SECTOR CYBERSECURITY GOVERNANCE

ÖMER SEVİNÇ¹

Abstract Education, healthcare, and government use AI for different missions, yet their AI systems face common technical attacks and governance failures. This chapter develops a cross-sector model for adversarial machine learning and trustworthy cybersecurity. It distinguishes attacks during data collection, model development, deployment, and operation; examines poisoning, evasion, extraction, privacy leakage, prompt injection, and agent misuse; and maps controls to the AI lifecycle. A comparative analysis shows why control intensity must vary with consequences: an incorrect tutoring response, a corrupted clinical recommendation, and a distorted public-benefit decision do not produce equivalent harm. The proposed governance model combines risk classification, secure development, independent evaluation, continuous monitoring, incident disclosure, and meaningful human control.

¹ Dr., Ondokuz Mayıs University, Vezirköprü Vocational School, Department of Computer Programming.

Keywords adversarial machine learning, AI governance, cybersecurity, model security, prompt injection, risk management

Introduction

Organizations increasingly use AI inside security operations and expose AI capabilities to students, clinicians, civil servants, and citizens. This creates a coupled problem: AI can defend digital systems, but AI components must also be defended. Conventional cybersecurity protects accounts, devices, applications, and networks. AI security adds the behavior of data-driven models, whose performance can change when inputs, training data, context, or dependencies change.

A cross-sector approach is useful because many technical attacks are shared, while acceptable risk is not. A chatbot that gives an incorrect study suggestion can usually be corrected. A clinical model that suppresses a warning may cause physical harm. A public system that incorrectly flags fraud may interrupt a legal entitlement. Governance must therefore combine reusable technical controls with sector-specific impact analysis (NIST, 2023; Vassilev et al., 2024).

Adversarial AI Across the Lifecycle

During data acquisition, an attacker may poison labels, insert backdoors, manipulate feedback, or compromise a data supplier. Provenance records, source authentication, statistical screening, and separation of duties reduce this risk. Data quality checks alone are insufficient because malicious samples may appear statistically ordinary while creating a targeted behavior.

During model development, risks include vulnerable packages, exposed credentials, unauthorized changes, and weak evaluation. Reproducible builds, signed artifacts, dependency inventories, isolated training environments, and controlled experiment records provide evidence about what was produced.

Red-team evaluation should test expected misuse and high-impact failure modes before release.

At deployment, attackers may craft evasion inputs, extract the model through repeated queries, infer sensitive training information, or exploit an application around the model. Rate limits, authentication, input validation, output filtering, privacy protections, and least-privilege tool access reduce exposure. Generative systems require special attention to prompt injection because instructions and untrusted content may share the same context.

During operation, model drift, supplier updates, and changing attacker behavior can invalidate earlier tests. Monitoring should cover both system security and model behavior. Organizations need thresholds for warning, restricting, rolling back, or disabling a model, and they must rehearse those actions before an incident.

Comparative Risk in Three Sectors

Education places particular weight on children's privacy, equal access, academic freedom, and due process. Monitoring designed to detect account compromise should not become an unrestricted behavioral profile. Automated restrictions need timely review because they may block learning or assessment.

Healthcare places greater weight on immediate safety and continuity. A containment action can itself create harm if it interrupts a therapeutic device or removes clinical information. Model security must be tested with clinical workflow, and humans need authority and time to evaluate consequential outputs.

Government carries duties of legality, equality, transparency, and continuity. An AI system may affect benefits, enforcement, or public communication. Security controls must preserve evidence for oversight and provide a route to challenge outcomes. Some technical

details may need protection, but secrecy cannot replace accountability.

Sector	Highest consequence	Required emphasis
Education	Loss of access or unfair discipline	Privacy, due process, and age-appropriate safeguards
Healthcare	Patient harm or unsafe delay	Clinical validation, continuity, and authorized containment
Government	Loss of rights or essential service	Legality, auditability, and public accountability

Integrated Governance Model

The first step is risk classification based on the decision supported, affected population, reversibility, data sensitivity, scale, and dependency of essential services. Classification determines testing depth, approval authority, monitoring, and incident reporting. A low-impact drafting assistant should not receive the same process as a diagnostic or eligibility system.

The second step is secure lifecycle management. Organizations should maintain an AI inventory, data lineage, dependency records, model and prompt versions, evaluation results, access policies, and named owners. Changes should move through controlled review. Independent evaluation is appropriate where developers have incentives or blind spots that could weaken assurance.

The third step is operational governance. Security and domain teams should share metrics, conduct scenario exercises, and define decision rights during incidents. Reports should separate model error, malicious manipulation, data failure, and infrastructure failure because each requires a different response. Lessons should update requirements, tests, and procurement terms.

Meaningful human control depends on competence and authority. A nominal reviewer who cannot understand uncertainty, access supporting evidence, or reverse an action does not provide

effective oversight. Training must therefore be role-specific and supported by interfaces that present evidence at the time of decision.

Taxonomy of Attacker Goals and Capabilities

Adversarial AI analysis should identify the attacker's objective, knowledge, access, and stage of intervention. Objectives may include causing any error, forcing a specific output, degrading service, extracting information, evading detection, or obtaining unauthorized tool execution. An attacker may know only that a model exists, may observe confidence scores, may query it repeatedly, or may have access to training data and parameters. These distinctions determine which tests and mitigations are relevant.

Capability also includes influence over surrounding systems. A user who cannot alter a model may still manipulate a document retrieved by it, compromise an API key, or exploit an overprivileged plug-in. Security assessment should therefore cover the complete AI system, including data pipelines, orchestration code, external tools, human review, monitoring, and update mechanisms. Model robustness alone cannot compensate for weak application security.

Testing Red Teaming and Independent Evaluation

Evaluation should begin with ordinary performance and extend to malicious conditions. Testers can introduce label errors, hidden triggers, distribution shifts, malformed inputs, prompt injections, excessive queries, and attempts to reveal restricted information. Results should be reported against predefined acceptance criteria, with separate measures for successful attacks, false alarms, performance degradation, and operational impact. A single benchmark score does not establish security across changing contexts.

Red teams should include domain specialists who understand realistic consequences. An education exercise may test exposure of examination material; a healthcare exercise may test corrupted clinical context; a government exercise may test unauthorized access to citizen records or public communication tools. Independent evaluation is valuable for high-impact systems because developers may know the intended behavior but overlook misuse paths or organizational incentives that weaken controls.

Security of Large Language Models and AI Agents

Large language models blur the boundary between data and instructions. Prompt injection exploits this property by placing adversarial instructions in user input, retrieved documents, web pages, or tool output. Current defenses can reduce but not eliminate the problem. Systems should treat external content as untrusted, constrain tools through explicit schemas and permissions, separate security decisions from free-form generation, and require confirmation for irreversible or high-impact actions.

AI agents magnify risk because they can plan sequences, call services, and act with delegated credentials. Each agent should have a defined purpose, limited environment, spending and action limits, complete audit trail, and a reliable stop mechanism. Credentials should be short-lived and scoped to the specific task. Simulated environments and staged deployment can reveal unsafe loops, unexpected tool combinations, and attempts to bypass approval before the agent reaches operational systems.

Assurance Evidence and Control Mapping

Organizations need an assurance case that connects risks to controls and evidence. The case may include data provenance, secure-development records, threat models, test results, access reviews, incident exercises, monitoring performance, and approval decisions. Evidence should be versioned because a model update,

new retrieval source, or infrastructure change may invalidate an earlier conclusion. Control mapping can align AI-specific safeguards with established cybersecurity, privacy, safety, and quality-management processes.

Residual risk should be stated explicitly. Adversarial training may improve resistance to tested inputs without covering unknown attacks. Output filters may reduce harmful disclosure while introducing false positives. Differential privacy may limit memorization while reducing utility. Decision-makers need these tradeoffs in concrete terms, along with conditions for restricting or ending use. Assurance is credible when it describes limitations and operational response, not when it presents a control as complete protection.

Incident Disclosure and Shared Learning

AI incidents can be difficult to classify because poor output may result from model error, malicious input, compromised data, software failure, or misuse. Incident processes should preserve the interaction, context, model and prompt versions, connected tools, identity events, and relevant system logs. Investigators should assess both immediate technical effects and downstream decisions made by people or automated services.

Responsible disclosure and sector information sharing can prevent repeated harm. Reports should describe the affected component, attack path, impact, containment, and durable corrective action without exposing personal information or providing unnecessary exploitation detail. Cross-sector learning is especially valuable for prompt injection, model supply-chain compromise, and privacy leakage because the same mechanisms recur in education, healthcare, and government even when the consequences differ.

Conclusion

Adversarial AI is not a single vulnerability that can be removed through one test. It is a lifecycle risk shaped by data, software, people, suppliers, and institutional purpose. Education, healthcare, and government can reuse core controls, but they should scale those controls to the harm that a failure or attack could create. Trustworthy deployment requires evidence that the system is secure enough for its specific use, can be monitored in practice, and can be stopped or recovered without abandoning the service it supports.

References

- European Union Agency for Cybersecurity. (2021). Securing machine learning algorithms. <https://www.enisa.europa.eu/publications/securing-machine-learning-algorithms>
- European Parliament and Council of the European Union. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence. Official Journal of the European Union.
- National Institute of Standards and Technology. (2023). Artificial Intelligence Risk Management Framework 1.0 (NIST AI 100-1). <https://doi.org/10.6028/NIST.AI.100-1>
- National Institute of Standards and Technology. (2024). Artificial Intelligence Risk Management Framework Generative Artificial Intelligence Profile (NIST AI 600-1). <https://doi.org/10.6028/NIST.AI.600-1>
- Organisation for Economic Co-operation and Development. (2025). Governing with artificial intelligence. <https://doi.org/10.1787/795de142-en>
- Vassilev, A., Oprea, A., Fordyce, A., & Anderson, H. (2024). Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. NIST. <https://doi.org/10.6028/NIST.AI.100-2e2023>
- World Health Organization. (2024). Ethics and governance of artificial intelligence for health: Guidance on large multi-modal models. <https://www.who.int/publications/i/item/9789240084759>

