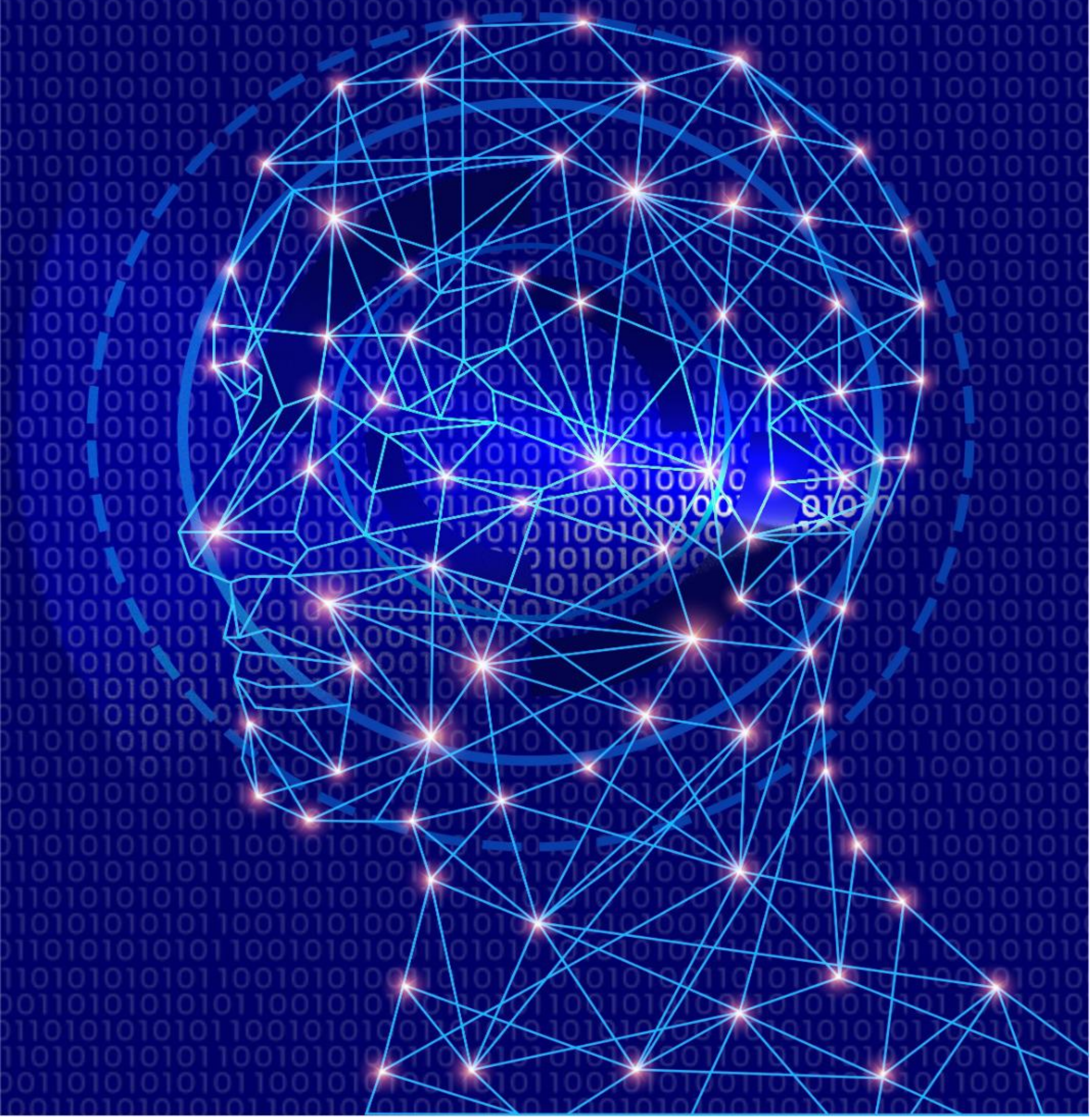
International Journal of

ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH

AAIR

Volume 4, Issue 2, 2024
ISSN: 2757-7422



International Journal of
ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH
AAIR

Advances in Artificial Intelligence Research

https://dergipark.org.tr/aair

Date of Publication	December 2024
Language	English
Frequency	Published twice in a year
Graphic designer	Özden Işıktaş
Correspondence Address	Muğla Sıtkı Koçman University, Faculty of Technology, Information Systems Engineering, 48000 Kötekli/MUĞLA
Phone	0 506 4887818 – 0252 211 5526
Correspondence Mail	osmanozkaraca@mu.edu.tr

Volume : 4 2024 Issue : 2

International Journal of
ADVANCES IN
ARTIFICIAL
INTELLIGENCE
RESEARCH
AAIR

Table of Contents

Volume : 4 2024 Issue : 2

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

Vol. 4 (No. 2), pp. 62-68, 2024
doi: 10.54569/aair.1582085
Published online: Dec 30, 2024

Advanced Machine Learning for Brain Tumor and Alzheimer's Disease Detection: A Comprehensive Review of Neuroimaging-based Classification Techniques

Vaibhav Narawade [1], Naman Kumar [1,*], Harisha Patkar [1], Kanish Chheda [1],
Aniket Mishra [1]

[1] Department of Computer Engineering, Ramrao Adik Institute of Technology, DY Patil Deemed To Be University,
Nerul, India;

Abstract

Alzheimer's disease with progressive neurodegeneration and brain tumors notably characterized by rapid, not limited cell proliferation poses significant health risks unless timely diagnosed and treated. Tumors have a diverse feature and characteristics, added to subtle changes in the brain whose hallmark is Alzheimer's, making accurate segmentation and classification quite challenging. Indeed, while there have been research in the last decade or so that have proven promising results, challenges still linger on. The present work discusses various approaches for image classification and staging of Alzheimer's disease and brain tumors by exploiting techniques in statistical image processing and computational intelligence. This paper includes discussion on morphology of brain tumors along with neuroimaging changes caused by Alzheimer's disease, existing datasets, data augmentation techniques, and methods for component extraction and classification within the DL, TL, and ML framework. Such specific systems have been given the metrics using the datasets; the descriptions of the implementations, however may vary with the case at hand.

Keywords: *Alzheimer Detection, Convolutional Neural Networks, Brain Tumor Detection*.

1. Introduction

AD and brain tumors are but two examples of the most common neurological diseases worldwide [1]. This disease might affect millions around the globe every year, yet in both conditions there is a problem relating to significant difficulties in diagnosis and treatment due to large, complexly interconnected networks of neurons (and other supporting cells). More than that, these diseases do not affect only the single patient but also affect their caregivers, families, and enormous healthcare systems. Lastly, a very high economic cost comes with brain tumors and Alzheimer's: direct medical costs, lost productivity, and long-term care. Although these issues are utterly overwhelming, plans to enhance knowledge and treatment better are underway.

Brain tumors are collections of abnormal cell growth in the brain, interfere with normal brain function, and can be lethal [2]. There are various forms of brain tumors including gliomas, meningiomas, and pituitary tumors, which vary in their characteristic features and require a specific mode of treatment to control the tumor. The brain tumors will be treated only if they are diagnosed early in the course of the disease, so that a basic treatment approach may be adopted that will allow the disease course in the patient to become enhanced. According to their location as well as their size, symptoms of brain tumors vary and may present manifestations ranging from headaches and seizures to alteration in personality and cognitive functions. It comes as no surprise that heterogeneity of symptoms leads to misdiagnosis, since they are initially attributed to other much more common conditions. A second critical role the blood-brain barrier plays is its role in the exclusion of neurotoxins from entry into the brain; on the other hand, it would also severely compromise access of therapeutic agents to tumors growing in the brain.

Alzheimer's is a neurodegenerative disease primarily linked to old age. The two hallmarks of Alzheimer's include the tau tangles and beta-amyloid plaques in the brain that progressively lead to degeneration of cognitive functions, loss of memory, and ultimately dependence [3]. The course of the illness is characterized by three progressive stages one after the other: EMCI followed by LMCI then full-blown AD [4]. Though making a diagnosis may be challenging at the early stages of the disease, intervention still appears to be warranted insofar as it may retard the process of this disease and likely enhance the quality of life for the patient as well as his or her caregiver. Researchers are trying to disentangle very complicated relationships among a wide variety of genetic, environmental, and lifestyle factors that may influence the bases of this disease. Current studies are also investigating different contributions of inflammation, vascular health, and the gut microbiome towards the initiation and development of AD. Because AD research requires multifaceted studies for diagnoses and treatments.

*Corresponding author
E-mail address: aniketmishra6669@gmail.com

Received: 9/Nov/2024; Accepted: 28/Dec/2024.

Advances in medical imaging technologies and, particularly in MRI have nearly revolutionized the diagnosis and monitoring capabilities for brain tumors and Alzheimer's disease. It provides very anatomical insight into the brain; hence, helps identify tumors, patterns of atrophy, and other abnormalities related to these conditions [5]. However, such images require much experience and time-consuming processing that would result in some delay in diagnosis and treatment. Besides structural MRI, functional MRI and diffusion tensor imaging proved to be very effective techniques for a better study of connectivity and patterns of functionality in the brain. These modalities shed further new light on the impact that tumors and neurodegenerative diseases have on the networks of the brain. In this respect, PET imaging is also important when it contains tracers that selectively target tau and amyloid proteins specifically; these became very relevant for diagnosis and study of Alzheimer's disease and also for in vivo visualization of pathological protein aggregates.

Convolutional neural networks are the signature, most recently, which gained extreme popularity as an extremely powerful tool in this area, with the additional benefit of very high precision for tasks of image classification. Deep learning models automatically extract features from medical images and identify complex patterns that could not be easily detected by the human eye. CNNs one of the good application processing huge amount of imaging data due to its power parallel processing, and speed reliability.

Researchers used the AI and machine learning technique, that is, deep learning in computer-based systems, and came up with devising them to classify brain tumors and Alzheimer's disease [6]. Further, training on other datasets could be done to include subtle differences within the presentation of disease among populations in an effort to increase the generalization of the algorithms toward diagnosis.

In many ways, one develops deep models for classification of neurological disorders. Among all the above strategies, two have been considerably considered as the most effective ones: those are multi-model approaches that aggregate different architectures towards acquiring a wide range of features and transfer learning that fine-tunes the pre-trained models towards an image-based medical task.

Recently, work has also been done in feature extraction technique with PCA and GLCM techniques in enhancement of the discriminative potential of such models. Thus, integration of structural MRI with PET or DTI looks quite promising in enhancing diagnostic accuracy and comprehensive evaluation of the health of the brain. Besides, other research work is being carried out to combine non-imaging data while developing high accuracy and personalized diagnostic tools within models of AI, such as genetic details, cognitive tests' results, and clinical history.

Classification for brain tumor and Alzheimer's disease has also been pretty promising with deep learning models. Many challenging areas are still there that require a solution. It runs the gamut from explainable AI systems where it would be possible for doctors to understand the reasoning behind their decisions, working through natural clinical workflows with these technologies to data sets with higher diversity in order to get increased model generalization [11]. In this domain, interpretability of model is of great value. This domain will not only give point why the patient has been diagnosed with some particular ailment but also show the importance of those features-a computer itself can be equally important or even more important than the diagnosis. Techniques such as relevance propagation layer-wise and attention mapping are used to "highlight to the researcher what features and regions of the brain an AI model thinks will be most important for classification." Thus, robust validation methodologies, performance indicators therefore have to be developed, so that the results can be reliable and comparable among different AI approaches within research and institutions.

2. Literature Review

Recently, deep learning has revolutionized medical imaging and especially in the detection and classification of complex conditions like Alzheimer's disease and brain tumor. Thus, for example, Santos Bringas et al. used data from accelerometers of smartphone devices for designing a deep learning architecture, employing CNNs, to enable distinguishing among various stages of Alzheimer's disease [12]. The network demonstrates that CNNs can be notably superior to other mainstream classifiers like SVM and Decision Trees when average hits 90.91%. Several limitations about generalisability have been pointed out for the model since it is only developed with the database of 35 subjects participating in the study. In addition, the inability of the model to be explained in its decision process also gives way to the need for developing more transparent AI methods that underpin clinical utility [12]. Similarly, (Santos and Santos 2024) applied the lightweight CNN model MobileNetV2 architecture in the identification process of the presence of brain tumors using MRI images [13]. This was achieved after training the model on a dataset of 3,762 MRI images with an accuracy of 89%, thus showing that MobileNetV2 is indeed effective for tasks involving medical imaging. The research in this subject has a limitation tied to the problems associated with the minimal number of samples used in the dataset, which may limit the strength of the model and its generalizability. In future work, the size of the dataset should be enhanced using more diverse deep architectures to build up further the performance as well as the reliability of the model [13].

Nayak et al. in 2024 later proposed the dense EfficientNet architecture for the classification of the subtypes of brain cancers, such as pituitary, meningioma, and glioma [14]. The training dataset consisted of 3,260 artificially augmented T1-weighted MRI images using advance data augmentation techniques. Although such a dense EfficientNet model would be much more difficult to use in real-time clinical environments, it reaches great values of accuracy-on the training set at 99.97% and on the testing set at 98.78%. In short, it is emphasized that model simplification is relevant for the reduction of overhead computations while maintaining high-accuracy results. In addition, since the size of the dataset was not large, further research studies should focus on increasing the size of the dataset and testing the model in further clinical environments to confirm its generalization capability [14]. In the same direction, Nassar et al. (2024) proposed a hybrid deep learning approach that utilizes the power of different CNN architectures by adopting a majority vote method [15]. The above system obtained an accuracy of 99. 31% in terms of classification. This outperformed the performances obtained by other separate CNN-based classifiers. The authors highly recommend a larger and diverse dataset for the improvement of the model in terms of robustness, along with adding other deep learning techniques in order to exploit classification improvements further [15].

Helaly et al. (2024) utilized MRI images of the ADNI dataset to classify the stages of Alzheimer's disease by using the most minimalistic CNN architectures and transfer learning on a pre-trained VGG19 model [16]. Optimized for the task, the VGG19 model proved with accuracy the prospects of transfer learning in medical imaging with an accuracy rate of 97%. Although these results are promising, the study sheds light on pretty much a thin dataset applied and encourages more advanced data augmentation techniques which may get the robustness of the model high [16]. Also, Odusami et al. (2024) proposed an explainable deep learning model to facilitate the diagnosis of Alzheimer's disease, which is developed by fusing multimodal input into PET and MRI images. At the same time, with structural adjustments to the architecture of ResNet18 in order to include the merged data, the model correctly classified it to be either EMCI or LMCI with 73. 9% accuracy. Along with several applications of explainable AI methods, excellent interpretability of the model was achieved-the all-important requirement for clinical applications. Despite all this, it identified a host of challenges related to the complexity of the model and thus recommended future work relate to the capabilities of XAI and refining the model for real-time applications [17].

Besides that, Küstner et al. (2024) presented the development of AI application for MRI and MRS [18]. In this research, they used deep learning in discussing the phases of MRI which includes planning, acquisition, and reconstruction. The study showed that applying the above techniques, the diagnosis and reconstruction of MRI and MRS can be extended without the requirement of fully labeled data. Besides that, new architectures for neural networks must be further researched to strengthen trust and usability of models for various clinical settings. Problems related to instability of the model, hallucinations, and shifts in the domain have indeed occurred within clinical practice of natural language models [18]. The last, Sharif et al. (2024) proposed a decision support system which utilised advanced feature selection techniques and an enhanced version of model densenet201 for the classification of multimodal brain tumors [19]. Their model reported success in the strategy formulated by their model by achieving above 95% accuracy upon testing on datasets of multimodal MRI: BRATS2018 and BRATS2019. It was based on this consideration that this study identified that high-dimensional feature spaces present challenges to the effectiveness and generalizability of the model and therefore recommended further research to develop feature selection techniques and extend the method to other medical imaging scenarios [19].

2.1. Deep Learning Techniques

Deep learning algorithms have also dramatically revolutionized the process of medical image processing, including review brain MRIs to classify different neurological conditions. Among all architectures, CNNs have emerged as the most popular due to widely reported success in many applications, including notably image segmentation, tumor detection, and disease classification. According to Xie et al., CNNs have outperformed other approaches in machine learning based on efficiency and accuracy for the detection of MRI brain imagery using even limited datasets [5]. Indeed, several recognitions have been presented based on the success of CNN in attaining a capability to acquire, automatically from images, features without human involvement and bypass complications involved with feature engineering techniques. It demonstrated its effectiveness on the more modest datasets typical to medical imaging when applied.

Singh et al. transferred pre-trained models, such as VGG16 and InceptionV3, to win in the task of pneumonia detection from chest X-rays [6]. The proposed approach was successfully applied for analysis of brain MRIs without failure. Mehmood et al. [4] employed transfer learning employing a modified version of VGG-19 architecture to achieve high classification accuracy in the possibility of several stages of cognitive impairment and ease of detection at an early Alzheimer's disease. With these findings, the authors concluded that freezing down some of the layers of the pre-trained network while fine-tuning others brings enormous improvement in the model combined with the data augmentation techniques. With the newest techniques of multitask learning, it has been promising to solve a suite of classification problems all at once, which subsequently can make the model more effective and performative. Liu et al. proposed a multitask deep learning architecture with 3D DenseNet for feature extraction and a multitask CNN for hippocampus segmentation and disease classification [11]. This is one example of how a large set of related tasks might be applicable toward improvement of the model, such as high classification accuracy in the case of Alzheimer's. 3D CNNs would capture volumetric MRI information much better in terms of spatial information than their 2D counterparts and may lead eventually to more precise diagnoses.

Moreover, other techniques for concatenation of features and ensemble methods for improving the classification accuracy have been explored as well. Noreen et al. achieved 99. 34% and 99. 51% accuracy for a system to identify brain tumors with features from various levels of Inception-v3 and DenseNet201 models, respectively, by using a concatenation-based technique [8]. The method further enhances the classification model by utilizing various levels of feature extraction as well as different architectures of the network. Similarly, Irmak developed CNN models based on automatically learned hyperparameters to assist in multi-classification of the brain tumors [2]. The classifying accuracy for tumor grading was 98. 14%, tumor detection results showed 99. 33% accuracy, and the type of the tumor with an accuracy of 92. 66%. An optimization technique via grid search for the adjustment of the hyperparameter reveals how an automatic model optimization actually enhances the accuracy of the classification.

2.2. Machine Learning Methods

1. Preparatory stage: Applying preprocessing techniques, such as picture normalization and data augmentation in the case of usability, and noise reduction in the case of high-quality data is of great importance for medical image analysis using machine learning. Among the most applied techniques for improving the performance of the next stages and improving the quality of the image, the following are utilized: Gaussian filtering and histogram equalization. These preprocessing techniques help in normalizing input data, reduce heterogeneity, and therefore, make machine-learning models stronger for MRI brain imaging [1, 3].

2. Segmentation: This is the important part of the process of medical image analysis, where segmentation focuses on splitting up an image into sections which need more inspection. Many machine learning techniques have been used for segmentation of medical imaging challenges, especially CNNs and U-Nets models. These models are trained to identify and discriminate between objects like tumors or organs from an MRI picture. CNNs show great promise in accurately detecting brain tumors from MRI images, which is crucial for both diagnosis and therapy planning [2, 7].

3. Feature Extraction: This stage is where the prominent features or patterns that may be present in the image can be extracted and may later be used to classify images. Inasmuch as deep learning models, such as CNNs, directly identify and extract the characteristics from the raw picture data, this process is usually taken over for automating this stage of medical image analysis. But with these advanced models other approaches like PCA and Gabor filters are developed further in order to achieve better feature extraction techniques. Studies on brain MRI which compare the features of the extracted tissue, like texture, shape, and intensity are compared against healthy and ill tissue, these techniques have been proven [4, 6].

4. Classification: In the last stage of classification, the features that are extracted from an image get classified into more than one class. For example, brain tumor classification and Alzheimer's disease diagnosis also falls in the category of classifications. SVMs and DNNs are two of the most famous machine learning models with regard to this. As an example, such a high degree of specificity in subclassification of brain tumors into such categories as pituitary tumors, gliomas, and meningiomas can be nothing but admired. Using the approach of transfer learning algorithms, pre-trained models may be re-used for specific domains in medical images to enhance the classification performance [4, 8].

3. Results and Findings

These researches collectively show how profoundly deep learning models may aid in improving the precision and efficiency of medical images, like classifying brain tumors and detecting Alzheimer's. This can apparently be achieved with Convolutional Neural Networks when processing mobility data from an accelerometer in classifying the stages of Alzheimer's disease with a mean accuracy of 90.91% as supplied by Santos Bringas et al. (2024) [12]. This translates to complex patterns regarding neurological conditions subjected to complex processing and analysis by deep learning models. Santos and Santos, in 2024, capitalized on this need for MRI images and the MobileNetV2 architecture to successfully realize the detection of brain tumors with an accuracy rate of 89% [13]. From their study, they indicate that lightweight CNNs are feasible in clinical applications, particularly for cases where processing capabilities are limited [13]. Meanwhile, Nayak et al. (2024) said it was actually the Dense EfficientNet model that was employed in the classification of brain tumors and achieved an amazing accuracy of 99.97% by using the training set and 98.78% by using the testing set [14]. This research work also underscores that not only complex neural network architectures can result in achieving near-perfect results in classification but, most importantly, data augmentation techniques also play a significant role in achieving better performance of the model [14]. Likewise, the hybrid deep learning approach adopted by Nassar et al., (2024) was also able to attain high classification accuracy of 99.31% and proved that multi-CNN architectures combining using majority voting techniques improve the reliability of the diagnoses [15].

Apart from presenting some important conclusions with respect to issues and future tracks for deep learning in medical imaging, some of the reviews presented high accuracy levels. Helaly et al. (2024) proved that transfer learning is applicable in medicine by reporting a 97% accuracy classification concerning the stages of Alzheimer's disease using an optimized version of the VGG19 model [16]. As in other research studies, they further also pointed out the limitation of the small data-sets used that could influence the ability of deep learning-based models to generalize well. In the diagnosis of AD, Odusami et al. (2024) suggested using XAI methods within a tailored ResNet18 architecture to classify between EMCI and LMCI achieved 73. 9% classification accuracy [17]. Their study gives testimony to the interpretability of AI models; in a medical setting as much lies behind the right diagnosis as does the reason behind the result obtained. Küstner et al.'s work with AI for MRI and MRS in 2024 proves that indeed deep breakthrough had been achieved in the reconstruction of imaging and in the actual diagnosis [18]. Model instability was also mentioned, and indeed the need for more than just a validation in limits was discussed [18]. Lastly, Shamir et al. developed DSS reached an accuracy over 95%. (2024) Applied the state-of-the-art densenet201 model along with innovative feature selection methods, and is the most concrete demonstration of ability of deep learning to enhance algorithms of clinical decision-making [19]. Table 1 shows the methodology and accuracy of all the deep learning and CNN models along with the datasets on which they have been trained on.

Table 1. *Analysis of Deep Learning Models).*

Paper Title	Methodology	Algorithm	Accuracy	Dataset Used
Image Classification of MRI Brain Image based on Deep Learning	Review of traditional and deep learning methods for MRI image classification	CNN, DNN, Transfer Learning, SVM	CNN: 96.97%, Transfer Learning: 92.8%	Various MRI datasets for Alzheimer’s and brain tumors
Identification of Brain Diseases Using Image Classification: A Deep Learning Approach	Deep learning techniques for classifying brain diseases from MRI images	CNN	Not specified	MRI scans for Alzheimer's, glioma, meningioma, pituitary tumor
Medical Image Classification for Disease Diagnosis based on Deep Convolutional Neural Network	Multiple methods for pneumonia detection from X-rays	SVM, Transfer Learning (VGG16, InceptionV3), CapsNet	VGG16: 94.7%, CapsNet: 93.6%	Chest X-ray images, 5,232 training images and 624 testing images
CNN and SVM-Based Brain Tumor Image Classification Performance Analysis	CNN and SVM models for brain tumor classification	CNN, SVM	CNN: 98.85% (Pituitary)	3,064 MRI images (Meningioma, Glioma, Pituitary)
Using a Neural Network Model Enhanced with PCA and SWLDA, improving Alzheimer’s Disease Classification in Brain MRI images	PCA and SWLDA combined with ANN for AD classification	PCA, SWLDA, ANN	99.35% (Weighted Avg.)	Not specified
Convolutional Neural Networks for Classification of Alzheimer's Disease:	Systematic review and CNN evaluation for AD classification	CNN	Similar to SVM: 76%-89%	Data from ADNI, AIBL, OASIS datasets
Method of Transfer Learning for AI-Powered Brain Tumor Classification	Transfer Learning using pre-trained CNN models	AlexNet, GoogLeNet, ResNet18, ResNet50, VGG16	99.04%	696 T1-weighted MRI images
A practical Deep Learning-Based Brain Imaging Classifier for Alzheimer's Disease on 85,721 Samples	Transfer Learning and CNN for AD classification	Inception-Resnet-V2	94.5% (AIBL dataset)	85,721 MRI scans from 50,876 participants
Medical Image Analysis Using Machine Learning and Deep Learning: Diagnosis to Detection	Literature review and experimental comparison of ML and DL models	PCA, LDA, CNN, SVM	CNN outperformed ML models	MRI datasets for various medical conditions
A Review of Brain MRI Image Classification Methods for Neurological Disorders	Review of classification techniques for neurological disorders	KNN, SVM, CNN, Decision Trees, Neural Networks	CNN: up to 98%	MRI images, including 6 normal and 4 abnormal brain images

A Multi-Model Deep Convolutional Neural Network for Automatic Hippocampus Segmentation and Classification in Alzheimer's Disease	Multi-task learning combining hippocampus segmentation and disease classification	3D DenseNet, Multi-task CNN	AD vs. NC: 88.9%, MCI vs. NC: 76.2%	Baseline T1-weighted MRI from ADNI (97 AD, 233 MCI, 119 NC subjects)
Classification of Autoimmune Disease and Brain Tumors via Ensemble Learning	Using ensemble learning to categorize autoimmune disorders and brain tumors	SVM, Majority Voting	Accuracy: 98.719%, Sensitivity: 97.5%	2,399 MRI images (Glioma, Meningioma, Pituitary Adenoma, Multiple Sclerosis)
MRI Brain Image Classification and Abnormality Detection Using Convolutional Neural Networks	CNN classification using feature extraction in Curvelet domain	AlexNet (25 layers), K-Means (segmentation)	Not specified	Public MRI brain image databases
Multi-Classification of Brain Tumor MRI Images via a Fully Optimized Deep	CNN with grid search optimization for multi-class brain tumor classification	Custom CNN models	Tumor detection: 99.33%, Multi-class: 92.66%, Tumor	RIDER, REMBRANDT, TCGA-LGG, Cheng datasets

3.1. DL Models

Some of the state-of-the-art front models of Deep Learning applied to medical image analysis include Convolutional Neural Networks, especially for automated feature extraction besides outstanding accuracy. Among some of the state-of-the-art architectures for CNN are ResNet, VGG, and Inception models that were achieved for outstanding results besides being very deep indeed for brain tumor identification. As such models are trained on larger datasets, they could be capable of dealing with complex patterns found in medical images. In addition, various studies have recently noted the utilization of transfer learning-that is, fine-tuning of pre-trained models on large datasets about specific medical image datasets to optimize efficiency and curtail consumption of processing power [4, 8].

3.2. Parameters

These three hyperparameters decide the efficiency of the deep learning model in detecting tumors in the brain include, Layer Count, Learning Rate, and Batch Size. These three hyperparameters are highly sensitive in terms of changing the performance of the model by changing them. Dropout is also often used to avoid overfitting and in addition to enhancing the robustness of the model. Some examples include layers that may most likely impact the capacity of the model to learn complex features, while the right choice of learning rates will impact not only the precision but also the convergence time of the model [10].

3.3. Drawback of ML Over DL Approaches

ML methods for medical image analysis are useful but have many disadvantages compared to deep learning methods. In ML features are always manually extracted. The method is time-consuming and prone to human error. Perhaps this very approach does influence the general performance of the model, which considers suboptimal feature presentation. Second, DL models outperform the ML models, especially when they need to handle high-dimensional data such as images of the medical picture as it allows for the direct extraction of hierarchical features directly from raw data. This is why DL models are generally superior to ML models in tasks, such as identification and classification of tumors, hence excellent for generalization and accuracy [2, 5].

3. Future Research Directions

As presented below are some of the salient points regarding which future research in deep learning models for brain tumor detection is likely to focus. Such precision in diagnosis and classification calls for integration of data from various imaging modalities such as CT, MRI, and PET scans. Another area of interest would be the development of more robust and interpretable AI models, which in turn leads further to the enhancement of capabilities of deep learning models in decision making and greater reliance on the model in clinical applications. Further, two relevant research areas include applications of DL in real-time diagnostics and federated learning, during which the model gets trained across decentralized sources of data. Accordingly, thanks to new features in this sphere, patients may note the better performance of such cases by using deep learning models within clinical practice [6, 11].

References

[1] Yadav, S. S., & Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis. Journal of Big Data, 6(1), 1-18.

[2] Irmak, E. (2021). Multi-Classification of Brain Tumor MRI Images Using Deep Convolutional Neural Network with Fully Optimized Framework. Electronics, 10(2), 184.

[3] Wen, J., Thibeau-Sutre, E., Diaz-Melo, M., Samper-Gonzalez, J., Routier, A., Bottani, S., ... & Colliot, O. (2020). Convolutional neural networks for classification of Alzheimer's disease: Overview and reproducible evaluation. Medical Image Analysis, 63, 101694.

[4] Mehmood, A., Yang, S., Feng, Z., Wang, M., Ahmad, A. S., Khan, R., ... & Yaqub, M. (2021). A transfer learning approach for early diagnosis of Alzheimer's disease on MRI images. Neuroscience, 460, 43-52.

[5] Xie, X. (2021). Deep learning-based image classification of MRI brain image. College of Medicine and Biological Information Engineering, Northeastern University, Shenyang, China.

[6] Singh, J., Singh, A., Singh, K. K., Lal, B., William, R. A., Turukmane, A. V., & Kumar, A. (2021). Identification of Brain Diseases using Image Classification: A Deep Learning Approach.

[7] Woźniak, M., Siłka, J., & Wieczorek, M. (2021). Deep neural network correlation learning mechanism for CT brain tumor detection. Neural Computing and Applications, 33(4), 1143-1155.

[8] Noreen, N., Palaniappan, S., Qayyum, A., Ahmad, I., Imran, M., & Shoaib, M. (2020). A deep learning model based on a concatenation approach for the diagnosis of brain tumor. IEEE Access, 8, 55135-55144.

[9] Mehrotra, R., Ansari, M. A., Agrawal, R., & Anand, R. S. (2020). A transfer learning approach for AI-based classification of brain tumors. IEEE Access, 8, 41667-41676.

[10] Ahmad, I., Siddiqi, M. H., Alhujaili, S. F., & Alrowaili, Z. A. (2022). Improving Alzheimer's disease classification in brain MRI images using a neural network model enhanced with PCA and SWLDA. Biomedical Signal Processing and Control, 71, 103186.

[11] Lu, B., Li, H. X., Chang, Z. K., Li, L., Chen, N. X., Zhu, Z. C., ... & Yan, C. G. (2023). A practical Alzheimer's disease classifier via brain imaging-based deep learning on 85,721 samples. Medical Image Analysis, 83, 102645.

[12] Santos Bringas, S., Salomón, R., Duque, C., Lage, J. L., Montaña, J. L. (2024). Alzheimer's Disease Stage Identification Using Deep Learning Models. Fundación Centro Tecnológico de Componentes CTC, Universidad de Cantabria.

[13] Santos, D. F. Santos, E. (2024). Brain Tumor Detection Using Deep Learning. BRIDGE – Instituto de Tecnologia e Pesquisa, Faculdade Estácio, Brown University.

[14] Nayak, D. R., Padhy, N., Mallick, P. K., Zymbler, M., Kumar, S. (2024). Brain Tumor Classification Using Dense EfficientNet. School of Engineering and Technology, GIET University, India; Kalinga Institute of Technology, India; South Ural State University, Russia.

[15] Nassar, S. E., Yasser, I., Amer, H. M., Mohamed, M. A. (2024). A Robust MRI-Based Brain Tumor Classification via a Hybrid Deep Learning Technique. Electronics and Communication Engineering Department, Mansoura University, Egypt.

[16] Helaly, H. A., Badawy, M., Haikal, A. Y. (2024). Deep Learning Approach for Early Detection of Alzheimer's Disease. Electrical Engineering Department, Damietta University, Egypt; Computers and Control Systems Engineering Department, Mansoura University, Egypt; Department of Computer Science and Informatics, Taibah University, Saudi Arabia.

[17] Odusami, M., Maskeliūnas, R., Damaševičius, R., Misra, S. (2024). Explainable Deep-Learning-Based Diagnosis of Alzheimer's Disease Using Multimodal Input Fusion of PET and MRI Images. Kaunas University of Technology, Lithuania; Silesian University of Technology, Poland; Institute of Energy Technology, Norway.

[18] Küstner, T., Qin, C., Sun, C., Ning, L., Scannell, C. M. (2024). The Intelligent Imaging Revolution: AI in MRI and MRS Acquisition and Reconstruction. University Hospital of Tuebingen, Imperial College London, University of Missouri-Columbia, Brigham and Women's Hospital, Eindhoven University of Technology.

[19] Sharif, M. I., Khan, M. A., Alhussein, M., Aurangzeb, K., Raza, M. (2024). A Decision Support System for Multimodal Brain Tumor Classification Using Deep Learning. Department of Computer Science, COMSATS University Islamabad, Pakistan; Department of Computer Science, HITEC University, Pakistan; Computer Engineering Department, King Saud University, Saudi Arabia.

[20] Baranwal, S. K., Jaiswal, K., Vaibhav, K., Kumar, A., Srikantaswamy, R. (2024). Performance Analysis of Brain Tumor Image Classification Using CNN and SVM. Department of Electronics and Communication Engineering, R.V. College of Engineering, Bangalore, India.

[21] Rana, M., Bhushan, M. (2024). Machine Learning and Deep Learning Approach for Medical Image Analysis: Diagnosis to Detection. Department of Computer Science, LNM Institute of Information Technology, Jaipur, India.

[22] Krishnammal, P. M., Raja, S. S. (2024). Convolutional Neural Network-Based Image Classification and Detection of Abnormalities in MRI Brain Images. Department of Computer Science, PSG College of Technology, Coimbatore, India.

[23] Tyagi, V. (2024). A Review on Image Classification Techniques to Classify Neurological Disorders of Brain MRI. Department of Computer Science, University of Delhi, India.

[24] Alzakri, P. J., Koller, M., Thuet, P., Leu, S., Diebo, T., Schwab, F., Lafage, V. (2024). Risk Factors for Proximal Junctional Kyphosis and Proximal Junctional Failure After Spinal Deformity Surgery: A Systematic Review. Department of Orthopedic Surgery, University Hospital of Geneva, Switzerland.

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

Vol. 4 (No. 2), pp. 69-77, 2024
doi: 10.54569/aair.1549781
Published online: Dec 30, 2024

A Comparative Study of Machine Learning Classifiers for Different Language Spam SMS Detection: Performance Evaluation and Analysis

Samrat Kumar Dev Sharma [1,*]

[1] Department of Statistics, Jagannath University, Bangladesh

Abstract

With the continuous rise in the number of mobile device users, SMS (Short Message Service) remains a prevalent communication tool accessible on both smartphones and basic phones. Consequently, SMS traffic has experienced a significant surge. This increase has also led to a rise in spam messages, as spammers seek financial or business gains through activities like marketing promotions, lottery scams, and credit card information theft. Consequently, spam classification has become a focal point of research. In this paper, we explore the effectiveness of 11 machine learning algorithms for SMS spam detection, including multinomial Naïve Bayes, K-Nearest Neighbors (KNN), and Random Forest, among others. Utilizing datasets from UCI and Bangla SMS collections, our experimental results reveal that the multinomial Naïve Bayes algorithm surpasses previous models in spam detection, achieving accuracies of 98.65% and 89.10% in the respective datasets.

Keywords: *Spam SMS Detection, NLP, Machine Learning, Deep Learning, Naïve Bayes.*

1. Introduction

In today's digital age, mobile phones have become an indispensable part of daily life, with over 5.4 billion people worldwide having at least one mobile subscription, as reported by GSMA. This proliferation of mobile devices has led to a staggering number of mobile subscriptions surpassing the global population for the past several years, reaching over 8.58 billion subscriptions, according to the International Telecommunication Union (ITU). However, this unprecedented connectivity also presents challenges, one of which is the pervasive issue of spam SMS (Short Message Service) messages. Spam SMS messages, often utilized by fraudsters, have proliferated alongside the increasing usage of mobile devices. These unsolicited messages, ranging from marketing promotions to lottery scams and credit card information theft, pose significant risks to recipients, resulting in financial losses and privacy breaches. The prevalence of spam SMS is evident in statistics, with 39.3% of recipients being female and 59.4% male. Furthermore, Americans received a staggering 78 [1]billion automated spam texts in just the first half of one recent year alone. Given the detrimental impact of spam SMS on individuals and the increasing frequency of such messages, there is a pressing need for effective detection and mitigation strategies. Machine learning offers promising solutions in this regard, leveraging algorithms to analyze message content and classify messages as spam or legitimate (ham). In this study, we conduct a comparative analysis of 11 machine learning classifiers for spam SMS detection. By evaluating the performance of these classifiers using datasets from various sources, including the UCI repository and Bangla SMS collections, we aim to provide insights into the effectiveness of different algorithms in combating spam SMS. Through our research, we seek to contribute to the development of robust and reliable spam detection techniques, ultimately empowering users to better protect themselves from the scourge of spam SMS messages.

2. Related Work

In recent decades, researchers have explored various approaches and techniques to address this challenge, with a focus on leveraging machine learning algorithms for efficient detection. Gupta et al. [2] of a comparative study of spam SMS detection using 8 different classifiers including Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), Logistic Regression (LG), Random Forest (RF), AdaBoost, ANN, and CNN. The test conducted [3] dataset that the authors show that the CNN and ANN have better accuracy compared to the other machine learning classifiers. The authors show that CNN is 98.25% and ANN is 98.00%, respectively. Xiaoxu Liu et al. [4] proposed 4 machine learning classifiers and deep learning transformers including LG, NB, RF, SVM, LSTM, CNN-LSTM, and spam Transformer of Spam SMS collection data set that the authors show logistic regression, Naïve Bayes, and SVM are better-performed machine learning classifiers. Here authors show accuracy is 98.56%, 98.38%, and 98.62%, respectively. Gadde et al. [5] using TF-IDF [6] word embedding technique and 6 classifiers algorithm. The authors proposed that the best algorithm is LSTM with an accuracy is 98.5%. Here authors also proposed RF and SVM whose accuracy achieved is 97.5% and 97% respectively on the SMS Spam Collection v.1 dataset. Then Suleiman et al. [7] using H2O framework to

*Corresponding author
E-mail address: samratkumardev2001@gmail.com

Received: 13/Sep/2024; Accepted: 28/Dec/2024.

achieve the highest accuracy of random forest algorithm. The authors claim accuracy is 97.7% respectively on the SMS Spam Collection v.1 dataset and using TF-IDF vectorizer algorithm to detect spam SMS and the authors get the accuracy for ham message is 99.46% and for spam accuracy is 95.90%. Haq et al. [8] proposed that logistic regression (LR) achieves a high accuracy of 99% respectively on the SMS Spam Collection v.1 dataset in detecting spam in mobile message communication, outperforming k-NN and decision tree (DT). Abayomi-Alli et al. [1] proposed that the highest accuracy is the Bi-LSTM model using SMS Spam Collection v.1 dataset that attained accuracy is 98.6% respectively. The authors also used SGD and Bayes Net models with accuracy, precision, recall, and F-measure of 96.8%, 96.9%, 91.7%, and 94.23%, respectively. Lim et al. [9] proposed the Cost-sensitive classifiers and Bayesian network to achieve the highest accuracy is 99.8\% respectively using the SMS Spam Collection v.1 dataset. Most researchers like [10, 11] and Himani Jain et al. [12] proposed the most efficient algorithms are Naïve Bayes and SVM. They are achieving accuracy of almost 97.93%, and 98.57% respectively. Tasmia et al. [13] proposed an ensemble approach to classify spam SMS Bengali text. Then Khan et al. [14] to proposed BERT Bangla SMS data set. and achieve accuracy 94% respectively. Lunna and Robert et al. [15, 16] proposed Bernoulli Naive Bayes model effectively detects and filters out spam in SMS texts with 96.63% accuracy, reducing security threats and fraud risks. Oyeyemi et al. [17] proposed the Naïve Bayes Alqahtani et al. [18] classifier + BERT model effectively detects and classifies SMS spam with a 97.31% accuracy and low false-positive rate, safeguarding users' privacy and assisting network providers. Yerima et al [6] proposed one class SVM classifier and Maqsood et al. [19] effectively detects and eliminates SMS spam with 98% overall accuracy and a low 3% false positive rate. Gupta et al. [20] using a voting classifier, a combination of four different classifiers including Gaussian NB, Bernoulli NB, Multinomial NB, and Decision Tree, provides more accurate spam detection than individual classifiers, with a mobile application to serve the purpose. The authors achieved 98.295% respectively of the SMS Spam Collection v.1 dataset.

3. Methodology

We started the workflow shown in **Figure 1**.at first data loading various sources, preprocessing and splitting data and applying the different classifier algorithms to train and test, and finally, evaluation and comparison to find the best models. Briefly discuss below

3.1. Dataset Description

In these experiments, we applied two different language datasets. The first dataset is the English SMS Spam Collection v.1 [3] dataset downloaded from the UCI repository. The dataset has 5572 rows and two columns. "v1" is the label (ham or spam) and "v2" is the messages.

The second dataset is Bangla SMS which was collected by personal survey. The researcher collected 504 data. This data set has two columns-target which label (ham or spam) and messages. The dataset is available as requested. It is shown **Table 1.**

Table 1. *Data Distribution between SMS Spam Collection v.1 and Bangla SMS*

	SMS Spam Collection v.1	Bangla SMS
Spam	747	217
Ham	4825	287
Total	5572	504

3.2. Data Preprocessing

The text-based data preprocessing pipeline involves several key steps. It begins with text cleaning to eliminate noise such as HTML tags, special characters, emoji, and punctuation. Tokenization then breaks down the text into smaller units, such as words or sentences. Next, stop words that do not significantly contribute to the text's semantics are removed. Finally, lemmatization is applied to reduce words to their base or root form, further streamlining the text data.

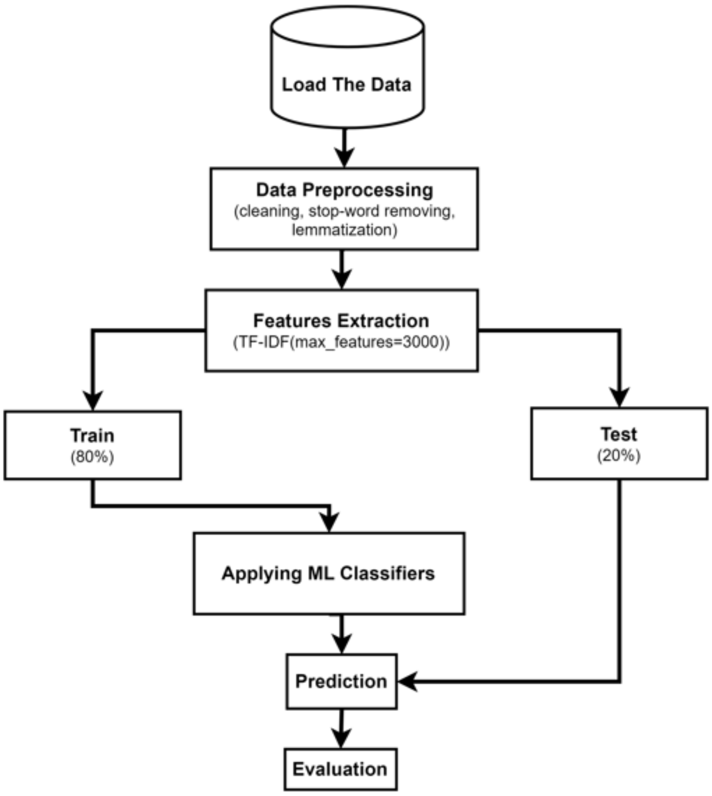


Figure 1. *Stepwise workflow*

After text data is preprocessed, it needs to be converted into a numerical format that machine learning algorithms can process. One common approach is to use the Term Frequency-Inverse Document Frequency (TF-IDF) technique, which assigns weights to words based on their frequency in the document and across the entire corpus. Each document is then represented as a vector, with each element corresponding to the TF-IDF score of a specific word. Transforming the text data into a TF-IDF matrix with a maximum of 3000 features allows us to capture the most relevant information while simultaneously reducing the dimensionality of the data. For the traditional machine learning approaches, the data is split into a training set (80%) and a test set (20%). This step is crucial for mitigating overfitting and handling imbalanced data. To ensure a representative distribution of classes in both the training and test sets, we employ a stratified splitting strategy. Stratified splitting preserves the relative proportions of different classes in both the training and test sets, thereby maintaining the balance between classes. By stratifying the data based on the target variable, we can effectively train the machine learning models on diverse samples while ensuring reliable evaluation of unseen data. This approach is particularly valuable when dealing with imbalanced datasets, where one class may be significantly more prevalent than others.

3.3. Machine Learning Classifiers

After completing text preprocessing, vectorization, and splitting, the data is ready for model classification. We apply 11 different classification models to the preprocessed and vectorized text data. These models include:

3.3.1. Naïve Bayes

Naïve Bayes is a simple probabilistic classifier based on Bayes' theorem with strong independence assumptions between features. The Multinomial Naïve Bayes (NB) classifier is a variant of the Naïve Bayes algorithm that is specifically designed for text classification tasks where the features are discrete and represent word counts or term frequencies.

$$\mathrm{P}(\mathrm{C_k}|\mathrm{doc}) = \frac{\mathrm{P}(\mathrm{C_k}) \times \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{P}(\mathrm{w_i}|\mathrm{C_k})}{\mathrm{P}(\mathrm{doc})} \tag{1}$$

Where:

- $P(C_k|doc)$ is the probability of classes C_k given the document.
- $P(C_k)$ is the prior probability of class C_k
- $P(w_i|C_k)$ is the probability of words w_i given class C_k
- $P(doc)$ is the probability of the document.

3.3.2. Logistic Regression

Logistic regression is a statistical method used for binary classification tasks. The logistic regression model applies a logistic function (also known as the sigmoid function) to linearly combine the features of the input data.

$$\mathrm{P}(\mathrm{Y}=1|\mathrm{X})=\frac{1}{1+\mathrm{e}^{-(\beta_0+\beta_1\mathrm{X}_1+\beta_2\mathrm{X}_2+\cdots+\beta_\mathrm{n}\mathrm{X}_\mathrm{n})}} \tag{2}$$

3.3.3. Support Vector Machine (*SVM*)

Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification tasks. Its main concept involves finding the hyperplane that best separates data points into different classes while maximizing the margin between the hyperplane and the closest data points (support vectors). This approach ensures robustness to noise and outliers in the data.

3.3.4. K-Nearest Neighbors (*KNN*)

The k-Nearest Neighbors (KNN) classifier is a simple, yet effective algorithm used for classification tasks. Its main concept involves assigning a class label to a data point based on the majority class among its K-nearest neighbors in the feature space. The choice of k determines the number of neighbors considered when making predictions.

3.3.5. Random Forest

The Random Forest classifier is an ensemble learning method that combines multiple decision trees to make predictions. It works by constructing a multitude of decision trees during training and outputting the mode of the classes (classification) or the mean prediction (regression) of the individual trees. Each tree in the forest is trained on a random subset of the training data, and during prediction, the results of all trees are aggregated to produce the final prediction. This approach helps reduce overfitting and improves the accuracy and robustness of the classifier.

3.3.6. Decision Tree

Decision tree classifiers are a type of supervised learning algorithm used for both classification and regression tasks. The main concept behind the decision trees is to recursively split the feature space into regions that are as pure as possible concerning the target variable. Each internal node of the tree represents a decision based on a feature, and each leaf node represents the predicted outcome. The decision tree is built by selecting the best feature to split the data at each node based on criteria such as Gini impurity or information gain. This process continues until a stopping criterion is met, such as reaching a maximum depth or a minimum number of samples per leaf. Decision trees are easy to interpret and visualize, making them popular for both exploration analysis and predictive modeling.

3.3.7. AdaBoost (Adaptive Boosting)

AdaBoost is an ensemble learning technique that combines multiple weak learners (often decision trees) to create a strong learner. It sequentially trains a series of weak learners, where each subsequent learner focuses more on the misclassified data points by giving them higher weights. The final prediction is made by aggregating the predictions of all weak learners, typically using a weighted majority voting scheme.

3.3.8. Bagging Classifier

Bagging (Bootstrap Aggregating) is an ensemble learning method that builds multiple base models (e.g., decision trees) on random subsets of the training data with replacement. Each base model is trained independently, and their predictions are combined using averaging (for regression) or voting (for classification) to make the final prediction.

3.3.9. ExtraTreesClassifier

Extra Trees (Extremely Randomized Trees) is an ensemble learning technique like Random Forest, where multiple decision trees are trained on random subsets of the training data. However, Extra Tree goes one step further by selecting random thresholds for each feature at each split point, resulting in even greater randomness and potentially reducing overfitting.

3.3.10. GradientBoostingClassifier

Gradient Boosting is an ensemble learning technique that builds multiple decision trees sequentially, with each tree aiming to correct the errors of its predecessor. In Gradient Boosting, each new tree is trained on the residual errors of the previous trees, gradually reducing the overall prediction error. The final prediction is obtained by summing the predictions of all trees.

3.3.11. XGBoost (Extreme Gradient Boosting)

XGBoost is an optimized implementation of Gradient Boosting that uses a more regularized model formalization to control overfitting and improve performance. It incorporates advanced features such as parallelized tree construction, hardware optimization, and efficient memory usage, making it one of the most popular and powerful gradient-boosting frameworks.

3.4. Model Training

All models were trained using the NVIDIA GeForce MX110 GPU, leveraging the Scikit-learn 1.4.2 library for machine learning tasks. This setup ensured efficient processing and utilization of computational resources during model training and evaluation.

4. Result and Analysis

4.1. Evaluation metrics

Evaluation metrics are used to assess the performance of the machine learning algorithm. It provides quantitative measures that help in comparing different models and selecting the most suitable one for a particular task. Common evaluation metrics are included as

Accuracy: It measures the proportion of correctly classified instances out of the total number of instances. It can be

$$\text{Accuracy} = \frac{\text{TP}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (3)$$

Where, TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative)

Precision: It measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It can be formulated as:

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

Recall: It measures the proportion of correctly predicted positive instances out of all actual positive instances. It can be formulated as:

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

F1-Score: It is the harmonic means of precision and recall, providing a balance between the two metrics. It can be formulated as:

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (6)$$

4.1. Experiment Results

We applied 11 different machine learning classifiers to two distinct SMS spam datasets. Our analysis revealed that the Multinomial Naïve Bayes classifier emerged as the top-performing model across both datasets, considering evaluation metrics such as accuracy, precision, recall, and f1-score, along with computational efficiency. We further improved its performance through hyperparameter tuning, as presented in **Table 2**. The experiment results for the SMS Spam Collection data are summarized in Table 3, while Table 4 presents the results for the Bangla SMS dataset.

Table 2. *Hyperparameters*

Model	Hyperparameters
SVC	Kernel: sigmoid, Gamma: 1.0
KNN	Default hyperparameters
Multinominal NB	Alpha: 0.01
DT	Max depth: 5
LG	Solver: liblinear, Penalty: l1
RF	Estimators: 50, Random state: 2
AdaBoost	Estimators: 50, Random state: 2
BG	Estimators: 50, Random state: 2
ET	Estimators: 50, Random state: 2
BG	Estimators: 50, Random state: 2
XGB	Estimators: 50, Random state: 2

Table 3. *Results of SMS Spam Collection v.1 dataset*

Algorithm	Accuracy %	Precision	Recall	F1-Score
MNB	98.65	0.9680	0.9236	0.9453
SVC	98.06	0.9826	0.8625	0.9186
RF	97.67	0.9652	0.8473	0.9024
BG	97.38	0.9482	0.8396	0.8906
AdaBoost	97.19	0.9047	0.8702	0.8871
ET	96.90	0.9541	0.7938	0.8666
XgBoost	96.61	0.8870	0.8396	0.8627
GB	95.45	0.9468	0.6793	0.7911
LG	95.16	0.9764	0.6335	0.7685
DT	92.94	0.8222	0.5648	0.6696
KNN	90.42	1.0000	0.2442	0.3926

Table 4. *Results of Bangla SMS dataset*

Algorithm	Accuracy%	Precision	Recall	F1-Score
MNB	89.10	0.8809	0.8604	0.8705
SVC	85.14	0.8500	0.7906	0.8192
RF	85.14	0.8333	0.8139	0.8235
BG	83.16	0.8250	0.7674	0.7951
AdaBoost	82.17	0.8048	0.7674	0.7857
ET	82.17	0.8205	0.7441	0.7804
XgBoost	82.17	0.8048	0.7674	0.7857
GB	79.21	0.7894	0.6976	0.7407
LG	78.21	0.7837	0.6744	0.7250
DT	70.29	0.7241	0.4883	0.5833
KNN	65.34	0.8333	0.2325	0.3636

4.3. Result Analysis

After analyzing the results, we propose selecting classifiers based on multiple factors including performance metrics and execution time. Models with higher accuracy, precision, recall, and F1-score are preferable, indicating better performance. Additionally, lower execution times are favored as they signify faster model training. From the summarized results in **Table 3**. Multinomial Naive Bayes emerges as the top performer, boasting high accuracy (98.65%), precision (96.80%), recall (92.37%), and F1-score (94.53%), alongside minimal execution time. Although SVC also demonstrates strong performance, it comes with a slightly longer execution time. While models like Extra Trees and Random Forest show competitive accuracies exceeding 97%, their longer execution times may hinder scalability. Overall, Multinomial Naive Bayes stands out for its exceptional performance and efficiency, though SVC remains a viable alternative depending on specific requirements. Considering the results from **Table 4**, the Multinomial Naive Bayes classifier exhibited the highest accuracy (89.11%), precision (88.10%), and F1-score (87.06%). Additionally, it achieved a respectable recall score of 86.05%. Moreover, it demonstrated the shortest execution time among all classifiers, taking only 0.015 seconds.

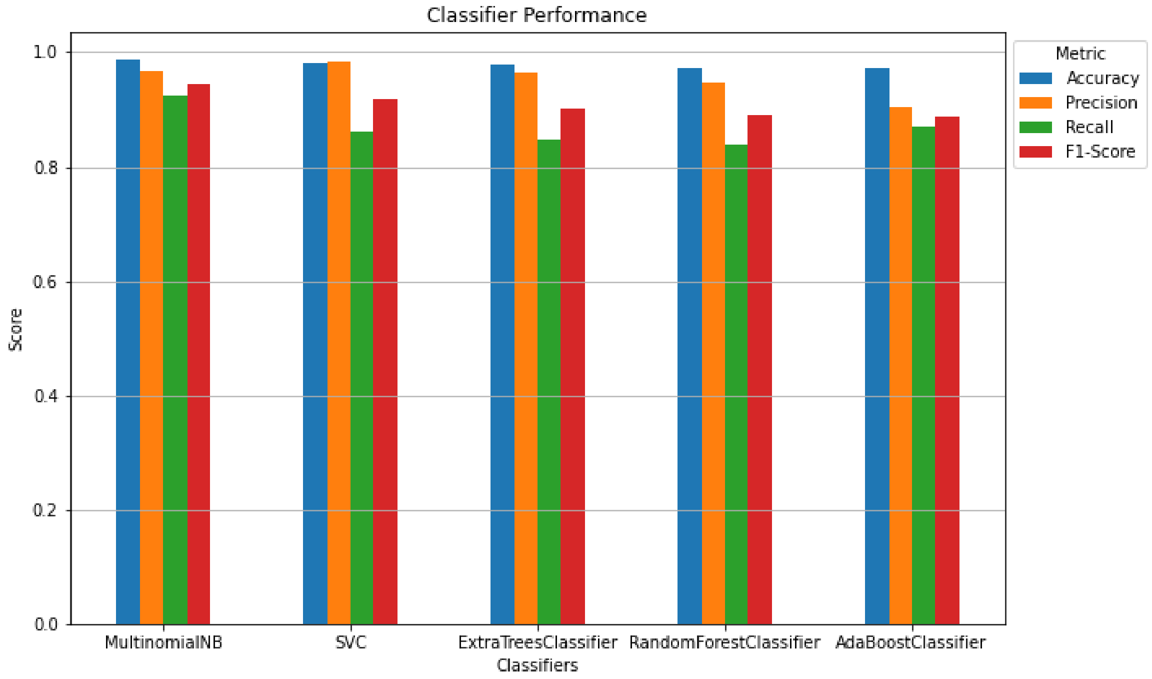


Figure 2. *Top five model performance SMS Collection v.1 dataset*

While the Support Vector Classifier (SVC) and Random Forest Classifier also delivered competitive performance, Multinomial Naive Bayes outperformed them in terms of both accuracy and execution time. The Bagging Classifier, AdaBoost Classifier, and Extra Trees Classifier followed suit, each showing comparable but slightly lower performance metrics compared to Multinomial Naive Bayes. It shows **Figure 2.** and **Figure 3**. for both datasets.

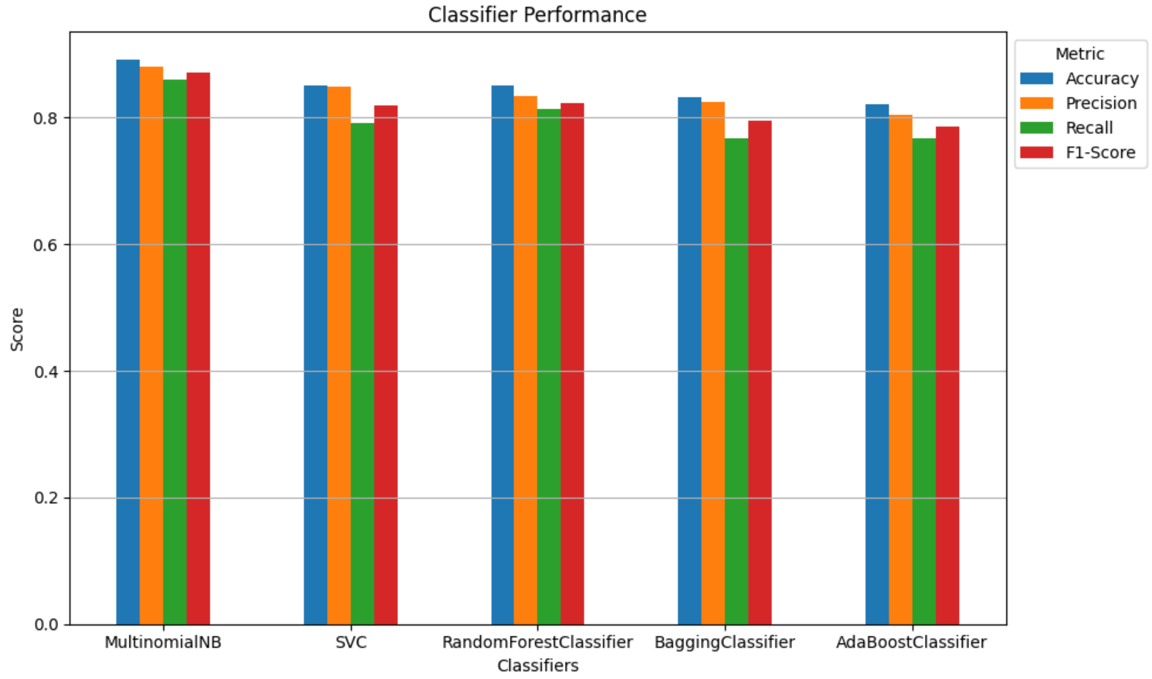


Figure 3. *Top five model performance Bangla SMS dataset*

On the other hand, the Logistic Regression, Gradient Boosting Classifier, Decision Tree Classifier, and K-Nearest Neighbors Classifier exhibited lower accuracy scores and longer execution times, making them less preferable choices in this scenario. Overall, Multinomial Naive Bayes stands out as the most efficient and effective classifier for two different datasets, offering a favorable balance between performance and computational cost. Thus, it is the recommended model for classification tasks on these datasets.

5. Conclusion and Future work

In conclusion, our analysis indicates that Multinomial Naive Bayes is the top-performing classifier for the given SMS spam datasets, exhibiting high accuracy and efficiency. However, further investigation could explore ensemble methods or deep learning techniques to potentially enhance classification performance. Additionally, incorporating more advanced feature engineering methods or exploring different text representation techniques may lead to improved model accuracy. Future work could also focus on deploying the selected classifier in real-world scenarios and evaluating its performance in a production environment.

References

[1] A. Alli and S. Misra, "A deep learning method for automatic SMS spam classification: Performance of learning algorithms on indigenous dataset," *Concurrency and Computation: Practice and Experience,* vol. 34, p. 34, 2022.

[2] S. D. Gupta, S. Saha and S. K. Das, "SMS spam detection using machine learning," in *Journal of Physics: Conference Series*, 2021.

[3] T. Almeida and J. Hidalgo, "SMS Spam Collection," 2011.

[4] X. Liu, H. Lu and A. Nayak, "A Spam Transformer Model for SMS Spam Detection," *IEEE Access,* vol. 9, pp. 80253-80263, 2021.

[5] S. Gadde, A. Lakshmanarao and S. Satyanarayana, "SMS Spam Detection using Machine Learning and Deep Learning Techniques," *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS),* vol. 1, pp. 358-362, 2021.

[6] P. J. Yerima and S, "A comparative study of word embedding techniques for SMS spam detection," *2022 14th International Conference on Computational Intelligence and Communication Networks (CICN),* pp. 149-155, 2022.

[7] D. Suleiman and G. Al-Naymat, "SMS spam detection using H2O framework," *Procedia computer science 113,* pp. 154-161, 2017.

[8] G. L. Haq, S. Nazir and H. U. Khan, "Spam Detection Approach for Secure Mobile Message Communication Using Machine Learning Algorithms," *Secur. Commun. Networks,* vol. 2020, pp. 8873639:1-8873639:6, 2020.

[9] L. P. Lim and M. M. Singh, "Resolving the imbalance issue in short messaging service spam dataset using cost-sensitive techniques," *Journal of Information Security and Applications,* vol. 54, p. 102558, 2020.

[10] E. Wijaya, G. Noveliora, K. D. Utami, Rojali and G. Z. Nabiilah, "Spam Detection in Short Message Service (SMS) Using Naïve Bayes, SVM, LSTM, and CNN," *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE),* pp. 431-436, 2023.

[11] E. Sankar, "Sms Spam Detection Using Machine Learning," *Interantional Journal Of Scientific Research In Engineering And Management*, 2023.

[12] Mahadev and H. Jain, "An Analysis of SMS Spam Detection using Machine Learning Model," *2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT),* pp. 151-156, 2022.

[13] A. A. M. Tasmia, A. A. N. ,. Jidney and Z. M. A. M. Haque, "Ensemble Approach to Classify Spam SMS from Bengali Text," in *Springer Nature,* kolkata, 2023.

[14] F. Khan, R. Mustafa, F. Tasnim, T. Mahmud, M. S. Hossain and K. Andersson, "Exploring BERT and ELMo for Bangla Spam SMS Dataset Creation and Detection," in *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 2023.

[15] R. G. d. Luna, V. C. Magnaye, R. A. L. Reaño, K. L. Enriquez, D. Astorga, T. Celestial, A. M. Española, B. A. Lanting, D. Mugar, M. Ramos and J. Redondo, "A Machine Learning Approach for

Efficient Spam Detection in Short Messaging System (SMS)," *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON),* pp. 53-58, 2023.

[16] R. G. d. L. Redondo, V. C. Magnaye, R. A. L. Reaño, K. L. E. a. D. Astorga, T. Celestial, A. M. Española, B. A. Lanting, D. Mugar, M. Ramos and Jenjazel, "A Machine Learning Approach for Efficient Spam Detection in Short Messaging System (SMS)," *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON),* pp. 53-58, 2023.

[17] Ojo and D. A. Oyeyemi, "SMS Spam Detection and Classification to Combat Abuse in Telephone Networks Using Natural Language Processing," *Journal of Advances in Mathematics and Computer Science,* 2023.

[18] S. Alghazzawi and D. Alqahtani, "A survey of Emerging Techniques in Detecting SMS Spam," *Transactions on Machine Learning and Artificial Intelligence,* 2019.

[19] U. M. Kundi, S. Rehman, T. Ali, K. Mahmood and T. Alsaedi, "An Intelligent Framework Based on Deep Learning for SMS and e-mail Spam Detection," *Appl. Comput. Intell. Soft Comput.,* vol. 2023, pp. 6648970:1-6648970:16, 2023.

[20] M. Gupta, A. Bakliwal, S. Agarwal and P. Mehndiratta, "A comparative study of spam SMS detection using machine learning classifiers," in *IEEE,* 2018.

[21] T. A. H. Almeida and Y. A. Jos'e Maria G, "Contributions to the study of SMS spam filtering: new collection and results," in *Association for Computing Machinery,* New York, NY, USA, 2011.

[22] Bashar and S. Yerima, "Semi-supervised novelty detection with one class SVM for SMS spam detection," *2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP),* Vols. CFP2255E-ART, pp. 1-4, 2022.

[23] S. Gadde, A. Lakshmanarao and S. Satyanarayana, "SMS spam detection using machine learning and deep learning techniques," *2021 7th international conference on advanced computing and communication systems (ICACCS),* vol. 1, pp. 358-362, 2021.

[24] E. W. Nabiilah, G. Noveliora, K. D. Utami, Rojali and G. Zain, "Spam Detection in Short Message Service (SMS) Using Naïve Bayes, SVM, LSTM, and CNN," *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE),* pp. 431-436, 2023.

[25] P. K. Roy, J. P. Singh and S. Banerjee, "Deep learning to filter SMS Spam," *Future Generation Computer Systems,* vol. 102, pp. 524-533, 2020.

[26] S. Yadav and A., "Mobile SMS Spam Filtering for Nepali Text Using Naïve Bayesian and Support Vector Machine," *International Journal of Intelligent Systems,* vol. 04, pp. 24-28, 2014.

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

Vol. 4 (No. 2), pp. 78-86, 2024
doi: 10.54569/aair.1585057
Published online: Dec 30, 2024

Enhancing Efficiency in Virtualized Environments: Intelligent Solutions for VMware Errors through Artificial Intelligence and API Integration

Çetin Budak [1], Gül Fatma Türker [1,*]

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

Abstract

Today, virtualization technologies play a critical role in the information technology sector, enabling businesses to manage their infrastructures more efficiently and flexibly. However, the complex structure and extensive feature set of virtualization programs often lead users to encounter error codes and technical issues. This can result in significant time losses and decreases in productivity, especially for inexperienced users. To address these challenges quickly and effectively, the IT sector is increasingly focusing on integrated solutions developed with artificial intelligence and smart assistant technologies. This study introduces the "VMware Assistant" software, designed to address technical issues related to VMware products. VMware Assistant utilizes pre-trained AI models, APIs from various websites, and comprehensive error and warning datasets to automatically detect and provide real-time solutions for user issues. The system aims to alleviate the complexities associated with virtualization programs and offer practical support to users. VMware Assistant consolidates error and warning data from multiple sources, enabling users to swiftly access the information they need. As a result, it accelerates the resolution process for technical issues encountered in virtualized environments, allowing users to maintain workflow continuity. VMware Assistant was developed to facilitate the use of virtualization technologies, potentially enhancing productivity in the IT sector and contributing to technical support processes.

Keywords: *API integration, Artificial intelligence, Error detection, Virtualization, VMware.*

1. Introduction

With the rapid advancement of information technology, businesses are seeking new methods to make their infrastructures more flexible, scalable, and efficient. Virtualization technologies address these needs by enabling more effective use of physical resources [1]. Virtualization allows multiple operating systems and applications to run on a single physical machine by abstracting hardware resources [2], optimizing resource utilization and reducing hardware costs [3]. VMware, a global leader in virtualization solutions, provides comprehensive products and services to businesses across various sectors [4]. VMware vSphere offers an extensive platform for server virtualization, management, and business continuity [5]. However, the complex structure of VMware products and their continuously evolving features often lead users to encounter technical issues, error messages, and intricate configurations [6]. Failure to address these issues promptly and accurately can degrade system performance, disrupt business processes, and decrease operational efficiency [7]. Traditional support methods, particularly in complex virtualization environments, may not be sufficiently fast or effective, resulting in additional costs and time losses for businesses [8]. In this context, self-service and automated solutions are gaining importance [9]. Artificial Intelligence (AI) and Machine Learning (ML) technologies offer significant opportunities to overcome these challenges [10]. With Natural Language Processing (NLP) techniques, users can express their issues in natural language, enabling systems to comprehend these inputs and provide appropriate solutions [11]. AI-based intelligent assistants can analyze large datasets and historical support records to offer fast and accurate solutions for complex issues, enhancing user experience and satisfaction [12]. These assistants continuously learn and improve through user interactions and feedback [13, 14].

The literature presents various studies on the integration of AI and NLP into technical support and troubleshooting processes [15]. In this context, this study introduces the "VMware Assistant" application, developed to assist in diagnosing and resolving technical issues associated with VMware products. VMware Assistant leverages AI models, APIs from various websites, and comprehensive error and warning datasets to automatically detect user issues and provide real-time, effective solutions. The system analyzes issues expressed by users in natural language, identifies relevant error codes, assesses past solutions for similar issues, and provides the most appropriate solution recommendations. This approach aims to expedite troubleshooting processes in virtualized environments, minimize downtime, and enhance overall user satisfaction. The VMware

*Corresponding author
E-mail address: gulturker@sdu.edu.tr

Received: 14/Nov/2024; Accepted: 28/Dec/2024.

Assistant offers a platform accessible to users at any technical expertise level. Its user-friendly interface and interactive design enable users to quickly reach solutions without dealing with complex technical details. Additionally, using this assistant can reduce the workload of technical support teams, contributing to more efficient resource utilization.

The following sections of the study will detail the architecture of VMware Assistant, the AI and machine learning methods employed, system integration processes, and the results obtained. This study aims to facilitate the use of virtualization technologies, enhance productivity in the IT sector through an innovative approach to technical support, and minimize the challenges faced by users working with VMware products.

2. Materials and Methods

This study focuses on the development of "VMware Assistant," a software that provides comprehensive and user-friendly support for resolving errors and warnings encountered in VMware products. By analyzing error and warning codes reported by users, VMware Assistant aims to offer meaningful and effective solution recommendations through an integrated use of predefined real error and warning datasets as well as new error and solution information contributed by users. This approach not only facilitates the quick resolution of common issues but also encourages users to contribute to the system by sharing their own experiences and solutions. The general operation of VMware Assistant, including methods and data flow, is designed to detail the troubleshooting processes for issues encountered by users. The Flow Diagram is presented in Hata! Başvuru kaynağı bulunamadı..

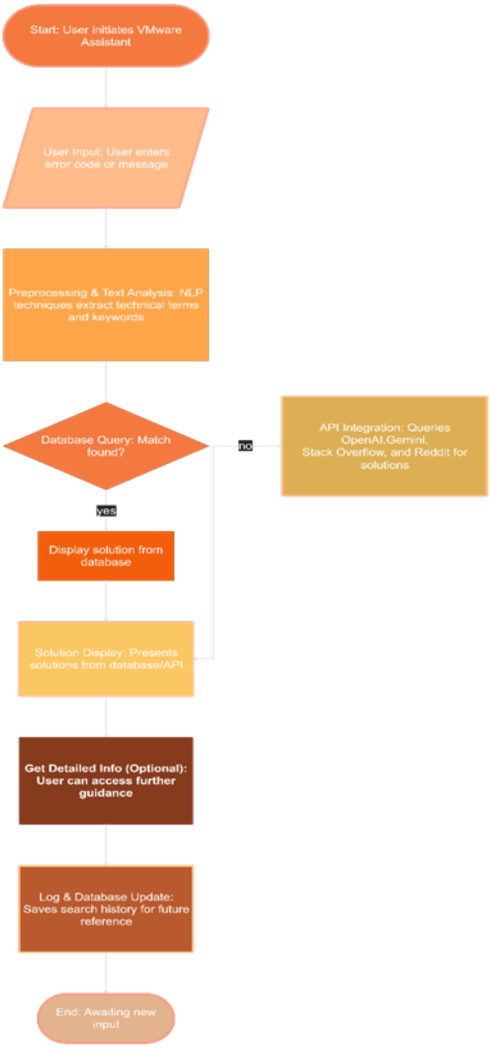

Start: User initiates VMware Assistant
User Input: User enters error code or message
Preprocessing & Text Analysis: NLP techniques extract technical terms and keywords
Database Query: Match found?
no
API Integration: Queries OpenAI,Gemini, Stack Overflow, and Reddit for solutions
yes
Display solution from database
Solution Display: Presents solutions from database/API
Get Detailed Info (Optional): User can access further guidance
Log & Database Update: Saves search history for future reference
End: Awaiting new input

Figure 1. VMware assistant application flow diagram

2.1. VMware assistant application and library requirements

VMware Assistant is developed using the Python programming language and is supported by various libraries and APIs. The primary components used in the development and operation of VMware Assistant are as follows:

- Python 3.x: Serves as the main programming language for the VMware Assistant software [16].
- CustomTkinter: Built on top of Tkinter, it is used to create a modern and customizable graphical user interface (GUI). This library facilitates the design of a user-friendly and aesthetically pleasing interface [17].
- SQLite3: A lightweight and embedded database solution used for storing error and warning codes, solution recommendations, search history, and user-contributed data. SQLite3 is preferred due to its lack of server dependency and ease of integration [18].
- OpenAI API: Provides natural language processing (NLP) and artificial intelligence capabilities, enabling the system to interpret free-text error and warning messages submitted by users and present relevant VMware solutions [19].
- PRAW (Python Reddit API Wrapper): Used to retrieve VMware-related topics and discussions on Reddit, presenting community-generated solution suggestions to users. This integration allows access to current and practical solutions [20].
- Stack Exchange API: Integrated to fetch VMware-related questions and answers from Stack Overflow, leveraging the experiences of a wide developer and user base [21].
- Logging: Used to record errors and events that occur during the operation of VMware Assistant, facilitating easier maintenance and helping to identify potential issues [22].
- Pygame: Employed to create the loading screen and other animated elements, enhancing the user experience through graphical components [23].

2.2. Dataset structure

The dataset underlying the VMware Assistant application includes real error and warning codes commonly encountered in VMware products. This dataset encompasses frequently observed issues within VMware's virtualization solutions, such as vCenter, ESXi, and vSphere, along with recommended solutions for these issues. The dataset is structured with various attributes: each error and warning is represented by a unique code and description, either officially defined by VMware or widely recognized by the user community. The solution recommendations provided for these codes include detailed troubleshooting steps and explanations based on official documentation, technical articles, and community contributions, offering users effective guidance.

Additionally, users can contribute by adding the error and warning codes they encounter, along with the corresponding solutions, which helps maintain the database's relevance and allows for the sharing of diverse experiences. The "Get Detailed Information" button within VMware Assistant uses OpenAI and Gemini APIs to provide supplementary insights regarding a specific error or warning. This feature enables users to perform a more in-depth analysis and assess issues from a broader perspective, even if comprehensive information about the error or warning is already available in the database.

2.3. Models and algorithms

VMware Assistant employs various algorithms and methodologies to provide users with the most effective and prompt solutions. Firstly, natural language processing (NLP) techniques are utilized by leveraging OpenAI’s GPT models to analyze errors and warnings expressed by users in natural language. This approach enables users to articulate their issues comfortably, even if they are unfamiliar with technical terminology, and receive relevant solutions accordingly [24]. Additionally, a keyword matching method is implemented to compare key terms and phrases in user inputs with records in the database, allowing for quick and direct matches.

When a user enters an error or warning code, it is compared against existing records in the database; if a match is found, the corresponding solution recommendations and supplementary information are provided directly to the user. In cases where no match is found in the database or where more detailed information is needed, the system switches to an API-based solution retrieval method. At this stage, additional solution recommendations and discussions are accessed through OpenAI, Stack Overflow, and Reddit APIs, enabling a multi-perspective approach to the user's issue.

This combination of NLP, keyword matching, and API-based solution retrieval provides a robust framework for VMware Assistant, ensuring users receive comprehensive and contextually relevant support.

2.4. Functional structure of VMware assistant equations

VMware Assistant is a software application designed to quickly and effectively analyze errors and warnings encountered by users of VMware products and provide solution recommendations. Utilizing text analysis techniques to interpret error messages from users, this assistant retrieves relevant solutions from the existing database. If no matching solutions are found, it sources up-to-date community solutions from external resources and presents them to the user. User queries and added solutions are regularly updated, evolving into a continuously expanding knowledge base. The process flow is illustrated through the following pseudocode:

```
BEGIN VMwareAssistant
//Get user input
INPUT userInput
//Pre-process and analyze input
userKeywords = NLP_Analyze(userInput)
 //Query the database with user input
queryResult = Database_Query(userKeywords)
//Check if match is found
IF queryResult is FOUND THEN
//Display the solution to the user
Display_Solution(queryResult)
ELSE
//Use API integration for additional information
detailedInfo = Get_Detailed_Info_API(userKeywords)
//Display API data to the user
 Display_Solution(detailedInfo)
ENDIF
//Present results to the user
Display_Result(userInput, queryResult OR detailedInfo)
//Update search history and database
Update_Database(userInput, queryResult OR detailedInfo)
END VMwareAssistant
```

The VMware Assistant application provides a faster, more stable, and multi-layered analysis approach to troubleshooting compared to standard search methods. This process involves several stages. First, in the user input processing stage, users enter the error or warning they encounter into a text box; this input can be an error code, error message, or description of the problem. Next, during the preprocessing and text analytics stage**,** the entered text is analyzed using natural language processing (NLP) techniques and keyword extraction methods, enabling the identification of technical terms, error codes, and critical expressions.

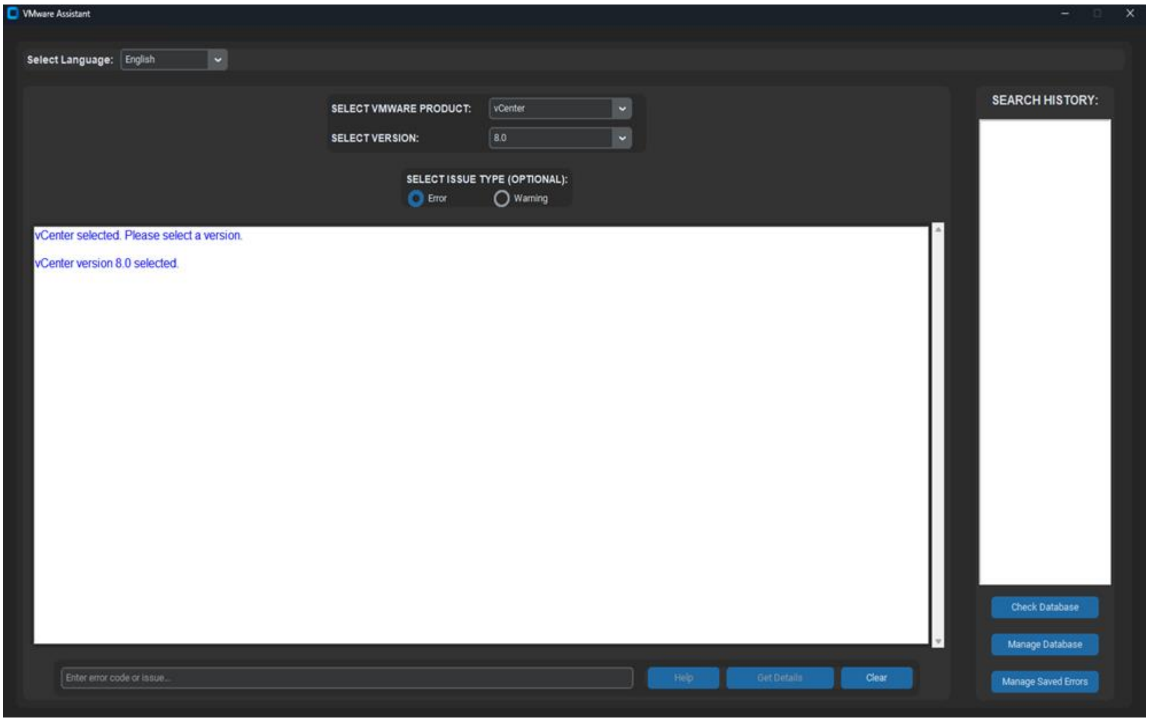
In the database querying and issue matching stage, the analyzed input is compared with existing records in the database. If a matching error or warning is found, the relevant solution recommendations and additional information are directly presented to the user. When no direct match exists in the database, or if additional information is required, external API integration is activated. At this point, alternative solution searches are conducted using the OpenAI, Stack Overflow, and Reddit APIs; through these queries, up-to-date and community-approved solution suggestions related to users' issues are retrieved.

Finally, in the solution presentation and user notification stage, all obtained solution recommendations are clearly and systematically displayed in the user interface. Searches and solutions contributed by users are logged in the search history and database updates, ensuring they are available for future access. This structure enhances user experience and enables the application to serve as a more effective support system.

2.5. Graphical user interface

The graphical user interface (GUI) of VMware Assistant is designed with a simple, intuitive, and user-friendly layout to ensure that users can access needed solutions easily and quickly. The interface has been meticulously structured to optimize user interaction and streamline the process of accessing error resolutions. **Figure 2** displays the graphical user interface of the VMware Assistant software.

The user interface of VMware Assistant incorporates various functional areas to maximize the user experience. First, the input field is structured as a text box where users can enter error or warning codes and descriptions, supporting a fast and efficient search process. The search history section provides a structure where users can view and quickly access previous searches, facilitating rapid access to solutions for the same or similar issues. The solution recommendations area lists suggestions obtained from the database and APIs, presenting users with the most suitable resolution paths. Additionally, through the "Get Detailed Information" button, users can access additional insights and analyses thanks to the integrations with OpenAI and Gemini APIs. The user contributions section allows users to share their own experiences and solutions, creating a collective knowledge base. Finally, the settings and help menus enable users to adjust application settings and access the support and guidance they need. The GUI design is responsive, adapting to different screen sizes to provide a seamless experience across various devices.



Select Language: English

SELECT VMWARE PRODUCT: vCenter

SELECT VERSION: 8.0

vCenter selected. Please select a version.

vCenter version 8.0 selected.

Figure 2. Graphical user interface (GUI)

3. Research Findings

The capabilities of VMware Assistant in detecting and analyzing errors and warnings encountered in VMware products are examined in detail. The software utilizes a continuously updated database, which includes not only commonly encountered error and warning datasets in the VMware ecosystem but also custom error and warning codes that users can add to the system. This structure allows the application to cover a wide range of errors, ensuring it remains up-to-date through user contributions.

3.1. Dataset and user-contributed error entries

VMware Assistant features a comprehensive dataset that includes frequently encountered error and warning codes for VMware products such as vCenter, ESXi, and Workstation. This dataset is supported by realistic error data gathered from various sources, enabling the software to provide accurate solution recommendations. A notable feature of VMware Assistant is its ability to allow users to add their own encountered errors and warnings to the system. This feature permits users to share their experiences and contribute new solution information in cases where errors and warnings are not already recorded in the system. This approach enables the database to expand with user experiences, offering flexible solutions for a wider array of scenarios. Additionally, users can review existing solution recommendations and select the responses most suited to their own issues. This bidirectional structure distinguishes VMware Assistant from traditional static error management systems, providing a user-centered experience.

3.2. Model and API performance evaluation

VMware Assistant leverages various models and APIs to accurately analyze error and warning codes entered by users and provide relevant solutions. The OpenAI API uses natural language processing (NLP) techniques to analyze errors and offer VMware-specific solution recommendations. This approach enables the system to respond to even unrecorded errors with solution suggestions supported by OpenAI’s insights. The Gemini API, on the other hand, retrieves data from popular platforms like Reddit and Stack Overflow, granting access to community-shared, up-to-date solution recommendations. This API-based structure allows VMware Assistant to address a broad range of errors, providing users with a more comprehensive approach to their issues.

Additionally, VMware Assistant enables users to directly log their encountered errors and solutions into the system. This feature is especially valuable for resolving rarely encountered or custom-configuration errors unique to VMware. Through this feature, users can add unique issues to the system, making this information accessible to others. This structure not only enhances flexibility and encourages user interaction but also fosters community contribution, enriching the overall support ecosystem.

3.3. Performance evaluation and user experience

Performance testing of VMware Assistant has demonstrated its ability to accurately identify errors and provide realistic solution suggestions to users. VMware Assistant aims to enhance user satisfaction by offering quick and practical solutions, especially for VMware users facing complex issues. The user interface provides convenient areas for functions such as product and version selection and error and warning code entry, facilitating ease of use. This enables users to access solution recommendations through a fast and intuitive experience. Testing results indicate that VMware Assistant can recognize even complex errors and warnings, providing detailed solution recommendations and guiding users effectively.

During the performance evaluation process, improvements in user experience were noted, making error detection and solution recommendation processes more practical. VMware Assistant, with its user-friendly interface, not only aids in the rapid resolution of issues but also serves as a reliable resource for users facing complex VMware problems. Additionally, through the "Get Detailed Information" button, more comprehensive results are provided via the ChatGPT and Gemini APIs, offering users alternative solutions. This feature presents a more detailed and multi-dimensional solution perspective for users addressing their issues.

4. Results and Discussion

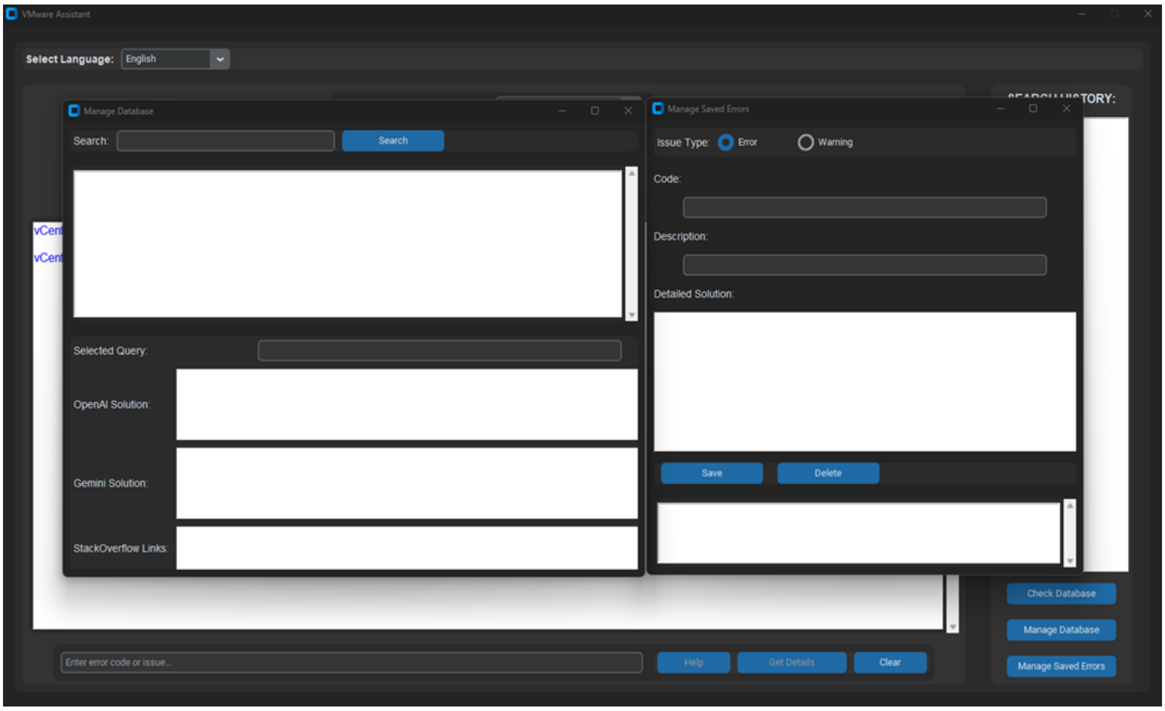
The error and warning detection and analysis capabilities provided by the VMware Assistant software have been evaluated. The analysis was conducted using a dataset of common error and warning codes encountered in VMware products. Additionally, with the ability for users to add their own errors and warnings to the system, the database is continuously updated and expanded. This dual-layered database structure allows VMware Assistant to provide solutions over a broader range of error types. The findings indicate that VMware Assistant offers significant advantages in error detection and solution recommendations. The software’s ability to combine a standard dataset with user-contributed entries ensures a comprehensive and dynamic knowledge base, enhancing the effectiveness and scope of support it provides to users working with VMware products.

4.1. Dataset and user-interactive error entries

VMware Assistant utilizes a dataset that includes frequently encountered error and warning codes for VMware products such as vCenter, ESXi, and Workstation. These codes consist of error messages sourced from official documentation and commonly known community errors. One of the software's key features is its ability to allow users to add the errors they encounter to the system. This feature enables VMware Assistant to offer solutions across a wide spectrum, including rare errors within the VMware ecosystem. Users contribute by adding their experienced issues, benefiting others as shown in **Figure 3**. This dynamic and interactive structure distinguishes VMware Assistant from static error management systems.

In a 2023 study by Li et al. [25], a machine learning-based approach was proposed for error detection and diagnosis in cloud-based systems. This study analyzes system logs to detect anomalies and identify potential error causes. However, the approach relies on a static dataset and does not incorporate user interaction. VMware Assistant, by contrast, differentiates itself with a continually evolving knowledge base supported by active user participation, making it a more comprehensive and adaptable solution.

Similarly, Bhardwaj et al.in a 2024 paper [26], proposed a deep learning-based system for error detection in Kubernetes clusters. This system detects anomalies by analyzing resource usage metrics and predicts potential errors. However, this study also lacks user interaction and community-based knowledge sharing. VMware Assistant, with its interactive features, reaches a broader user base and offers a more effective solution for resolving rare errors, leveraging the power of community contributions.



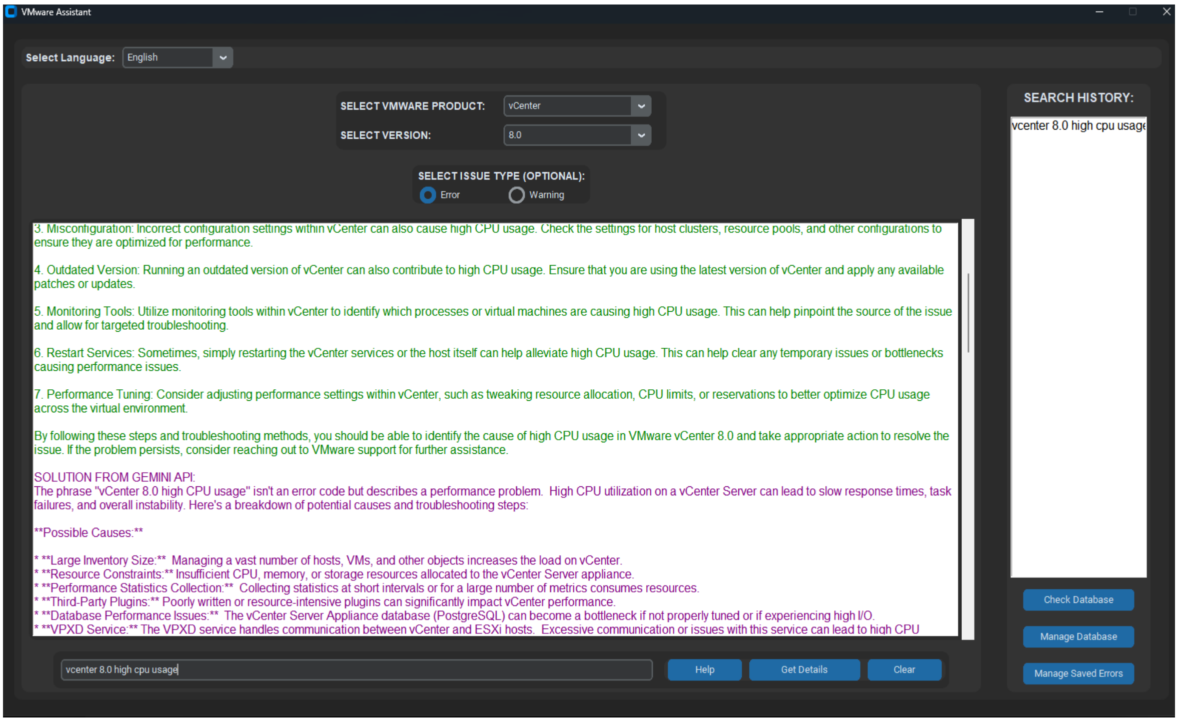
Select Language: English

Issue Type: ● Error ○ Warning

Figure 3. Active error management

4.2. Model and API performance evaluation

VMware Assistant employs various APIs and natural language processing (NLP) techniques to interpret errors and provide solution recommendations. The OpenAI API analyzes errors entered in free-text format by users, producing meaningful outcomes. This API plays a crucial role in resolving complex and naturally expressed issues. Additionally, the Gemini API retrieves up-to-date, community-based solutions from platforms like Reddit and Stack Overflow, presenting these to the user. This multi-layered solution search, illustrated in **Figure 4.** NLP technique, is particularly effective in providing solutions for new or rarely encountered errors not found in the database. VMware Assistant offers solution recommendations based not only on its database but also on results from external APIs. This hybrid approach provides users with the most suitable recommendations by addressing their issues from both a broad and current solution base. In a 2022 study published by Jadav et al. in the International Journal of Communication Systems [27], an AI-based system was proposed for error detection and troubleshooting in software-defined networks (SDNs). This system detects anomalies by analyzing network traffic data and identifies potential causes of errors. However, the AI model used in that study lacks the multi-API integration and NLP capabilities employed by VMware Assistant. Consequently, VMware Assistant covers a broader range of errors and offers more effective solutions, enhancing its utility and adaptability.



Select Language: English

SELECT VMWARE PRODUCT: vCenter

SELECT VERSION: 8.0

SELECT ISSUE TYPE (OPTIONAL): ● Error ○ Warning

SEARCH HISTORY: vcenter 8.0 high cpu usage

```
3. Misconfiguration: Incorrect configuration settings within vCenter can also cause high CPU usage. Check the settings for host clusters, resource pools, and other configurations to
ensure they are optimized for performance.

4. Outdated Version: Running an outdated version of vCenter can also contribute to high CPU usage. Ensure that you are using the latest version of vCenter and apply any available
patches or updates.

5. Monitoring Tools: Utilize monitoring tools within vCenter to identify which processes or virtual machines are causing high CPU usage. This can help pinpoint the source of the issue
and allow for targeted troubleshooting.

6. Restart Services: Sometimes, simply restarting the vCenter services or the host itself can help alleviate high CPU usage. This can help clear any temporary issues or bottlenecks
causing performance issues.

7. Performance Tuning: Consider adjusting performance settings within vCenter, such as tweaking resource allocation, CPU limits, or reservations to better optimize CPU usage
across the virtual environment.

By following these steps and troubleshooting methods, you should be able to identify the cause of high CPU usage in VMware vCenter 8.0 and take appropriate action to resolve the
issue. If the problem persists, consider reaching out to VMware support for further assistance.

SOLUTION FROM GEMINI API:
The phrase "vCenter 8.0 high CPU usage" isn't an error code but describes a performance problem.  High CPU utilization on a vCenter Server can lead to slow response times, task
failures, and overall instability. Here's a breakdown of potential causes and troubleshooting steps:

**Possible Causes:**

* **Large Inventory Size:**  Managing a vast number of hosts, VMs, and other objects increases the load on vCenter.
* **Resource Constraints:** Insufficient CPU, memory, or storage resources allocated to the vCenter Server appliance.
* **Performance Statistics Collection:**  Collecting statistics at short intervals or for a large number of metrics consumes resources.
* **Third-Party Plugins:** Poorly written or resource-intensive plugins can significantly impact vCenter performance.
* **Database Performance Issues:**  The vCenter Server Appliance database (PostgreSQL) can become a bottleneck if not properly tuned or if experiencing high I/O.
* **VPXD Service:** The VPXD service handles communication between vCenter and ESXi hosts.  Excessive communication or issues with this service can lead to high CPU
```

vcenter 8.0 high cpu usage

Figure 4. NLP technique

4.3. Performance evaluation and user experience

VMware Assistant's solution offerings and interface design are optimized to enhance user satisfaction and accelerate issue resolution. The interface options allow users to easily enter encountered error or warning messages and, by clicking the "Get Detailed Information" button, obtain a more comprehensive solution enriched with additional details from the OpenAI and Gemini APIs. VMware Assistant's ability to analyze complex errors and warnings provides solutions that closely reflect real-world scenarios, making it a valuable resource for technical experts who work extensively with VMware products.

Testing has shown that VMware Assistant offers a user-friendly experience in error detection and solution recommendation, effectively communicating solutions to users. The user-contributed database and API integration expand the information available, enabling solutions tailored to users' needs. Based on these findings, VMware Assistant can be regarded as a user-friendly and effective tool for managing errors and warnings encountered in VMware products.

A 2016 study by Gulzar et al., presented at the 8th USENIX Workshop on Hot Topics in Cloud Computing [28], introduced an interactive system for debugging in cloud environments, allowing users to visualize and analyze error information. However, the interface presented in that study is not as user-friendly and intuitive as that of VMware Assistant. VMware Assistant prioritizes user experience by offering a simple and effective interface, enabling users to quickly access errors and solutions. Compared to existing academic studies, VMware Assistant presents numerous innovative features. With its multi-layered database structure, API integration, community- based knowledge sharing, and hybrid solution approach, VMware Assistant stands out as a distinctive tool for error detection and solution recommendations tailored for VMware users.

5. Conclusion

VMware Assistant has been developed as a user-friendly and flexible support software that provides effective solutions for errors and warnings encountered in VMware products. With its comprehensive database and API-supported structure, it offers quick and efficient solution recommendations for common errors in products such as VMware vCenter, ESXi, and Workstation. The ability for users to add their own experiences with errors and solutions helps keep VMware Assistant up-to-date and fosters a community-based knowledge base. During development, various natural language processing techniques and APIs were integrated to enhance the understanding and solution delivery for complex error and warning messages. Free-text analysis via the OpenAI API, combined with community-based solutions from platforms like Reddit and Stack Overflow through the Gemini API, allows users to receive answers from multiple perspectives. The flexibility provided by this structure enables the software to support users with practical and current solutions even for errors not recorded in the system. Future work could focus on testing VMware Assistant with a larger dataset in real-world environments, creating more comprehensive data collection processes to improve solution accuracy, and expanding API integrations. Additionally, to enhance performance and user experience, the integration of advanced algorithms such as visual analysis and machine learning-based recommendation systems could offer solutions for other potential issues within the VMware ecosystem. Such improvements would make VMware Assistant a more effective error management tool for both businesses and individual users, potentially achieving widespread adoption in the virtualization field. Future developments may include the integration of advanced machine learning models, comprehensive community involvement, and feedback systems.

References

[1] Smith, J., Nair, R., "Virtual Machines: Versatile Platforms for Systems and Processes," *IBM Systems Journal*, vol. 44, no. 2, pp. 365–382, 2005.

[2] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T. L., Ho, A., ... and Warfield, A., "Xen and the art of virtualization," in *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, 2003, pp. 164–177, doi: 10.1145/1165389.945462.

[3] Clark, C., Fraser, K., Hand, S., Hansen, J. G., Jul, E., Limpach, C., ... and Pratt, I., "Live migration of virtual machines," in *Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation*, vol. 2, 2005, pp. 273–286.

[4] VMware, Inc., *White Paper: Understanding VMware vSphere*, Palo Alto, CA, 2023. [Online]. Available: https://www.vmware.com/resources/whitepapers/understanding-vsphere.html. [Accessed: Dec. 22, 2024].

[5] VMware, Inc., *VMware vSphere Documentation*, 2023. [Online]. Available: https://docs.vmware.com/en/VMware-vSphere. [Accessed: Dec. 22, 2024].

[6] Jin, H., and Patel, P., *Troubleshooting Techniques for VMware vSphere*. Pearson Education, 2011.

[7] Mell, P., and Grance, T., "The NIST definition of cloud computing," *NIST Special Publication*, vol. 800, no. 145, p. 7, 2011.

[8] Rosenblum, M., and Garfinkel, T., "Virtual machine monitors: Current technology and future trends," *Computer*, vol. 38, no. 5, pp. 39–47, 2005, doi: 10.1109/MC.2005.176.

[9] Nurmi, D., Wolski, R., Grzegorczyk, C., and Obertelli, G., "The Eucalyptus open-source cloud-computing system," in *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2009, pp. 124–131.

[10] Russell, S. J., and Norvig, P., *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

[11] Jurafsky, D., and Martin, J. H., *Speech and Language Processing*. Pearson Education, 2019.

[12] Maedche, A., Legner, C., Benlian, A., Berger, B., Gimpel, H., Hess, T., ... and Söllner, M., "AI-based digital assistants," *Business & Information Systems Engineering*, vol. 61, pp. 635–644, 2019.

[13] Janarthanam, S., and Nielsen, A., *Evolving Conversational Intelligence: How to Build an AI-Powered Chatbot*. Manning Publications, 2019.

[14] Fenu, G., and Repetto, M., "Chatbots in education: A survey," in *Proceedings of the 2nd International Workshop on Intelligent Bots*, 2018, pp. 23–34.

[15] Chen, Y., Xu, J., Zhang, Z., and Liu, Z., "A survey on chatbot technology," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–36, 2020.

[16] Van Rossum, G., "Python programming language," in *Encyclopedia of Computer Science and Technology*, vol. 43, Marcel Dekker, 2007, pp. 163–170.

[17] Schimansky, T., *CustomTkinter*. 2023. [Online]. Available: https://customtkinter.tomschimansky.com. [Accessed: Dec. 22, 2024].

[18] Hipp, D. R., *SQLite*. SpringerBriefs in Computer Science, Springer International Publishing, 2021.

[19] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... and Amodei, D., "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[20] Boe, B., *PRAW: The Python Reddit API Wrapper*. 2023. [Online]. Available: https://praw.readthedocs.io/. [Accessed: Dec. 22, 2024].

[21] Stack Exchange Network, *Stack Exchange API Documentation*. 2023. [Online]. Available: https://api.stackexchange.com/docs. [Accessed: Dec. 22, 2024].

[22] Python Software Foundation, *Logging HOWTO*. 2023. [Online]. Available: https://docs.python.org/3/howto/logging.html. [Accessed: Dec. 22, 2024].

[23] Shinners, P., *Pygame Essentials*. Packt Publishing Ltd., 2011.

[24] Chowdhury, M., and Sadek, A. W., "Using natural language processing for improving question answering in online forums," *Decision Support Systems*, vol. 54, no. 1, pp. 528–540, 2012.

[25] Li, Z., et al., "Anomaly Detection and Diagnosis in Cloud Systems Using Machine Learning," *IEEE Transactions on Cloud Computing*, 2023.

[26] Bhardwaj, A. K., Dutta, P. K., & Chintale, P., "AI-Powered Anomaly Detection for Kubernetes Security: A Systematic Approach to Identifying Threats," Babylonian Journal of Machine Learning, 2024, pp. 142–148, doi: 10.58496/BJML/2024/014

[27] Jadav, N. K., Nair, A. R., Gupta, R., Tanwar, S., Lakys, Y., & Sharma, R., "AI-Driven Network Softwarization Scheme for Efficient Message Exchange in IoT Environment Beyond 5G," *International Journal of Communication Systems*, 2022, e5336, doi: 10.1002/dac.5336.

[28] Gulzar, M. A., Han, X., Interlandi, M., Mardani, S., Tetali, S. D., Millstein, T., & Kim, M., "Interactive Debugging for Big Data Analytics," In: *Proceedings of the 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16)*, 2016.

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

Vol. 4 (No. 2), pp. 87-95, 2024
doi: 10.54569/aair.1585994
Published online: Dec 30, 2024

Prediction of Home Loan Approval with Machine Learning

Gamze Güder [1, *], Utku Köse [1]

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

Abstract

With the introduction of computers into our lives, the size and complexity of data have increased. The growing amount of data made manual processing more difficult, and machine learning methods were adopted to minimize human errors. In the banking sector, the increasing volume of data necessitated the use of machine learning techniques. Numerous studies have been conducted in the literature on the banking sector. In this study, machine learning methods, including k-nearest neighbors, random forest algorithm, support vector machines, and logistic regression, were used to predict whether a bank would approve a housing loan or not. Two different datasets were used for the analysis. The results were compared and presented using performance metrics. This study aims to minimize human errors, make the credit approval processes in banks safer, and provide faster results for loan applications.

Keywords: *Knn algorithm; Random Forest algorithm; Support vector machines; Logistic regression.*

1. Introduction

The concept of housing has always maintained its importance throughout history. In ancient times, humans sought shelter primarily for protection from wild animals. However, with the development of technology and the changing needs of society, the concept of the right to housing emerged [1]. As the importance of the right to housing increased, efforts were made to secure it through laws, declarations, and other means. In Turkey, the right to housing has been attempted to be secured through Article 57 of the Constitution [2].

Economic fluctuations and rising inflation have reduced people's purchasing power. Just as in the rest of the world, the decline in purchasing power in Turkey has made access to housing, one of the most basic human rights, more difficult. The increase in rental prices in Turkey has made taking out loans to buy a house more attractive. Loans, one of the main products of banks, are funded by savers who deposit their savings into the bank, and thus, banks aim to achieve the highest profit from loans [3]. Just as important as giving loans, the repayment of those loans is also crucial for banks. For this reason, it has become important for banks to use machine learning techniques to minimize human error and make predictions by utilizing past data during the loan approval process. By using machine learning algorithms, banks are able to conduct faster and more secure processes. In the growing banking sector, where technology plays an increasingly important role, machine learning techniques are frequently employed. The literature contains many studies on loan approval processes, using various machine learning techniques and performance metrics.

In this study, two different datasets were used, which distinguishes it from other studies. The goal was to measure the compatibility and consistency of the results of the algorithms using these two datasets. Additionally, the study observed how the models were affected by different types of data. The k-nearest neighbors algorithm, random forest algorithm, support vector machines, and logistic regression algorithms were used in housing loan approval processes, and the results were presented in a comparative manner using performance metrics. For each algorithm, 90% of the two datasets were used for training data, while 10% was used for testing data.

2. Related Works

Arun et al. [4]: The study focused on predicting the most suitable customer for credit by evaluating customer applications. Decision trees, RF, linear models, neural networks, and AdaBoost algorithms were used in the study. However, the article did not mention the performance evaluation criteria or which algorithm produced the best results.

Gautam et al. [5]: This study focuses on predicting whether a loan will be approved by financial institutions by analyzing the history and reliability of customers applying for credit using machine learning methods. The dataset used is a collection from the banking sector. The results of the study show that the majority of the standard needs of bank employees are met. Additionally, the study emphasizes that the system was trained and tested with current data. The potential for these data to lose their relevance over time was taken into account. Therefore, the importance of integrating artificial intelligence systems with automation systems to incorporate new data into the system was highlighted.

Aphale et al. [6]: This study used a dataset from a real cooperative bank. Two-thirds of the dataset was used

*Corresponding author
E-mail address: gamzeyildirimmm@hotmail.com

Received: 15/Nov/2024; Accepted: 28/Dec/2024.

for training, and one-third for testing. Various algorithms were employed in this study, and the models were compared using performance evaluation criteria. The results showed that the performance evaluation metrics for the Nearest Centroid and Gaussian Naive Bayes algorithms performed reliably well. The algorithms that showed good performance each achieved values ranging from 76% to 80%. Additionally, a predictive model was formulated using linear regression to forecast customers' creditworthiness.

Gupta et al. [7]: This study attempts to predict whether a customer is reliable for a loan by analyzing their previous credit records using machine learning techniques. The dataset used in this study was obtained from Kaggle’s open access section. A heatmap was used to show the relationships between the parameters. Supervised learning approaches, specifically Logistic Regression and Random Forest algorithms, were used in the study. Customer information (such as gender, number of dependents, marital status, whether they run their own business, income, etc.) is entered through a user interface and the resulting credit approval status is displayed on the screen.

Ndayisenga et al. [8]: The study aimed to predict whether the repayment of loans in the Rwandan banking sector would occur by assessing an individual's past data, essentially estimating the credit risk and determining whether loan applicants would be approved. The dataset from Kigali Bank, operating in Rwanda, was used. When various models were compared based on performance evaluation metrics, the best results were observed with the Gradient Boosting model. The study concluded that customers with a credit score of B had a low probability of defaulting on their loans.

Fati et al. [9]: In the study, credit status prediction was performed using machine learning algorithms. A dataset from Kaggle’s open access, containing 615 rows of training data and 368 rows of test data across 13 columns, was used. The models were compared using performance evaluation criteria, and it was concluded that logistic regression performed better than the others.

Kadam et al. [10]: The study used machine learning algorithms to predict which customers' loans would be approved or rejected by financial institutions. The dataset from Dream Housing Finance Company was used. Support Vector Machines and Naïve Bayes algorithms were applied in this study. The results showed that the Naïve Bayes algorithm had the best performance.

Khan et al. [11]: The study examined and compared different models for predicting loan approval, aiming to identify the model that produced the best results with the least margin of error in determining loan approval. Data mining, statistics, and probability were used to develop the prediction model. Data from various sources were gathered to create a statistical model. As the amount of data increased, the model became more precise, reducing the error rate, thereby lowering the credit risk for banks and saving time for both customers and financial institutions. Accuracy rates were observed as 80.945% for Logistic Regression, 93.648% for Decision Tree, and 83.388% for the Random Forest (RF) algorithm. Following cross-validation, accuracy rates were 80.945% for Logistic Regression, 72.213% for Decision Tree, and 80.130% for the Random Forest algorithm. Based on these results, it was concluded that the Random Forest algorithm provided the best outcome with the least error.

Udhbav et al. [12]: In this study, a predictive modeling system was developed to help mortgage finance companies assess consumers' eligibility for home loans by evaluating eligibility criteria. Two classification models, Logistic Regression and Gradient Boosting, were used. To obtain better accuracy and performance, the Gradient Boosting model was applied to eliminate the errors that occurred in the results of Logistic Regression. In the results section of the study, when comparing Gradient Boosting and Logistic Regression, it was concluded that Gradient Boosting provided better accuracy and precision. As a result of this study, the credit approval process became faster for both consumers and mortgage finance companies.

Tütüncü et al. [13]: This study aims to identify the algorithm that best predicts whether a loan will be repaid by calculating credit risks. In this context, various algorithms were used to build models, which were then compared using different performance evaluation criteria. The dataset was obtained from Home Credit, available in Kaggle's open access. The dataset was divided into 60% for training, 20% for validation, and 20% for testing. In this study, while the KNN algorithm produced less successful results for customers who defaulted and those who did not, it was concluded that the Gradient Boosting algorithm showed the best classification performance. Additionally, when examining the ROC curve results, it was observed that the KNN algorithm had a lower success rate compared to the other algorithms.

Oral et al. [14]: In this study, the Madrid Real Estate Market dataset, available in Kaggle’s open access, was used as the dataset. 25% of the data was used for testing, and 75% for training. Machine learning algorithms included in the MATLAB program were applied to the dataset. The top 5 models with the best R^2 values were included in the study. The results showed that the methods with the best performance, in order, were Bagged Trees Ensemble, Fine Tree, Exponential GPR, Wide Neural Network, and Quadratic SVM.

Anand et al. [15]: This study aims to determine the algorithm that best predicts whether a loan will be repaid by calculating the criteria banks use to decide on granting loans, in other words, the credit behaviors of banks. The datasets are composed of consumer credit application requests collected from various websites, including Kaggle's open access. A total of 15 classification algorithms were used. The results of the top five algorithms that performed the best are provided. The study found that Extra Trees Classifier and Random Forest models produced the most accurate results, and through the comparisons, it was determined that the most effective criteria were income, work experience, and debt status. The significant impact of the credit score on loan approval decisions by banks was emphasized. Taking these factors into account, risky and problematic customers were quickly identified.

Tumuluru et al. [16]: In the study, the dataset was obtained from Kaggle's open-access resources, with 70% allocated to training and 30% to testing. A comparison of algorithm accuracy showed that Random Forest performed best with an accuracy of 81%.

Viswanatha et al. [17]: This study focuses on predicting whether individual loan applications will be approved by financial institutions using machine learning methods. The dataset used is a credit dataset available in the open access section of Kaggle. The libraries used for data processing and model building are scikit-learn, pandas, and numpy. It was observed that the best accuracy result, 83.73%, was achieved by the Naive Bayes algorithm.

Uddin et al. [18]: This study focuses on developing a community-based machine learning credit prediction system to identify reliable customers who are likely to repay loans issued by financial institutions. The dataset used in this study was obtained from Kaggle’s open access section. SMOTE (Synthetic Minority Over-sampling Technique) was applied for data preprocessing and balancing the dataset. Among the nine models used, the top three machine learning models were applied for the task. To compare the machine learning models with deep learning models, a Dense Neural Network, Long Short-Term Memory (LSTM), and Recurrent Neural Network (RNN) were employed. Performance evaluation criteria were used to assess the models. Before applying community machine learning, the best-performing models were observed to be Extra Trees, Random Forest, and KNN, in that order. The accuracy of Extra Trees was 86.64%. After applying a voting ensemble to the top three models, the accuracy increased to 87.26%. Additionally, a desktop application was developed for financial institutions, enabling them to check the credit eligibility of their customers. This interface facilitates faster credit approval processes and improves operational efficiency.

Çelik et al. [19]: This study focuses on predicting bank loans using different algorithms. The aim of this study was to make accurate predictions regarding credit risk and credit assessments. Various algorithms were compared based on different dataset sizes and features in terms of classification performance. The dataset used in this study was obtained from Kaggle's open access section. 80% of the dataset was used for training, and 20% was used for testing. According to performance evaluation metrics and the ROC curve, the best-performing algorithm was the CatBoost algorithm. Additionally, it was observed that ensemble learning algorithms performed better in predicting bank loans.

Prasad et al. [20]: The dataset used consists of historical data from the credit market. The libraries used for data processing and model building are scikit-learn, matplotlib, pandas, NumPy, and PyTorch. Various machine learning methods were used and compared using performance evaluation metrics. The best results were observed with the RF and KNN algorithms.

3. Methodology

3.1. K-Nearest Neighbor Algorithm (KNN)

The k-nearest neighbor algorithm (KNN) is a supervised learning algorithm. It is a simple algorithm that performs classification by using similarities within the dataset. KNN classifiers, while classifying, use a predetermined number, k, to check the data points that are closest to the new data based on a distance function. The most commonly used distance functions are Euclidean, Manhattan, and Minkowski. While classifying, the algorithm checks the distance data to determine which group has the most similar data, and the new data is assigned to that group. For most datasets, the optimal value for k is chosen to be between 3 and 10. In some studies, the value of k is selected as the square root of the number of rows in the dataset [8][21].

3.2. Random Forest Algorithm

RF algorithm is an ensemble learning method. It is also used for classification and regression operations [22].

Working Principle of the Random Forest Algorithm:

The data subsets used in the Random Forest algorithm are created by randomly selecting from the dataset. These subsets have the same number of rows as the original dataset and randomly selected columns [23]. Bootstrapping is the process of creating a new dataset using the original dataset. In the Random Forest algorithm, during the bootstrapping process, the same data is used for each decision tree [24]. Decision trees are then trained on these new datasets [25]. A new data row is tested on the trained decision trees, and the resulting outcomes are recorded. The aggregation of these results refers to the process where the most frequently occurring result is selected [22].

3.3. Logistic Regression

Logistic regression, one of the classification algorithms, is used to estimate the probability of a dependent variable taking two values such as 1 or 0, true or false. [26]. Logistic regression is expressed by the formula $f(z) = 1/ 1+e^{-z}$. If the value of z is −∞ the function's value is 0, and if z is +∞ the function's value is 1. Regardless of the value of z, the result of logistic regression is always between 0 and 1 [27].

3.4. Support Vector Machines (SVM)

In Support Vector Machines, which is a supervised learning algorithm, data is considered as points with coordinates in n-dimensional space. SVM performs classification by drawing a hyperplane. SVM algorithms set the distance between the points in different categories and closest to each other to be maximum and this maximum distance is called margin. The points that we call support vectors are the points that touch the margin [8].

4. Results and Discussion

In this study, the development environment used was www.kaggle.com. For each algorithm, 90% of the dataset was allocated for training and 10% for testing.

The value of k can be determined by taking the square root of the number of rows in the dataset or by selecting k between 3 and 10 [21]. In this study, knn algorithm was used and the value of k was chosen over a wide range to train the model. To observe the results over a wider range, 29 different values for the k parameter (values between 5 and 35 were tested one by one) were applied, and the accuracy rates for both datasets are shown in Figure 1 and Figure 2. As shown in the graph for the first dataset in Figure 1, the highest accuracy rate of 87.096 was achieved when k=11; for the second dataset, as shown in the graph in Figure 2, the highest accuracy rate of 92.974 was obtained when k=16.

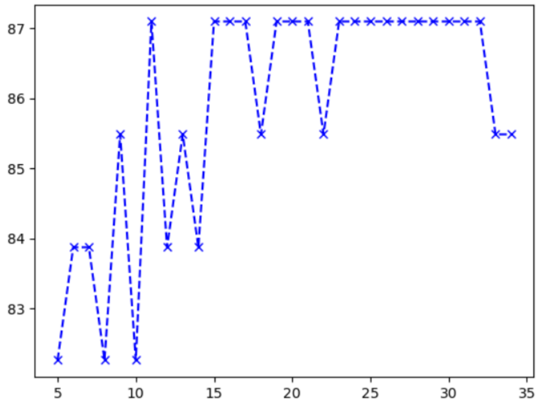


Figure 1. *Accuracy rate chart for the first dataset according to the value of k*

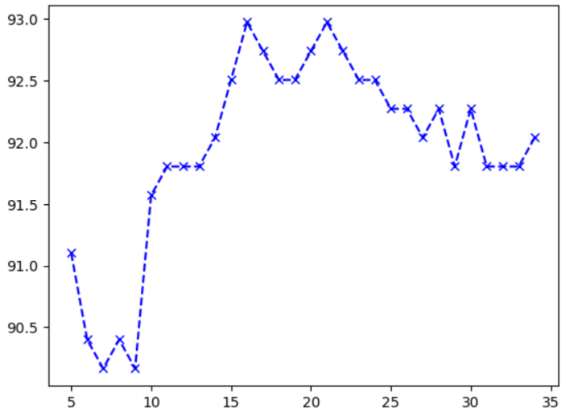


Figure 2. *Accuracy rate chart for the second dataset according to the value of k*

This study used the RF algorithm, where the number of trees was selected between 10 and 101, and the accuracy rates were calculated based on this range. As shown in Figures 3 and 4, the accuracy rates for both datasets are displayed according to the number of trees selected within the given range. For the Dataset_1, the highest accuracy rate was 87.09% when the number of trees was 10; for the Dataset_2, the highest accuracy rate was 98.82% when the number of trees was 28. The fact that the Dataset_1 reaches the highest accuracy with 10 trees indicates that the Dataset_1 is less complex compared to the Dataset_2. On the other hand, the higher accuracy of the Dataset_2 compared to the first suggests that the model has better learned the complexity of the data and made more accurate decisions with more trees.

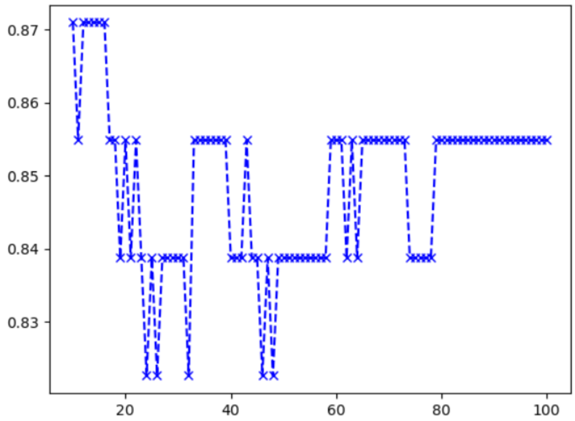


Figure 3. *Accuracy rates for the* Dataset_1 *according to the number of trees*

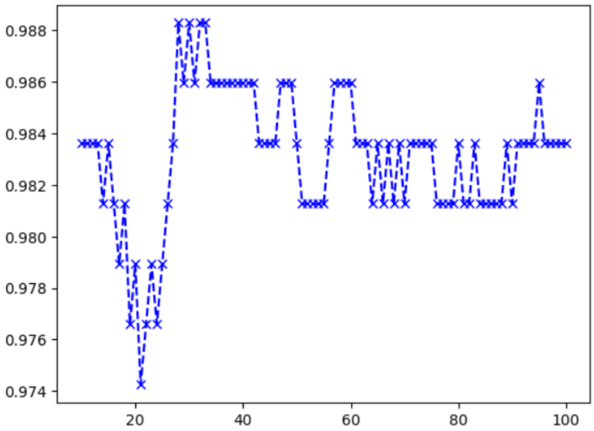
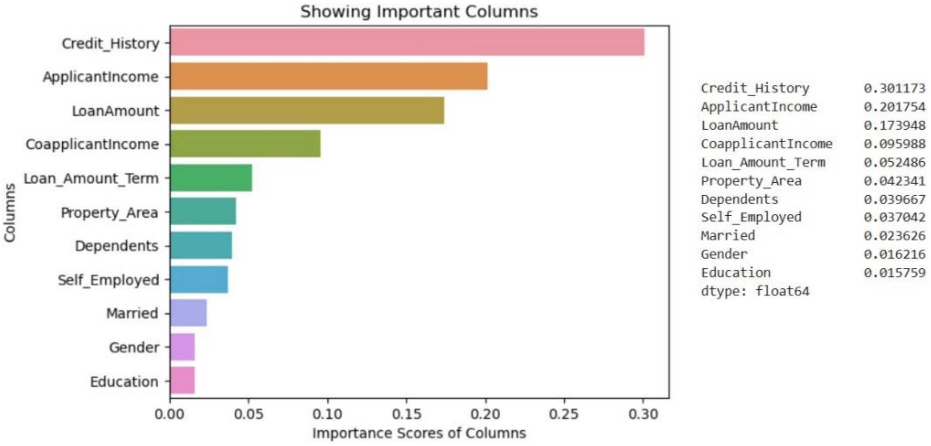


Figure 4. *Accuracy rates for the* Dataset_2 *according to the number of trees*

For both datasets, the columns are visualized in Figures 1.5 and 1.6 in order of importance. When examining the Random Forest Algorithm graphs, it is observed that the factor most influencing the credit approval process for the first dataset is credit history, while for the second dataset, it is credit score. The factor least affecting the credit approval process in both datasets is education status. The feature with the lowest importance, which ranks last, was removed for both datasets. The model was then retrained using only the features with higher importance. For the first dataset, the accuracy rate was 0.870, but after retraining the model, the accuracy rate decreased to 0.8226. For the second dataset, the accuracy rate was 0.9882, and after retraining the model, the accuracy rate increased to 0.9906.

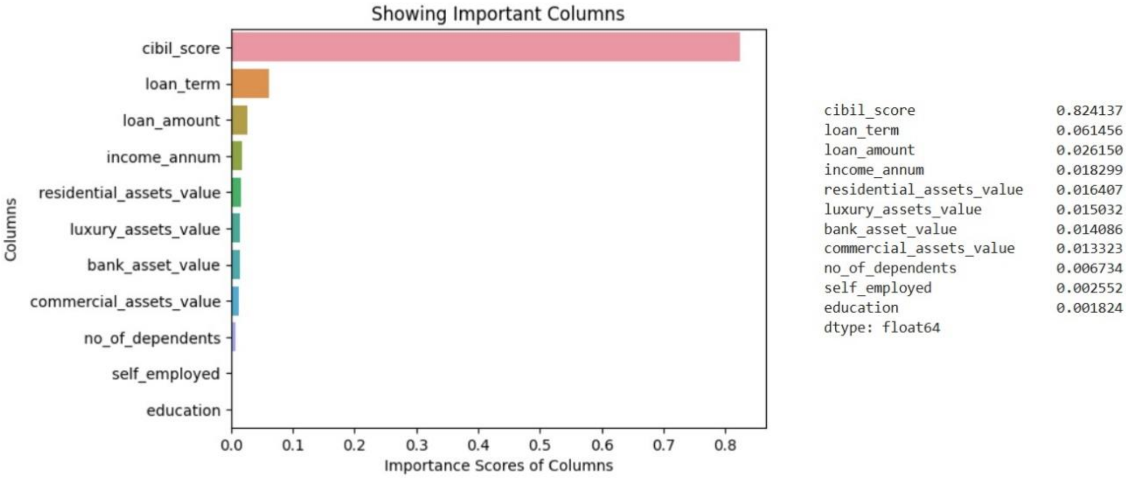
The decrease in accuracy for the first dataset indicates that the education and gender columns play a role in the model's decision-making process. It also suggests that these columns may be interrelated. For example, higher education levels generally increase the likelihood of working in higher-paying jobs and finding employment. Having a higher income is also a factor that increases the likelihood of repaying a loan. The reason gender affects the accuracy rate may be because men typically have longer periods of uninterrupted work in the labor force compared to women, who may have breaks in their careers due to cultural or biological reasons, such as childbirth and childcare. This can represent a higher risk for the lending institution, which is an undesirable situation for them.

In the second dataset, the columns for self-employment and education level are of less importance compared to other columns. Lending institutions tend to prefer customers with stable incomes. A self-employed individual may not always provide reliable information about income stability, as self-employed individuals may have either high or low incomes, and their income level can fluctuate throughout the year. Credit score, however, holds more weight in the second dataset compared to other columns. A good education level does not necessarily indicate a high income. For someone with a good education, factors such as not being able to find employment due to national conditions could also affect their credit score. Assets owned by the customer, the loan term and amount, and the credit score have a greater impact on the loan approval process, whereas education level and self-employment status, due to their lower importance, have a lesser effect on the credit decision process.



```
Credit_History       0.301173
ApplicantIncome      0.201754
LoanAmount           0.173948
CoapplicantIncome    0.095988
Loan_Amount_Term     0.052486
Property_Area        0.042341
Dependents           0.039667
Self_Employed        0.037042
Married              0.023626
Gender               0.016216
Education            0.015759
dtype: float64
```

Figure 5. *Column importance and percentage in the RF algorithm for the first dataset*



```
cibil_score                 0.824137
loan_term                   0.061456
loan_amount                 0.026150
income_annum                0.018299
residential_assets_value    0.016407
luxury_assets_value         0.015032
bank_asset_value            0.014086
commercial_assets_value     0.013323
no_of_dependents            0.006734
self_employed               0.002552
education                   0.001824
dtype: float64
```

***Figure 6**. Column importance and percentage in the RF algorithm for the second dataset*

In this study, using the Logistic Regression algorithm, the accuracy rate was found to be 87.09% for the Dataset_1 and 90.86% for the Dataset_2. Achieving an accuracy rate of 87.09% on the first dataset indicates that the model performs adequately on a small and sparse dataset. In the second dataset, the accuracy rate of 90.86% can be explained by the data being more homogeneous and within a narrower range. The second dataset, with a smaller standard deviation, allowed the model to generalize better. The organization of the data in this way helped logistic regression produce more stable and consistent results, which in turn led to an increase in accuracy.

These results reveal that one of the key factors affecting the performance of the logistic regression model is the structure of the dataset. More homogeneous and lower-variance datasets lead to higher accuracy and better results. This highlights the importance of considering the statistical characteristics of the dataset when analyzing model performance.

In this study, using the Support Vector Machine algorithm, the accuracy rate was found to be 88.7% for the Dataset_1 and 93.9% for the Dataset_2. The difference in accuracy rates is related to the fact that the two datasets have different characteristics. The size and distribution of the dataset are important factors that affect the model's learning ability and accuracy.

4.1. Performance Evaluation Criteria

Accuracy is a classification metric, but it is not sufficient on its own for evaluating classification results [28]. Therefore, in this study, in addition to the accuracy rate, other classification metrics such as precision, recall, F1-score, specificity, negative predictive value, and false discovery rate have been used.

A confusion matrix is the tabulation of actual and predicted values. Performance evaluations are made using the data in the matrix [29]. To determine the performance evaluation metrics of the classification algorithms, the confusion matrices for the algorithms on both datasets are shown in Table 1. Upon examining the table, it is observed that in Dataset_1, the highest number of TN (credit rejection) was correctly predicted by the KNN algorithm, while the highest number of TP was correctly predicted by the support vector machines. In Dataset_2, both the highest number of TN and TP were correctly predicted by the random forest algorithm.

Table 1. *Confusion matrices obtained from classification results*

Confusion matrices			Dataset_1		Dataset_2	
			Prediction			
			N	P	N	P
KNN	Actual values	N	9	6	149	10
		P	2	45	20	248
LR		N	8	7	140	19
		P	1	46	20	248
RF		N	9	6	156	3
		P	2	45	2	266
SVM		N	8	7	149	10
		P	0	47	16	252

Performance evaluations results of the classification algorithm for both datasets are shown in Table 2.

Table 2. *Performance evaluations of the classification algorithm*

Metric	Dataset_1				Dataset_2			
	KNN	RF	LR	SVM	KNN	RF	LR	SVM
Accuracy	0.870	0.870	0.870	0.887	0.929	0.988	0.908	0.939
Precision	0.882	0.882	0.868	0.870	0.961	0.989	0.929	0.962
Recall	0.957	0.957	0.979	1.000	0.925	0.993	0.925	0.940
F1 Score	0.918	0.918	0.920	0.931	0.943	0.991	0.927	0.951
Specificity	0.600	0.600	0.533	0.533	0.937	0.981	0.881	0.937
Negative Predictive Value (NPV)	0.818	0.818	0.889	1.000	0.882	0.987	0.875	0.903
False Discovery Rate (FDR)	0.118	0.118	0.132	0.130	0.039	0.011	0.071	0.038

The accuracy rate is highest in Dataset_1 with 88.70% using the SVM algorithm, and in Dataset_2 with 98.8% using the Random Forest (RF) algorithm. The accuracy rate in dataset_2 is higher compared to dataset_1. This indicates that the dataset_2 model generalizes better.

In Dataset_1, the highest precision metric is 0.882 for the Random Forest algorithm, while the highest recall metric is 1.000 for the SVM algorithm. In Dataset_2, the highest precision is 0.989 and the highest recall is 0.993 for the RF algorithm. The high precision metric for both datasets means that most of the positive predictions are correct. The precision metric reached its highest values for both datasets using the Random Forest algorithm. A high precision metric reduces credit risk. The high recall metric for both datasets means that most of the actual positive cases were correctly predicted. A high recall helps financial institutions accurately identify customers who are likely to repay their loans, thereby contributing to an increase in profitability.

The F1 score in Dataset_1 is highest at 0.931 for the SVM algorithm, and in Dataset_2, it is highest at 0.991 for the RF algorithm. A high F1 score indicates that the test performs well overall, meaning it has both a high number of true positive results and low numbers of FP and FN.

In Dataset_1, the highest specificity metric is 0.600 for the Random Forest algorithm, and in Dataset_2, it is 0.981 for the RF algorithm. The low specificity in the first dataset indicates that there is a high likelihood of customers who were not granted credit being incorrectly predicted as positive (approved for credit). The high specificity in the second dataset, on the other hand, suggests that customers with high risk can be rejected, helping to prevent financial losses. It also contributes to a more accurate assessment of customers during the credit approval process and increases trust in decision-making by financial institutions.

In Dataset_1, the highest Negative Predictive Value is 1.000 for the SVM algorithm, and in Dataset_2, it is 0.987 for the RF algorithm. The high negative prediction values in both datasets help financial institutions avoid economic losses by accurately identifying customers who should not be granted credit.

In Dataset_1, the highest False Discovery Rate is 0.132 for the Logistic regression algorithm, and in Dataset_2, it is 0.071 for the Logistic Regression (LR) algorithm. Although the rates are low in both datasets, the lower false discovery rate in the second dataset indicates that the model produces fewer false positives and provides more reliable results.

5. Conclusion

In this study, various machine learning methods were employed to predict whether a bank would approve a home loan. Two different datasets were used in the study. The reason for using two datasets was to compare the compatibility and results of the algorithms across multiple datasets and to evaluate the consistency of the results. The two datasets were not considered as conflicting or competing elements in this study, but rather as control groups to validate the work. Additionally, the goal was to observe how the varying sizes of the two datasets would affect the results. In the classification phase of the study, KNN, RF, SVM, and Logistic Regression algorithms were compared. In the study conducted on the two separate datasets, the highest accuracy rates for Dataset_1, excluding precision and specificity, were obtained with the SVM algorithm (Accuracy: 88.7% (SVM), Precision: 88.2% (RF), Recall: 100% (SVM), F1: 93.1% (SVM), Specificity: 60% (RF), NPV: 100% (SVM), FDR: 13.2% (LR)). In Dataset_2, the highest accuracy rates across all performance metrics were observed for the Random Forest and Logistic Regression algorithms (Accuracy: 98.8% (RF), Precision: 98.9% (RF), Recall: 99.3% (RF), F1: 99.1% (RF), Specificity: 98.1% (RF), NPV: 98.7% (RF), FDR: 0.071% (LR)). The results obtained from dataset_1 show that the small and sparse nature of the dataset limits the model's performance. Although SVM demonstrated high success, particularly in metrics like sensitivity and F1 score, it fell behind the RF algorithm in terms of precision and specificity. The low performance in specificity, especially, indicates that the model struggled to correctly predict the negative class. In dataset_2, however, the RF algorithm performed overall better, thanks to the larger amount of data and narrower distribution. The grouping of data points in a closer range and more homogeneous manner in this dataset improved the model's accuracy. The high precision, recall, and F1 score of the RF algorithm indicate that the model accurately classifies both classes, while the specificity and NPV rates further reinforce its overall success. These results demonstrate that larger and more homogeneous datasets increase the model's generalization capacity and allow for more accurate predictions.

The results obtained help financial institutions predict customer behavior and make quicker decisions about whether to approve a loan or not. Additionally, financial institutions can perform credit risk assessments by considering both the advantages and disadvantages of the algorithms used. Furthermore, with these results, financial institutions can increase the potential pool of customers eligible for credit.

Acknowledgement

This article is derived from the Master's thesis titled “Prediction of Home Loan Approval with Machine Learning”.

References

[1] Kösedağ, E. (2023). Türkiye’de Konut Hakkı ve Bu Hakkın Kullanılmasında Ortaya Çıkan Sorunlara Yönelik Değerlendirme. Kent Akademisi, 16(1), 595-611. https://doi.org/10.35674/kent.1220084

[2] Mevzuat Bilgi Sistemi, 1982. Türkiye Cumhuriyeti Anayasası Erişim Tarihi: 08.01.2024. https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=2709&MevzuatTur=1&MevzuatTertip=5

[3] Koyuncu, C., & Berrin, S. (2011). Takipteki Kredilerin Özel Sektöre Verilen Krediler Ve Yatırımlar Üzerindeki Etkisi. Dumlupınar Üniversitesi Sosyal Bilimler Dergisi, (31).

[4] Arun, K., Ishan, G., & Sanmeet, K. (2016). Loan approval prediction based on machine learning approach. IOSR J. Comput. Eng, 18(3), 18-21.

[5] Gautam, K., Singh, A. P., Tyagi, K., & Kumar, M. S. (2020). Loan Prediction using Decision Tree and Random Forest. International Research Journal of Engineering and Technology (IRJET), 7(08), 853-856.

[6] Aphale, A. S., & Shinde, S. R. (2020). Predict loan approval in banking system machine learning approach for cooperative banks loan approval. International Journal of Engineering Trends and Applications (IJETA), 9(8).

[7] Gupta, A., Pant, V., Kumar, S., & Bansal, P. K. (2020, December). Bank loan prediction system using machine learning. In 2020 9th International Conference System Modeling and Advancement in Research Trends (SMART) (pp. 423-426). IEEE.

[8] Ndayisenga, T. (2021). Bank loan approval prediction using machine learning techniques (Doctoral dissertation).

[9] Fati, S. M. (2021). Machine learning-based prediction model for loan status approval. Journal of Hunan University Natural Sciences, 48(10).

[10] Kadam, A. S., Nikam, S. R., Aher, A. A., Shelke, G. V., & Chandgude, A. S. (2021).Prediction for loan approval using machine learning algorithm. International Research Journal of Engineering and Technology (IRJET), 8(04).

[11] Khan, A., Bhadola, E., Kumar, A., & Singh, N. (2021). Loan approval prediction model a comparative analysis. Advances and Applications in Mathematical Sciences, 20(3), 427-435.

[12] Udhbav, M., Kumar, R., Kumar, N., Kumar, R., Vijarania, D., & Gupta, S. (2022). Prediction of Home Loan Status Eligibility using Machine Learning. Swati, Prediction of Home Loan Status Eligibility using Machine Learning (May 27, 2022).

[13] Tütüncü, T. E. (2022). Makine öğrenmesi algoritmaları ile kredi temerrüt riskini tahmin etme (Master's thesis, Bursa Uludağ Üniversitesi).

[14] Oral, M., Okatan, E., & Kırbaş, İ. (2021). Makine öğrenme yöntemleri kullanarak konut fiyat tahmini üzerine bir çalışma: Madrid örneği. Uluslararası Genç Araştırmacılar Öğrenci Kongresi, Burdur, Turkey.

[15] Anand, M., Velu, A., & Whig, P. (2022). Prediction of loan behaviour with machine learning models for secure banking. Journal of Computer Science and Engineering (JCSE), 3(1), 1-13.

[16] Tumuluru, P., Burra, L. R., Loukya, M., Bhavana, S., CSaiBaba, H. M. H., & Sunanda, N. (2022, February). Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms. In 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS) (pp. 349-353). IEEE.

[17] Viswanatha, V., Ramachandra, A. C., Vishwas, K. N., & Adithya, G. (2023). Prediction of Loan Approval in Banks Using Machine Learning Approach. International Journal of Engineering and Management Research, 13(4), 7-19.

[18] Uddin, N., Ahamed, M. K. U., Uddin, M. A., Islam, M. M., Talukder, M. A., & Aryal, S. (2023). An ensemble machine learning based bank loan approval predictions system with a smart application. International Journal of Cognitive Computing in Engineering, 4, 327-339.

[19] Çelik, E., & Gür, Ö. Banka kredisi tahmini için makine öğrenmesi algoritmalarının performans analizi: Topluluk öğrenmesi algoritmalarının üstünlüğü. Artıbilim: Adana Alparslan Türkeş Bilim ve Teknoloji Üniversitesi Fen Bilimleri Dergisi, 7(1), 1-20.

[20] Prasad, P V V S V, Nageswara Rao, P.V (2024). Loan Approval Prediction System Using Machina Learning. International Journal of Innovative Science and Research Technology, 9(4), 278-281

[21] Sendel, E. (2023). Skin lesion classification with machine learning, Makine öğrenmesi ile cilt lezyonu sınıflandırması.

[22] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[23] Peker, Özkaraca, O., & Kesimal, B. (2017). Enerji tasarruflu bina tasarımı için ısıtma ve soğutma yüklerini regresyon tabanlı makine öğrenmesi algoritmaları ilemodelleme. Bilişim Teknolojileri Dergisi, 10(4), 443-449.

[24] Sarica, A., Cerasa, A., & Quattrone, A. (2017). Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: a systematic review. Frontiers in aging neuroscience, 9, 329.

[25] Ekelik, H., & ALTAŞ, D. (2019). Dijital Reklam Verilerinden Yararlanarak Potansiyel Konut Alıcılarının Rastgele Orman Yöntemiyle Sınıflandırılması. Journal Of Research İn Economics, 3(1), 28-45.

[26] Kazan, S., & Karakoca, H. (2019). Makine öğrenmesi ile ürün kategorisi sınıflandırma. Sakarya University Journal of Computer and Information Sciences, 2(1), 18-27.

[27] Kleinbaum, D. G., Klein, M. (2010). Logistic regression: a self-learning text (Third Edition). New York: springer.

[28] Zheng, A. (2015). Evaluating Machine Learning Models, Farnham, U.K.:O'Reilly Media, Inc.

[29] Santra, A. K., & Christy, C. J. (2012). Genetic algorithm and confusion matrix for document clustering. International Journal of Computer Science Issues (IJCSI), 9(1), 322.

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

Vol. 4 (No. 2), pp. 96-110, 2024
doi: 10.54569/aair.1597930
Published online: Dec 30, 2024

Comparison of The Performances of Clustering and Dimensionality Reduction Approaches in Collaborative Filtering

Özge Tas [1,*]

[1] Cappadocia University, Cappadocia Vocational School, Department of Computer Programming

Abstract

Recommendation systems (RS) can be defined as systems that aim to offer personalized product and service recommendations to users based on users' past product preferences and similarities with other users in the system, especially in systems that provide e-commerce services. The main purpose of RS is to reveal meaningful information from large-scale data to users and to recommend systems that aim to simplify the analysis of user behaviors and product attributes. It is possible to divide the techniques used in RS into two main categories content-based and collaborative filtering (CF) according to the information they receive as input. Content-based recommendation systems focus on analyzing the attributes of items such as articles, movies or music to generate tailored recommendations. CF methods analyze user-generated scores for products and services to identify patterns and preferences. The success of CF techniques hinges on accurately identifying user similarities within large datasets. However, in CF techniques, large-scale data sets consisting of a large number of users and the scores given by users to these products are used. Consequently, identifying user similarities in such extensive datasets poses significant challenges. Two different methods are used to overcome this problem. The first method applies clustering analysis to divide the dataset into smaller subsets (user or product), followed by the application of CF techniques. In the other method, dimensionality reduction is performed on a product (object) basis using Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) methods. Up to now, many studies have been carried out using clustering analysis and variable dimensionality reduction methods Despite extensive research, a thorough comparison of clustering and dimensionality reduction methods on real-world datasets remains unexplored. This study aims to compare the performances of eleven clustering techniques of eleven clustering techniques, four of which are non-hierarchical seven of which are hierarchical clustering algorithms, and two variable dimensionality reduction techniques, consisting of SVD and PCA METHODS, in CF.

Keywords: *Recommender Systems, Collaborative Filtering, Cluster Analysis, Dimension Reduction, Big Data.*

1. Introduction

Collaborative Filtering (CF) techniques are categorized into two main types: model-based and memory-based. Model-based CF techniques are based on estimating a parametric model suitable for the training data set consisting of the scores that users give to products and predicting the scores that active users can give to products using this model. In these techniques, methods such as Bayesian Networks, Regression Analysis, Clustering Techniques, and Rule-Based and Latent Semantic Models are generally used for modeling [1-6]. Memory-based CF techniques are also divided into two main categories: user-based and object-based. User-based CF techniques assume that the best way to find products that may be of interest to the active user is to identify other users with similar interests [7]. Therefore, the first step in these methods is to identify users with whom the active user is similar. In the second step, the scores that the active user may give to the products are estimated based on the scores given to the products by the neighboring, i.e. the most similar users. Products with high predicted scores are recommended to the active user. Object-based CF techniques have a similar working principle. However, similarities between objects (products) are calculated instead of similarities between users [8]. Hybrid recommender systems integrate the strengths of content-based and CF techniques and eliminate the shortcomings arising from the individual use of these methods. Various methods, such as weighting, blending, and cascading, are employed to combine these techniques. However, hybrid systems are generally based on the use of CF techniques to score both products and their contents and to estimate the score that the active user can give to the products. Comparison of the performance of clustering and size reduction methods is necessary to improve the effectiveness of recommender systems. These comparisons help to understand under what conditions different algorithms and techniques give better results. For example, hierarchical clustering methods give better results in certain situations, while other methods such as K-Means can offer faster results [9]. Therefore, considering the advantages and disadvantages of both

*Corresponding author
E-mail address: ozge.tas@kapadokya.edu.tr

Received: 7/Dec/2024; Accepted: 28/Dec/2024.

approaches, choosing the most appropriate method will increase the success of recommendation systems. In this study, we provide an in-depth analysis of collaborative filtering techniques in recommender systems. In collaborative filtering systems, there are some disadvantages such as the high dimensionality of the data and the identification of products and users. Some techniques are used to minimize the computational risks of these disadvantages. Some of these techniques aim to reduce the number of users by clustering, while others aim to reduce the number of objects by using techniques such as Singular Value Decomposition (SVD) and Principal Component Analysis (PCA). Some of the studies on CF based on dimensionality reduction can be summarized. Dimensionality techniques play an important role, especially in large data sets. Big data is a factor affecting the performance of recommendation systems, and therefore appropriate pre-processing and size reduction methods need to be applied [10]. For example, the K-Means clustering algorithm makes datasets more manageable while also improving the accuracy of recommender systems [11]. In addition, size reduction methods help to obtain more meaningful results by reducing the noise in the dataset [12]. The researchers investigated the effect of clustering methods of K-means, SOM, and Fuzzy C-Means (BCO) on the predictive performance of CF. For this purpose, they used the MovieLens dataset. As a result of their study, they concluded that the prediction performance of the BCO clustering algorithm is better [13]. Chen et al. [14], proposed a new CF technique based on evolutionary heterogeneous clustering. To evaluate the performance of the proposed technique, raw CF (without clustering method), CF based on K Means and CF based on their proposed clustering technique were applied to MovieLens and CiaoDVD datasets and it was observed that the proposed clustering method improved the prediction performance of CF. Liao and Lee [15], proposed a new CF technique based on self-constructed clustering. Ba et al. [16], used the CF approach that combines CCA and clustering. They compared the performance of CF based on SVD and clustering, CF based on SVD and traditional CF techniques and concluded that the performance of the proposed approach is better. However, in most of the mentioned studies or studies conducted for similar purposes, both a small number of data sets and a limited number of clustering and dimensionality reduction techniques were used. This study aims to compare the performance of CF techniques based on clustering analysis and dimensionality reduction techniques on real data sets. For this purpose, 11 clustering techniques (7 hierarchical and 4 non-hierarchical), 2 dimensionality reduction techniques (SVD and PCA) and 9 real data sets were used.

2. Materials and Methods

2.1. Cluster Analysis

Cluster analysis identifies natural groupings within a distributed dataset. Cluster analysis identifies inherent groupings within a distributed dataset, aiming to maximize intra-cluster similarity while minimizing inter-cluster similarity. There are many cluster analysis techniques in the literature, and it is possible to group these techniques under different headings according to different criteria.

2.1.1. Hierarchical Clustering Analysis Methods

Hierarchical clustering methods measure the similarities between data points. Based on these measurements, similar data points are merged, while dissimilar ones are separated. As can be understood, clusters are formed in a stepwise manner in these methods. There are two types of hierarchical clustering methods in the literature: additive and partitional. In agglomerative clustering methods, all individuals in the data set are initially considered as a separate cluster. Then, the closeness or distance between individuals is calculated. At each step, clusters that are close to each other are merged and this process continues until there are no clusters to be merged according to the predetermined clustering criterion. Agglomerative clustering methods therefore have a top-down approach.

Partitioning clustering has the opposite working principle of agglomerative clustering. In this method, the entire data set is initially treated as a single cluster. Similarly, the distances between individuals within the same cluster are calculated and the distant individuals are separated. This process continues until there are no clusters to be separated according to the predetermined clustering criterion. In practice, additive clustering is much more common than partitional clustering [17]. The biggest advantage of hierarchical clustering methods is that the number of clusters is automatically determined by the algorithm and therefore the number of clusters for the data set does not need to be known in advance. However, the necessity of calculating the distances between all individuals and repeating this calculation at each step makes it difficult to use hierarchical clustering methods, especially in large-scale data sets Agglomerative hierarchical clustering processes are commonly visualized using dendrograms, which depict the hierarchical structure in a tree-like format. As an example, **Figure 1** shows the clusters obtained because of applying the agglomerative hierarchical clustering method to a data set consisting of eight data points.

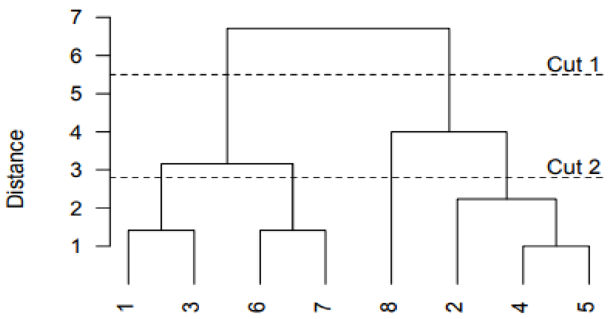


Figure 1. *An example of a dendrogram for hierarchical clustering of eight observations is given for two cuts K=2 (Cut 1) and K=4 (Cut 2) [18].*

In **Figure 1**, the values on the x-axis indicate the data points, while the values on the y-axis indicate the proximity (distance, remoteness) between the clusters. Accordingly, clusters are formed according to a predetermined distance value. For example, if the distance value is given as 2 units, the individuals that merge under 2 distance units in the dendrogram form clusters. According to **Figure 1**, when the distance is given as 2 units, the clusters obtained are Cluster 1 ={1,3}, Cluster 2 ={6,7}, Cluster 3 ={8}, Cluster 4 = {2}, Cluster 5 ={4,5}.

The standard algorithm followed by additive clustering methods is shown below. This algorithm starts with n clusters and iteratively merges clusters until only one cluster remains.

Algorithm 1: Standard algorithm for combinatorial clustering.

1. Start the algorithm with n clusters, where n is the number of individuals in the dataset.
2. Proximities between all individuals are calculated.
3. The two closest clusters are merged.
4. Steps 2 and 3 are repeated by decreasing the number of clusters by one.
5. There are seven different hierarchical clustering calculation methods based on this algorithm.

Table 1. *Calculations used in clustering analysis*

Clustering Methods	Formulas	
Single Linkage (TeB) [26]	$d_{k(i,j)} = \min\,(d_{kj}, d_{ki})$	(1)
Complete Linkage (TB) [31].	$d_{k(i,j)} = max\,(d_{kj}, d_{ki})$	(2)
Centroid Linkage (MeB) [31].	$d_{ij} = \lVert \bar{x}_i - \bar{x}_j \rVert$	(3)
Average Linkage (OB) [31].	$d_{ij} = \frac{\sum_{k=1}^{n_i}\sum_{l=1}^{n_j} d(x_{ik}, x_{jl})}{n_i * n_j}$	(4)
Median Linkage (MB) [31].	$d_{ij} = \lVert \tilde{x}_i - \tilde{x}_j \rVert_2$ $\tilde{x}_i = (1/2)(\tilde{x}_k + \tilde{x}_l)$	(5)
Weighted Average (AB) [31].	$d_{k(i,j)} = \frac{n_i}{n_i+n_j} d_{ki} + \frac{n_i}{n_i+n_j} d_{kj}$	(6)
Ward's Method (WB) [31].	$d_{ij} = \sqrt{\frac{2n_i n_j}{(n_i+n_j)}} \lVert \bar{x}_i - \bar{x}_j \rVert$	(7)

2.1.2. Non-Hierarchical Clustering Cluster Analysis Methods

Non-hierarchical clustering methods are based on starting from an initial clustering and iteratively repeating the process until the optimal cluster structure is found. In these methods, the number of clusters should be determined by the researcher in advance. In this study, non-hierarchical clustering methods such as K-Means, K-Medoid, Fuzzy C-Means and Self-Organizing Mapping (SOM) clustering methods are presented.

K-Means (KO): K-Means (KO) is a non-hierarchical clustering method widely used in many applications. The name K-Means comes from the fact that there are k clusters and the center of each cluster corresponds to the arithmetic mean of the clusters [19]. The KO clustering algorithm is based on minimizing the objective function given in Eq. 8.

$$J(X,V)=\sum_{j=1}^{k}\sum_{x\epsilon S_j}^{n}\left\|x-v_j\right\|^2 \quad (8)$$

K-Medoidler (KM): Since the KO clustering algorithm is based on the arithmetic mean, it is highly sensitive to outliers and noisy values in the data set. To alleviate this disadvantage of the KO algorithm, Kauffman and Rousseeuw [20], proposed the KM clustering algorithm. In this algorithm, the cluster centers use the center point of the regions where clusters are dense, called medoids, instead of the arithmetic mean. It shows the difference between the cluster centers of KO and KM clustering algorithms. Medoid is calculated in Eq. 9:

$$z_j=\min\left(\sum_{t=1}^{n_j}\sum_{k=1}^{n_j}||x_k-x_t||^2\right)$$
$$v_j=x_{z_j} \quad (9)$$

Fuzzy C-Averages (BCO): KM and KO clustering algorithms are based on classical logic. In other words, in these methods, an individual belongs to one and only one cluster. Therefore, these methods force an individual to belong to only one of the clusters, even if the individual is equidistant from more than one cluster. The BCO clustering algorithm is based on fuzzy logic. Therefore, BCO allows an individual to belong to multiple clusters simultaneously with different degrees of belonging. Here, membership degrees are used to determine how much individuals belong to the clusters and how much they have the characteristics of the clusters. In all clustering methods based on fuzzy logic, membership degrees have the following properties.

$$0\le u_{ij}\le 1 \quad \text{i=1,2,…,n} \quad \text{j=1,2,..,c}$$
$$\sum_{j=1}^{c}u_{ij}=1\ \forall i \quad (10)$$
$$\sum_{i=1}^{n}u_{ij}>0\ \forall j$$

Similar to KO and KM, BCO is based on the minimization of an objective function and the objective function for this algorithm is in Eq. 11:

$$J(U,V,X)=\sum_{i=1}^{n}\sum_{j=1}^{c}u_{ij}^{m}\,d_{ij}^{2}(x_i;v_j) \quad (11)$$

In the equation, m is the fuzziness index, $d_{ij}^2(x_i;v_j)$, is the Euclidean distance between individual i and cluster center j. The update equations for cluster centers and membership degrees that minimize the objective function are obtained in Eq. 12.

$$u_{ij}=\frac{1}{\sum_{k=1}^{c}\left(d_{ij}^2/d_{ik}^2\right)^{1/(m-1)}} \quad \text{i}=1,2,\ldots,\text{n j}=1,2,.\,.\,,\text{c} \quad (12)$$

$$v_j=\frac{\sum_{i=1}^{n}u_{ij}^{m}x_i}{\sum_{i=1}^{n}u_{ij}^{m}} \quad j=1,2,\ldots,c \quad (13)$$

Self-Organizing Mapping (SOM): SOM is a clustering algorithm with an architecture similar to artificial neural networks. It was proposed in 1995 by Kohonen [21]. It is therefore also known as Kohonen networks. SOM transforms high-dimensional data sets into a two-dimensional map. SOM consists of two layers, input and output. The input layer contains the number of features of the individuals to be clustered, and the output layer contains as many neurons as the number of clusters.

$$\varphi_{zk}=exp\left(\frac{||r_k-r_z||^2}{\sigma^2(t)}\right) \quad (14)$$

2.2. Size Reduction Methods

2.2.1. Principal Components Analysis

PCA is one of the linear size reduction methods based on the covariance matrix of variables. The main goal in PCA is to transform the p variable in the original data set into a smaller number of orthogonal linear components with the highest variance explanation rate, so that the relationship between the variables is eliminated [22]. As can be understood from this, the correlation between the linear components obtained at the end of PCA, also called the principal component, is zero. The size reduction with PCA can be summarized as p-dimensional $X_1, X_2, \dots, X_p$ let be the original data matrix. As mentioned earlier, the main purpose of the PCA method is to find the basic components in its shape. Mathematically, PCA is based on the spectral decomposition of the covariance matrix, which is defined in eq. 15. $TB_1, TB_2, \dots, TB_{d \ll p}$

$$\Sigma = \mathrm{A}\Lambda\mathrm{A}' \tag{15}$$

In other words, the principal components are generally calculated in eq. 16:

$$TB = A'(X - \mu) \tag{16}$$

After calculating the principal components as given in Eq. 16, the dimensionally reduced data is obtained by selecting the number of principal components that explain most of the variance [23].

2.2.2. Singular Value Decomposition

SVD is one of the most widely used dimension reduction techniques in CF. SVD is basically based on decomposing an Nxn matrix X (where n is the number of users and p is the number of products and objects) into 3 matrices in eq. 17:

$$X = U_{nxp} \lambda_{pxp} V^T{}_{pxp} \tag{17}$$

2.3. Recommendation Systems

GLCM Recommender systems are filtering systems that are used to generate information based on the behavior of users, to examine their interests and behaviors, and to predict the products they may be interested in by using the information entered online [24, 25]. There are different ways to design recommendation systems. The first and simplest of these is to provide a streaming style service by directing recommendation around the content stream. Examples of this are music services such as Spotify and Youtube music. After each item, the user is allowed to evaluate the item and the user is presented with content based on these evaluations. These reviews influence the algorithm for the next song or item. This process is repeated until the user leaves the platform. Another way is the catalog-based website method. An example of this is the Netflix website, which is one of the movie websites. It helps users to make ratings on the movie and then categorize the movie content and have information about the movies. There are usually pages dedicated to each movie with detailed movie content specifications. This collaborative filtering method is further enhanced by using algorithms to encourage users to rate any movie they see, and to generate prediction ratings of personalized features next to the movie cover image in the detailed information. These predictions help users quickly decide whether a movie is worth learning more about. As a whole, recommendation systems can be divided into 3 types, content-based recommender systems are used to recommend new objects and information that may be of interest to users by taking into account the content information of objects that have previously attracted users' attention on the Internet and the basic characteristics of users. Collaborative filtering systems usually take into account the scores that users give to products or objects in the system. The main purpose here is to examine the objects that users give high scores and suggest new objects to them. Hybrid Methods are a combination of contextual and collaborative filtering systems.

2.3.1. Collaborative Filtering Methods

The CF method is the most successful recommendation system [26]. The main purpose of the CF methods is to make suggestions and predictions to the active user based on the opinions of like-minded users. In CF, user opinions can be obtained directly or implicitly from users differently.

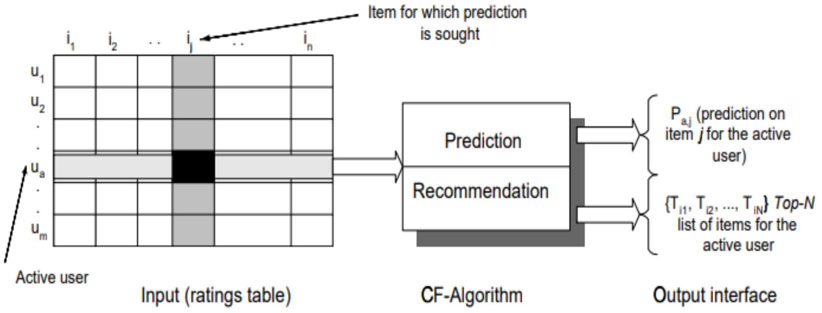

Item for which prediction is sought
i_1
i_2
. .
i_j
. .
i_n
u_1
u_2
u_a
u_m
Active user
Input (ratings table)
Prediction
Recommendation
CF-Algorithm
$P_{a,j}$ (prediction on item *j* for the active user)
$\{T_{i1}, T_{i2}, \dots, T_{iN}\}$ *Top-N* list of items for the active user
Output interface

Figure 2. *The collaborative filtering process [35].*

Collaborative filtering algorithms represent all mxn user-item data as a rating matrix. Each entered image in user set A represents the preference score of user i on item j. Researchers have developed collaborative filtering algorithms as memory-based (user-based) and model-based (item-based) algorithms [3].

There are many different methods for calculating the similarity or weight between users and items. Generally, in the similarity calculation, the number of users is considered as the size of the active user's neighborhood relations, and the similarity-based collaborative filtering method is considered as neighbor relation-based collaborative filtering.

Correlation-based similarity: In this case, the similarity $w_{u,v}$ between user u and v and the similarity $w_{i,j}$ between two items i and j are measured using Pearson's correlation or other correlation measures. Pearson correlation measures the relationship between two variables. User-based algorithm for similarity between users u and v;

$$w_{u,v} = \frac{\sum_{i\in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i\in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i\in I}(r_{v,i} - \bar{r}_v)^2}} \tag{18}$$

The item-based algorithm calculates the Pearson correlation of the degrees of the items i and j for the user set u ϵ U,

$$w_{i,j} = \frac{\sum_{u\in U}(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u\in U}(r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{u\in U}(r_{u,j} - \bar{r}_j)^2}} \tag{19}$$

Some variations of item-based and user-based Pearson correlations can be found [15]. Pearson correlation calculation is widely used in collaborative filtering.

Cosine-based similarity measures the similarity between two documents by treating each document as a vector of word frequencies and calculating the cosine of the angle formed by the frequency vectors. Generally, this type of similarity is preferred for collaborative selection based on users and items rather than on the frequencies of documents and ratings.

If R is an nxm-dimensional user item matrix, the similarity between two items i and j is calculated as the cosine of the n-dimensional vectors corresponding to items i and j in the R matrix.

Cosine similarity between element vectors i and j,

$$w_{i,j} = \cos(\vec{i},\vec{j}) = \frac{\vec{i} \bullet \vec{j}}{||\vec{i}|| * ||\vec{j}||} \tag{20}$$

Jaccard similarity is the easiest way to calculate the similarity between two users. It looks at the common items rated by both users, regardless of their ratings from the user (Charikar, 2002). Jaccard similarity is useful when items do not receive a reliable rating.

$$sim_{u,i} = |\frac{r_{u,i} \cap r_{v,i}}{r_{u,i} \cup r_{v,i}}| \tag{21}$$

The most important part of collaborative filtering is to determine the recommendation for the active user. Once the active user items are identified, the following two different equations are used to determine the user and item score.

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v\epsilon u}(r_{v,i} - \bar{r}_v).sim_{u,v}}{\sum_{v\epsilon u} sim_{u,v}}, \tag{22}$$

$$r(a,p) = {}^{1}/_{n} \sum_{i=1}^{n} r_{ip}, \tag{23}$$

3. Application and Result

The This study aims to compare the performance of collaborative filtering methods based on clustering and dimensionality reduction techniques. For this purpose, 9 real data sets were used. The data sets and their characteristics are given in **Table 2**.

Table 2. *Data sets used and their features.*

Dataset	Number of users	Number of Objects	Scoring
Anime	200	16384	1 with 10 between
BookDataset	671	150	1 with 10 between
Jester	24938	100	-10 with +10 between
Laptop	671	300	0 with 5 between
Mobile	671	183	0 with 5 between
Movie	943	1683	0 with 5 between
Restaurant 1	139	2551	0 with 2 between
Restaurant 2 (Lunch)	139	2551	0 with 2 between
Restaurant 3 (Service)	139	2551	0 with 2 between

Anime, Jester, Restaurant 1, Restaurant 2, Restaurant 3 data sets were downloaded from Link 1 [27] and BookDataset, Laptop, Mobile, Movie data sets were downloaded from Link 2 [28], Link 3 [29]. For performance comparisons, 6 different scenarios were run. For each scenario, 90% of the dataset was selected as the users in the system and 10% as the active user whose score would be calculated. From the 10% portion, the score of the products that each user rated was estimated and compared with the actual values. Three goodness-of-fit measures were used as comparison criteria: Root Mean Square Error (HKOK) in eq. 24, Mean Absolute Percentage Error (OMYH) in Eq. 25 and Mean Absolute Error (OMH) in Eq. 26:

$$HKOK = \sqrt{\frac{\sum_{i=1}^{n}(gp_i - tp_i)^2}{n}} \tag{24}$$

$$OMYH = \frac{\sum_{i=1}^{n} |gp_i - tp_i|/gp_i}{n} * 100 \tag{25}$$

$$OMH = \sum_{i=1}^{n} \frac{|gp_i - tp_i|}{n} \tag{26}$$

Comparison results are presented based on the average performance across nine datasets, followed by specific evaluations of the Jester dataset, secondly according to the Jester dataset with the highest number of users, and thirdly according to the Anime dataset with the highest number of objects.

Comparison Results for Scenario 1

In this section, we compare the performance of CF techniques based on BCO, KO, KM, SOM, WB, OB, MeB, MB, AB, TeB, TB clustering algorithms. Here, each clustering technique was run with 20, 30, 40, 50 and 60 clusters respectively. According to the average results obtained for all data sets for Scenario 1,

- The results indicate that the number of clusters does not significantly influence performance across all goodness-of-fit metrics,
- Across all scenarios, the BCO algorithm demonstrates the highest average performance,
- The performance rankings of the clustering methods are BCO, WB, TB, KO, TeB, MeB, AB, OB, SOM, KM, MB according to the HKOK criterion, BCO, WB, TeB, MB, OB, MeB, TB, KM, KO, AB, SOM according to the OMYH criterion, and BCO, WB, TeB, KO, MeB, KM, SOM, MB, TB, OB and AB according to the OMH criterion. Thus, for Scenario 1, it can be said that the WB clustering algorithm also provides better prediction results than the other methods, while SOM, AB and MB generally perform poorly.

For the Jester data set,

- When comparisons are made according to the HKOK and OMH criteria, the top three best-fit clustering algorithms are SOM, KO and BCO, respectively, and when comparisons are made according to the OMYH criterion, the top three algorithms are AB, OB and TB,
- The OLSR values are quite high for Scenario 1 and the OLSR values of the Jester dataset can significantly affect the overall performance,

- It is observed that the number of clusters does not have a significant effect on performance. For the anime dataset,
- It is seen that the CF technique based on the BCO clustering algorithm performs better than the other methods according to all three goodness of fit criteria, while the worst fit is obtained from the SOM clustering method.
- Apart from this, it can be said that the WB clustering method also performs well for the Anime dataset.

Comparison Results for Scenario 2

For the comparisons in this section, SVD dimension reduction technique was first applied to all datasets. In the next stage, users were clustered separately using 11 clustering algorithms and scores were calculated according to the clusters. According to the average goodness of fit values obtained from all datasets,

- Compared to non-hierarchical clustering methods in all three goodness-of-fit criteria, CF methods based on hierarchical clustering methods have a higher prediction success,
- The highest success was obtained from the WB clustering method and the worst success was obtained from the SOM clustering method,
- Among the non-hierarchical clustering methods, BCO and KM are the most successful algorithms,
- It is observed that the number of clusters does not significantly change the prediction success.

SVD feature extraction method for the Jester dataset,

- The best fit is when WB is run with the HKOK and OMH criteria and TB with the OMYH criterion in combination with the clustering method,
- The worst prediction performance was achieved when run with the KO clustering method according to all three criteria.
- In general, hierarchical clustering methods perform better for the Jester dataset.
- The number of clusters did not have a positive effect on performance.

For the anime dataset, the best predictions according to all goodness-of-fit criteria were obtained by running SVD with the WB clustering method, while the worst predictions were obtained by running it with the KM clustering method.

Comparison Results for Scenario 3

The results in this section include the goodness of fit values obtained by first applying the PCA dimensionality reduction technique to all data sets, and then applying the 11 clustering methods to the reduced dimensionality data sets. When run with the PCA dimensionality reduction technique,

- According to all three goodness of fit criteria, the WB clustering technique provides the best estimation performance,
- The worst results are obtained from the BCO clustering method according to the HKOK and OMH criteria, and from the KM clustering method according to the OMYH criteria,
- The estimation results obtained from the hierarchical clustering methods are closer to the real values,
- It can be seen that there is no relationship between the number of clusters and performance.

According to the CF method, which was performed by first applying PCA and then cluster analysis methods to the Jester data set,

- When HKOK and OMH criteria are taken into consideration, the best estimation result is obtained from the WB clustering method, and according to the OMYH criterion, from the AB clustering method.
- The worst estimation result is obtained from the SOM clustering method according to all criteria.

For the anime dataset,

- It can be seen that the prediction performance of hierarchical clustering methods is quite good compared to non-hierarchical clustering methods according to all goodness of fit criteria,
- The WB clustering method provides the best prediction results,
- The worst prediction results are obtained from the KO clustering algorithm according to all criteria.

Comparison Results for Scenario 4

This section gives the estimation results obtained when 11 clustering algorithms are applied to the raw data and the score estimation is made according to Jaccard similarity.

- According to HKOK and OMH criteria, the best point estimate on average was obtained when the KO clustering algorithm was applied to the data sets, and according to OMYH criteria, AB clustering algorithm was applied.
- It is seen that BCO algorithm behaves differently compared to other clustering methods and has the worst estimation results.
- The performance of all algorithms except BCO and SOM improved as the number of clusters increased.
- WB clustering method showed good performance in Scenario 4 of CF estimation.

For the Jester dataset,

- According to the HKOK and OMH criteria, the best estimation results were obtained from SOM and KO, and according to the OMYH criteria, AB and MB clustering algorithms.
- According to the HKOK and OMH criteria, BCO and according to the OMYH criterion, SOM provided the worst performance.

For the anime dataset,

- The performance of hierarchical clustering methods is better,
- Of the non-hierarchical clustering methods, the BCO algorithm appears to provide the worst prediction performance for all criteria.

Comparison Results for Scenario 5

Scenario 5 is based on the combined use of the SVD size reduction technique, clustering algorithms, and the score calculation given by the Jaccard similarity. According to the results of Scenario 5,

- Hierarchical clustering methods are more successful,
- According to all three criteria, the SOM clustering algorithm gives the worst prediction results,
- The WB algorithm, which is one of the hierarchical clustering methods, provides the best performance according to all criteria,
- In hierarchical clustering methods, it can be seen that performance improves as the number of clusters increases.

As a result of applying Scenario 5 to the Jester dataset,

- On average, the best estimation results were obtained from the WB algorithm according to the HKOK and OMH criteria, and from the AB algorithm according to the OMYH criteria.
- The worst performance was obtained from the SOM algorithm according to all criteria.

As a result of the application of Scenario 5 to the anime dataset,

- The best estimation results were obtained from the MeB clustering method according to all criteria, and the worst performance was obtained from the SOM clustering method.
- Apart from this, the success of hierarchical clustering methods is quite high compared to non-hierarchical methods.

Comparison Results for Scenario 6

In this section, the results obtained as a result of using PCA as a size reduction method and Jaccard similarity as a score estimation are included. Scenario 6 for average results

- Hierarchical clustering methods are more successful in prediction,
- That the worst predictions are derived from the SOM algorithm
- For non-hierarchical clustering methods, the most successful method is KM,
- It can be seen that there is no significant relationship between the number of clusters and performance.

Results on the Jester Dataset The results obtained as a result of the application of Scenario 6 to the Jester dataset are as follows. Hierarchical clustering methods have produced prediction values that are closer to reality. BCO and SOM showed worse predictive performance than other methods. Among the hierarchical clustering methods, the best estimation success was obtained from the WB method according to the HKOC and OMH criteria, and from the MB according to the OMYH criteria.

For the anime dataset,

- Similar to the general mean and Jester data, the best estimates are obtained from the CF technique based on hierarchical clustering,
- Among the non-hierarchical clustering methods, BCO has the worst performance and SOM has the

best performance,

Overall, BCO has the worst predictive success in almost all cluster numbers.

3.1. Comparison of Overall Performance of Clustering Methods and Size Reduction Techniques

In this section, firstly, the CF performances of clustering methods for 9 data sets, 6 different scenarios, and a total of 54 different situations were compared. **Figure 3** shows the average goodness of fit values by averaging 54 different conditions. When **Table 3** and **Figure 3** are examined, it can be concluded to say that the overall prediction performance of hierarchical clustering methods is better compared to non-hierarchical clustering methods. Apart from this, in **Table 3** and **Figure 3**, the best prediction results are obtained from the WB clustering algorithm on average, and the worst performance is obtained from the SOM algorithm according to the HKOK and OMYH criteria, and the KM algorithm according to the OMH criteria. Looking at the box-plot graphs, it is seen that the number of clusters does not have a significant effect on performance.

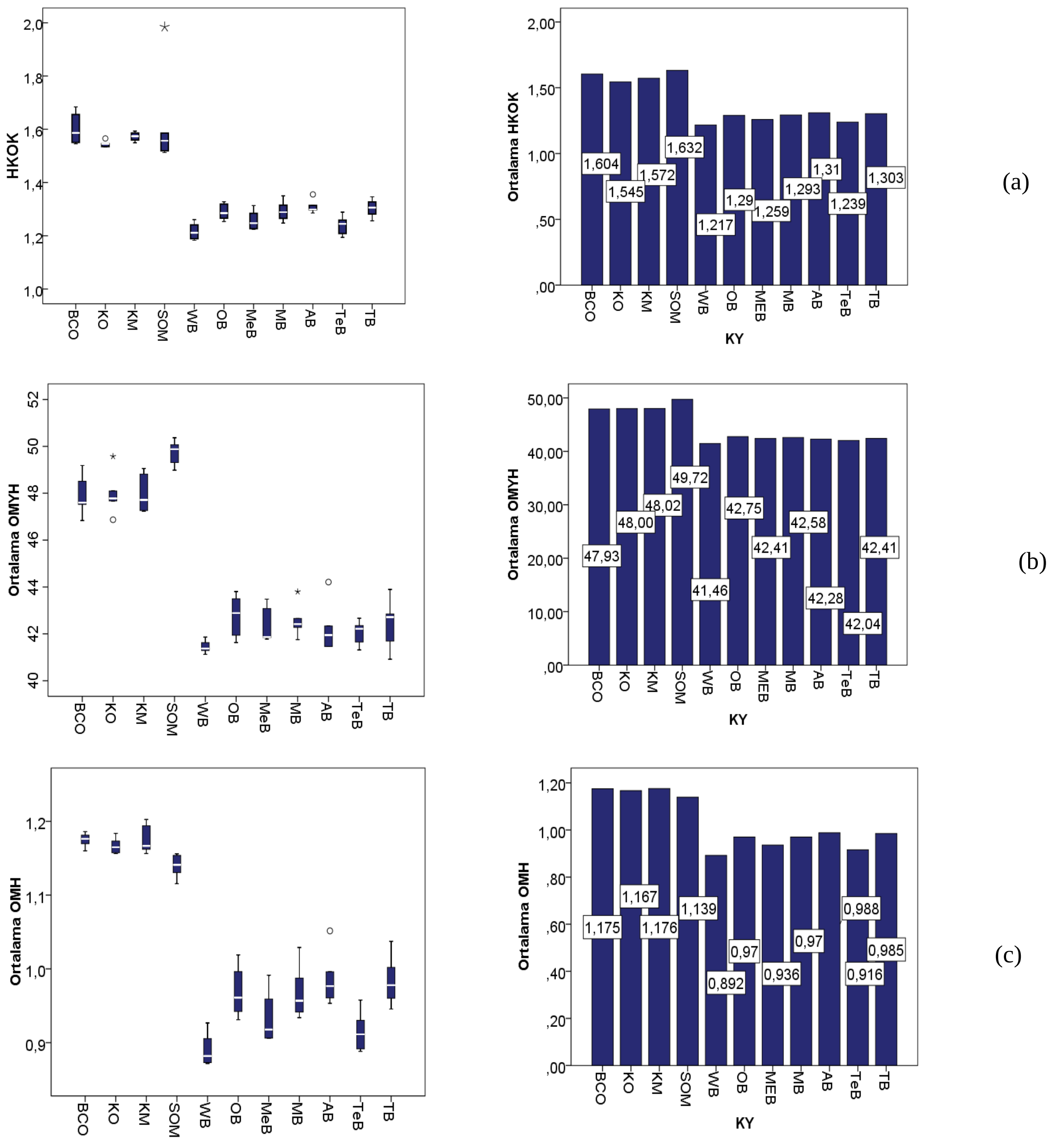


Figure 3. *Overall mean values of goodness of fit measures (a) Average HKOK (b) Average OMYH (c) Average OMH*

Table 3. *Overall average of goodness of fit measures for size reduction methods*

	HKOK		OMYH		OMH	
CN	**SVD**	**PCA**	**SVD**	**PCA**	**SVD**	**PCA**
20	1.430	1.475	45.720	47.930	1.061	1.121
30	1.408	1.471	45.080	46.939	1.037	1.107
40	1.375	1.499	44.521	47.061	1.015	1.093
50	1.375	1.441	44.792	47.321	1.011	1.091
60	1.366	1.564	44.221	46.784	1.005	1.095
Mean	**1.391**	**1.490**	**44.867**	**47.210**	**1.026**	**1.101**

Looking at the chart, it is seen that the values of the goodness of fit criteria of the SVD for all cluster numbers are lower than the PCA. From this, it is possible to say that the prediction performance of the SVD size reduction technique is better on average.

3.2. Average Comparison Tests

In this section, the non-parametric Wilcoxon test was performed to test whether the difference between the performances of the scenarios was statistically significant. **Table 4** shows the test results according to the HKOK criterion.

Table 4. *Wilcoxon Test significance values according to the HKOK criterion*

	Scenario1	**Scenario2**	**Scenario3**	**Scenario4**	**Scenario5**	**Scenario6**
Scenario1	-	0.657	0.594	**0.003**	**0.010**	0.328
Scenario2	-	-	**0.003**	**0.003**	**0.006**	**0.006**
Scenario3	-	-	-	**0.003**	**0.003**	**0.004**
Scenario4	-	-	-	-	**0.010**	**0.003**
Scenario5	-	-	-	-	-	0.197
Scenario6	-	-	-	-	-	-

Important conclusions that can be drawn from **Table 4** are as follows. Since the significance values were greater than 0.05, there was no statistically significant difference between Scenario 1-2, Scenario 1-3, Scenario 1-6, Scenario 5-6. There was no significant difference between the performances of the scenarios using Eq. 22 for score estimation. From this, it is possible to say that using raw data or reduced data does not have a significant effect on performance in scenarios where Eq. 22 is used for score estimation. If Eq. 23 is used for score estimation, there is a significant difference between the performances of the raw data and the use of data with reduced size with TKA and the use of raw data and PCA with reduced size.

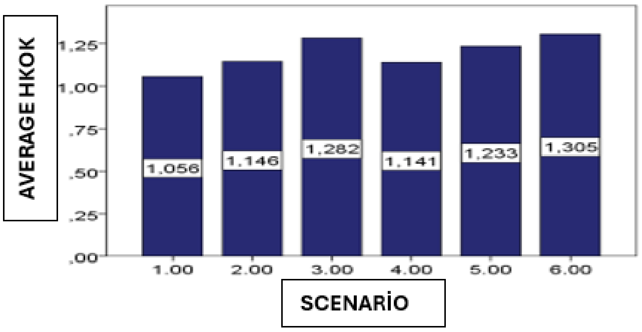


Figure 4. *Average HKOK values according to scenarios*

As can be seen from **Figure 4**, the best predictive performance on average for 9 data sets was obtained from Scenario 1, where clustering algorithms were applied to the raw data and the score estimation was made according to Eq. 22. The worst performance was obtained from Scenario 6, which corresponds to the PCA size reduction technique and its use with Eq. 23 for score estimation.

Table 5. *Wilcoxon Test significance values according to OMYH criteria*

	Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6
Scenario1	-	0.657	**0.004**	**0.003**	**0.003**	**0.026**
Scenario2	-	-	**0.003**	**0.003**	**0.003**	**0.003**
Scenario3	-	-	-	**0.003**	**0.003**	**0.003**
Scenario4	-	-	-	-	**0.003**	**0.003**
Scenario5	-	-	-	-	-	0.534
Scenario6	-	-	-	-	-	-

According to **Table 5**, the scenario binaries whose significance values are greater than 0.05 and therefore there is no significant difference between their performances are Scenario 1-Scenario 2, Scenario 5-Scenario 6. From this point of view, it is possible to say that there is no significant difference between the OMYH values obtained as a result of the use of raw data and the use of data reduced in size with SVD in cases where Eq. 22 is used for score estimation, and between the OMYH criteria obtained as a result of the use of data reduced in size with SVD and reduced in size with PCA in cases where Eq. 23 is used for score estimation. According to the OMYH criterion, the difference between all other scenario pairs is statistically significant. **Figure 5** shows the average OMIC criteria for all scenarios.

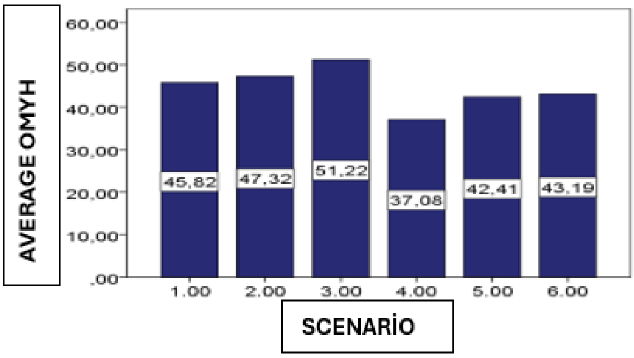


Figure 5. *Average OMYH values according to scenarios*

Looking at **Figure 5**, the best estimation results according to the OMYH criterion are obtained from Scenario 4. The worst performance was obtained from Scenario 3. Finally, in this section, Wilcoxon test results according to the OMH criterion are given in **Table 6**.

Table 6. *Wilcoxon Test intelligibility values according to OMH criteria*

	Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6
Scenario1	-	0.657	0.656	**0.004**	**0.006**	**0.013**
Scenario2	-	-	**0.003**	**0.003**	**0.003**	**0.003**
Scenario3	-	-	-	**0.003**	**0.003**	**0.003**
Scenario4	-	-	-	-	**0.009**	**0.010**
Scenario5	-	-	-	-	-	0.154
Scenario6	-	-	-	-	-	-

According to **Table 6**, the scenario pairs that did not have a significant difference between their performances according to the OMH criterion were determined as Scenario 1-Scenario 2, Scenario 1-Scenario 3, Scenario 5-Scenario 6. The difference between all other scenario pairs was found to be statistically significant.

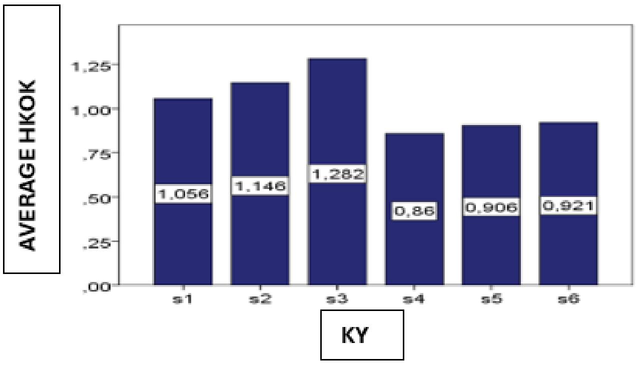


Figure 6. *Average OMH values according to scenarios*

In **Table 6**, it is seen that the scenario with the highest performance according to the OMH criterion is Scenario 4, the performance of the scenarios using Eq. 10 and Eq. 16 for score estimation is higher, and the scenario with the worst performance is scenario 3.

4. Conclusion

CF is a popular recommendation algorithm that uses the ratings or ratings of users in the system for the prediction of products that the active user might like. The main purpose of this technique is to identify the users who behave similarly to the active user among the users in the system in the most accurate way. Various approaches are used for this purpose. The basic steps of the most popular of these approaches are as follows. In the first step, similarities between the active user and all users in the system are calculated using similarity measures such as Pearson correlation coefficient, cosine similarity, and adjusted cosine similarity. In the second step, the similarities are sorted from largest to smallest, and k predetermined number of users are selected that are most similar to the active user. In the last step, it is tried to estimate the score that the active user can give to that product by using the scores or degrees given by the most similar users to the product to be predicted. Here, if it is predicted that the user will give a high rating to the product, the product is recommended to the user, otherwise it is not recommended. However, such an approach requires a high computational cost if the number of users or the number of products is high. For this reason, clustering analysis techniques are used to reduce the number of users in user-based CF techniques and the number of objects in object-oriented CF techniques. Similarly, size reduction techniques are used to reduce the number of objects in user-based CF techniques and the number of users in object-oriented CF techniques.

In approaches based on clustering, first of all, the data set is divided into c number of clusters using various methods or algorithms. In the next step, it is first determined which cluster the active user is closest to. The next step is to determine the k users that are the most similar from the cluster they are closest to and the score is calculated based on these users. The main goal of this approach is to reduce the number of users for whom similarity will be calculated.

The size reduction process, on the other hand, aims to reduce the number of objects and thus reduce the number of terms in the similarity calculation if we are talking about the user-based CF technique.

So far, various cluster analysis and size reduction techniques have been used for size reduction. However, there has not been a comprehensive study to compare the performance of these methods. The main purpose of this study is to compare the performances of the most popular 11 clustering algorithms and 2 dimensional reduction techniques using 9 datasets with different user and product numbers. For this purpose, 6 different scenarios were carried out.

As a result of the Wilcoxon tests carried out in order to determine whether there is a significant difference between the performances of the scenarios;

- A statistically significant difference was found in all scenario pairs except Scenario 1-Scenario2, Scenario1-Scenario3, Scenario5-Scenario6, Scenario1-Scenario6 according to the HKOK criterion. When the HKOC averages were examined, it was seen that the best performance was obtained from Scenario 1 and the worst performance was obtained from Scenario 6.
- When the scenarios were compared according to the OMYH criterion, the difference between all scenario pairs except Scenario1-Scenario2 and Scenario5-Scenario6 was found to be significant. When the averages were examined, it was seen that the best performance was obtained from Scenario 4 and the worst performance from Scenario 3.
- Finally, when the comparisons were made according to the OMH criterion in this section, the

difference between the performances of all scenario pairs except Scenario1-Scenario2, Scenario1-Scenario3, Scenario5-Scenario6 was found to be significant, similar to the HKOK criterion. When the averages were examined, it was determined that the scenario that provided the best performance was Scenario 4 and the scenario that provided the worst performance was Scenario 3.

The key findings of the study can be summarized as follows.
When the averages of the CF performances of the clustering methods for 9 data sets, 6 different scenarios, and a total of 54 different situations were compared, it was found that the top three most successful clustering methods were WB, TeB, MeB according to the HKOK and OMH criteria, and WB, TeB and AB according to the OMYH criteria. Hierarchical clustering methods consistently outperform other techniques in CF. Among dimensionality reduction techniques, SVD outperformed PCA, particularly in scenarios involving high-dimensional datasets.In the light of all this information, it was concluded that the performance of CF techniques was better than the use of Eq. 21, Scenario 4, SVD size reduction technique, WB and TeB clustering methods for score estimation

Declaration of interest

This study is derived from the thesis and there is no conflict of interest

Acknowledgements

The author would like to thank Nevin Guler Dincer for her contributions

Nomenclature

Abbreviations

RS	Recommendation Systems
CA	Cluster Analysis
CF	Collaborative Filtering
SOM	Self-Organizing Mapping
SVD	Singular Value Decomposition
PCA	Principal Component Analysis
TeB	Single Connection
TB	Complete Connectivity
MeB	Central Connectivity
OB	Average Connection
MB	Median Connection
AB	Weighted Connection
WB	Ward Link
KO	K-Averages
KM	K-Medoids
BCO	Fuzzy C-Averages
HKOK	Square root of the mean of squared error
OMYH	Average Absolute Percentage Error
OMH	Mean Absolute Error

References

[1] Cai, D., Wang, X., & He, X. (2009, June). Probabilistic dyadic data analysis with local and global consistency. In *Proceedings of the 26th annual international conference on machine learning* (pp. 105 112).

[2] George T, Merugu S., (2005), A scalable collaborative filtering framework based on co-clustering. In Proc. the 5th IEEE Int. Conf. Data Mining, Nov. pp.625-628.

[3] Hastie,T ,R.Tibshirani and J. Friedman (2009). The Elements Of Statistical Learning: datamining, inference and prediction (2 ed.). Springer, pp 745.

[4] Heckerman D., Chickering D., Meek C., Rounthwaite R. and Kadie C., (2001) Dependency networks for inference, collaborative filtering, and data visualization. The Journal of Machine Learning Research, 1:49–75.

[5] MacQueen, J. B., (1967), Some Methods for Classification and Analysis of Multivariate Observations, Proc. Symp. Math. Statist. and Probability (5th), 281– 297.

[6] Şenol, A., Kaya, M. ve Canbay, Y. (2024). Akan veri kümeleme probleminde ağaç veri yapılarının performans karşılaştırması. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi , 39 (1), 217-232.

[7] Groth, D., Hartmann, S., Klie, S. ve Selbig, J. (2013). Başlıca Bileşenler analizi. Hesaplamalı Toksikoloji: Cilt II, 527-547.

[8] Bakır, Ç., & Albayrak, S. (2014, April). User based and item based collaborative filtering with temporal dynamics. In *2014 22nd Signal Processing and Communications Applications Conference (Siu)* (pp. 252-255). IEEE.

[9] Sarwar B., Karypis G., Konstan J. and Riedl J., (2001) Item-based Collaborative Filtering Recommendation Algorithms. In Proceedings of the 10th International Conference on World Wide Web (WWW '01). ACM, 285– 295. DOI:http://dx.doi.org/10.1145/371920.372071.

[10] Şenol, A., Kaya, M. ve Canbay, Y. (2024). Akan veri kümeleme probleminde ağaç veri yapılarının performans karşılaştırması. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi , 39 (1), 217-232.

[11] Xu R,Wunsch D., (2005) . Survey Of Clustering Algorithms, IEEE Transactionson Neural Networks, 16(3):645–678.

[12] Altinisik, A., Yildirim, U., & Topcu, Y. I. (2022). Evaluation of failure risks for manual tightening operations in automotive assembly lines. *Assembly Automation*, *42*(5), 653-676.

[13] Koohi, H., Kiani, K. (2016), User based collaborative filtering using fuzzy c-means, Measurement, 91:134-139.

[14] Chen, J., Wang, H., & Yan, Z. (2018). Evolutionary heterogeneous clustering for rating prediction based on user collaborative filtering. *Swarm and Evolutionary Computation*, *38*, 35-41.

[15] Liao, C.L., Lee, S.J. (2016) A clustering based approach to improving the efficiency of collaborative filtering recommendation, Electronic Commerce Research and Applications,18:1-9.

[16] Ba, J. ve Frey, B. (2013). Derin sinir ağlarını eğitmek için uyarlanabilir bırakma. *Sinirsel bilgi işleme sistemlerindeki gelişmeler* , *26* .Chicago

[17] Hastie,T ,R.Tibshirani and J. Friedman (2009). The Elements Of Statistical Learning: datamining, inference and prediction (2 ed.). Springer, pp 745.

[18] Roelofsen, P. (2018), Time Series Clustering, Master Thesis, Vrıje Unıversıteıt, Amsterdam, 83s.

[19] MacQueen, J. B., (1967), Some Methods for Classification and Analysis of Multivariate Observations, Proc. Symp. Math. Statist. and Probability (5th), 281– 297.

[20] Kaufman, L. ve Rousseeuw, PJ (2009). *Verilerde grupları bulma: kümeleme analizine giriş* . John Wiley & Sons.

[21] Kohonen T. (1995) Learning Vector Quantization. In: Self-Organizing Maps. Springer Series in Information Sciences, vol 30. Springer, Berlin, Heidelberg pp 175-189.

[22] Groth, D., Hartmann, S., Klie, S. ve Selbig, J. (2013). Başlıca Bileşenler analizi. Hesaplamalı Toksikoloji: Cilt II, 527-547.

[23] X. Zhang, D. Rajan, and B. Story, "Concrete crack detection using context-aware deep semantic segmentation network," *Computer-Aided Civil and Infrastructure Engineering*, 34(11) (2019) 951–971; https://doi.org/10.1111/mice.12477.

[24] Konstan, J.A., Riedl, J. (2012) Recommender systems: from algorithms to user experience , Adapt Interact 22: 101–23 .

[25] Pan, C., Li. W. (2010) Research paper recommendation with topic analysis. In Computer Design and Applications IEEE 4, pp V4-264 .

[26] Konstan J.A., Miller B.N., Maltz D., Herlocker J.L., Gordon L.R., Riedl J., (1997), Applying collaborative filtering to Usenet news.Commun ACM; 40(3):77-87.

[27] Link 1 , (https://www.kaggle.com/datasets) , (Jester Collaborative Filtering Dataset) , (Restoran_tavsiye_sistemi) , (Recommendation System (CF) | Anime),01.08.2023

[28] Link 2, https://github.com/Ramakrishna05/Recommendation-Algorithm, 01.08.2023.

[29] Link 3, Web: https://bookdown.org/egarpor/PM-UC3M/lm-ii-dimred.html

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

Vol. 4 (No. 2), pp. 111-116, 2024
doi: 10.54569/aair.1601399
Published online: Dec 30, 2024

Performance Analysis Using CNN for Detecting Wood Knots

Nurşah Baş Uslu [1,*], Mevlüt Ersoy [2]

[1] Süleyman Demirel University, Graduate School of Natural and Applied Sciences, Department of Computer Engineering
[2] Süleyman Demirel University, Engineering Faculty, Department of Computer Engineering

Abstract

This study proposes a Convolutional Neural Network (CNN) model to quickly and accurately detect wood deformations. The performance of the CNN was enhanced by extracting structural deformation features, optimizing training parameters, and improving datasets. Experimental analyses demonstrate that the CNN achieved high accuracy rates and is an effective method for deformation detection. The CNN model was designed to identify various wood deformations. Its layered architecture was optimized to analyze deformations at different scales and levels of detail. Minimal preprocessing was applied to the images used during training, and data augmentation techniques were employed to enhance dataset diversity. The model was trained on a training dataset and tested on a validation dataset. Metrics such as loss function and accuracy were monitored throughout the training process. The CNN achieved an accuracy rate of 99.90% on the training dataset. This study highlights that the CNN model is an effective method for non-destructive detection of wood deformations. The proposed CNN model has potential applications in wood deformation detection and quality control processes.

Keywords: *Wood Deformation; Deep Learning; Convolutional Neural Network; Performance Analysis*

1. Introduction

Trees exhibit diverse characteristics due to their growth in varying natural environments. These characteristics arise from the development of branch-stem junctions throughout the tree's lifecycle. As a result, the structural properties of the tree may differ between the trunk and root wood, which significantly influences the quality of lumber utilized for industrial purposes. Among the key factors determining this quality is the formation of knots within the wood [1]. Knot formation adversely impacts the mechanical strength and performance of wood products intended for industrial applications.

The transformation of knots into sub-products is carried out by evaluating their location, type, size, and quantity within a specified length. However, the process of assessing these attributes to create sub-products imposes additional costs on factories and may lead to the production of non-standard items [2] . Knots are classified into circular, oval, and wing types based on their shapes. Furthermore, they are categorized by size as bird’s eye, small, medium, large, and very large [3].

In lumber factories, various processes can be employed to detect knots. Factories conduct knot removal operations by adhering to standardized rules for quality assessment. These operations involve determining different cutting points to remove knots from the lumber. Current systems are primarily based on the manual marking of knot locations with chalk by workers, followed by processing the lumber on machinery [4]. The involvement of human factors in identifying knot locations during the removal process can lead to errors, which, in turn, result in defective sub-products. Therefore, accurately detecting and identifying knot locations is crucial to minimizing errors and ensuring product quality.

The literature includes numerous studies on the detection of knots in lumber. These studies often employ computer vision systems in conjunction with machine learning algorithms [5-7]. A general review of these works reveals that most of the research has been conducted on static images [8]. Typically, the studies involve identifying features on the wood surface, followed by the use of classification methods to detect defects. Libraries such as YOLO [9], OpenCV [10] and TensorFlow [11] are commonly utilized for defect detection. For classification of the defects, artificial intelligence methods such as SVM [12] , KNN[13], ANN[14], CNN [15] and R-CNN [16] are employed. Results obtained from these studies demonstrate high levels of success in classification tasks.

In the product lifecycle from production to the consumer, identifying defective products holds significant importance for both manufacturers and consumers. Increasingly, the detection of such defects is being carried out by machines rather than humans. This shift is driven by the desire to reduce human labor, ensure a consistent operational structure, and minimize costs while maintaining continuous operation. In defect detection using machines, methods such as structural analysis of the product, shape and type recognition, electrical current-based detection, and pressure-based detection are commonly employed. In addition to these methods, image processing has emerged as another effective approach for defect detection.

*Corresponding author
E-mail address: nursahbas8@gmail.com

Received: 14/Dec/2024; Accepted: 28/Dec/2024.

In this study, texture feature descriptors were used to extract features from wood knot images for the classification of wood knots. The extracted features were learned using a Convolutional Neural Network (CNN) classifier to build classification models, and their performances were compared with statistical models. This paper provides a comparison between texture features and local features, as well as an analysis of the classification performances produced by the constructed models. The depth of the CNN was set to 64 layers. The results showed an accuracy rate exceeding 90%.

2. Material and Method

2.1. Dataset

The experimental dataset consists of 4,000 images obtained using high-resolution color cameras capable of capturing 1,024-line blocks and includes 8 different types of wood surface defects. The images have been resized to a resolution of 2800x1024, forming the dataset[17]. Due to the distinction between first-grade and second-grade wood in the forestry industry, the images have been divided into two classes: knotty and knotless.

Appropriate datasets are required to train or evaluate the performance of the algorithm. However, due to the structural properties of wood, the color of wood knots is generally darker compared to the surrounding wood. In some cases, however, heartwood can be darker than the knots. This situation may cause the neural network to misidentify wood knot defects and negatively affect the correct recognition of knot defects on heartwood during network training. To prevent this issue, images of knot defects on dark-colored heartwood underwent preprocessing steps. Similar images were removed from the dataset. A total of 3,000 wood knot images were used as experimental samples. The wood knot dataset consists of two classes, with 80% used as the training set and 20% as the test set, containing 2,400 training images and 600 test images, respectively (Table 1). Training on the data was performed on a computer with an Intel Xeon E5-1603 v3 processor and an NVIDIA Quadro K2200 4GB graphics card.

Table 1. *Number of Dataset*

Wood Knot State	Training Set	Test Set
Knoted	478	152
Knotless	1922	448

For each image, we created a JPG file representing the semantic map of labeled defects. During the labeling process, knotty regions in the displayed image were manually identified. These images were divided into two categories—knotted and knotless—and training was conducted using these two separate datasets. Within these images, various types and sizes of knots are present, which further contribute to enhancing the training quality. Figure 1 contains sample images from the dataset. These images consist of knot defects of varying sizes and under different lighting conditions.

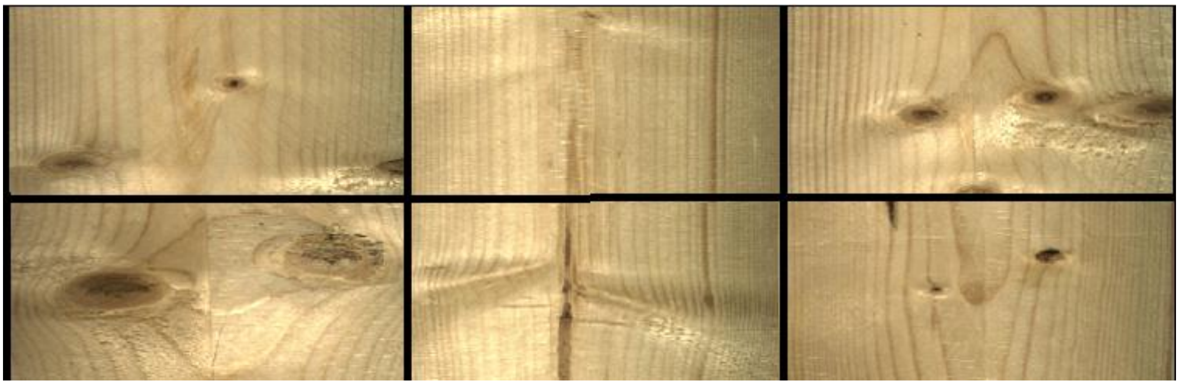


Figure 1. *Different knot images.*

2.2. Convolution Neural Network

The rapid development of computer technology and improvements in hardware performance have led to significant advancements in the field of deep learning. Artificial neural networks are widely used in various fields due to their outstanding success in areas such as image classification and recognition [18]. Convolutional Neural Networks (CNNs) are complex, multi-layered feedforward neural networks with strong fault tolerance and self-learning capabilities. They can effectively handle challenging environmental conditions and complex backgrounds. Their generalization ability is significantly superior to other methods.

CNNs typically consist of an input layer, multiple convolutional layers, pooling layers, fully connected layers, and an output layer. They support both supervised and unsupervised learning and are utilized in many domains such as computer vision and natural language processing. Moreover, CNNs are structures with parallel processing capabilities. By processing image tasks in parallel, CNNs can increase processing speed. Particularly when combined with hardware that has parallel processing capabilities, such as Graphics Processing Units (GPUs), CNN algorithms can process image data quickly and efficiently [19].

Several linked layers and convolutional blocks, such as convolutions, batch normalization, activation, ReLU, pooling, Max pooling, average pooling, completely connected, etc., make up the CNN architecture, as illustrated in Figure 2.

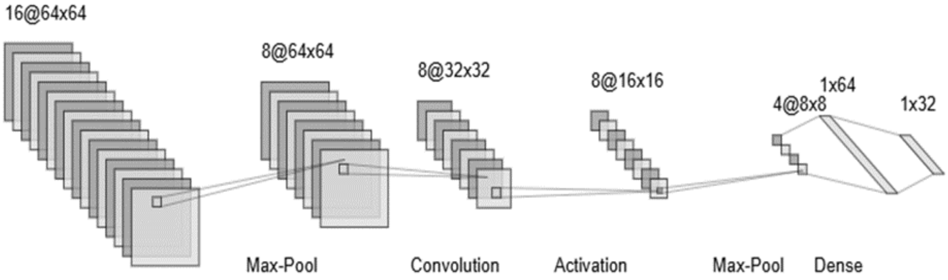


Figure 2. *Established CNN architecture.*

In Table 2, the layers and output dimensions of the CNN architecture are given. The model in Table 2 is proposed for image classification or object detection.

Table 2. *Parameters of CNN* architecture.

Layer Name	Output Size	Layer
Input	64 x 64	16
Max – Pooling_1	64x64	8
Convolution	32x32	8
Activation	32x32	8
Max-Pooling_2	16x16	8
Dense	8x8 1x64 1x32	4

The model starts with 16 channels and processes the input image of size 64x64. The first Max-Pooling layer reduces only the number of channels without affecting the spatial dimensions, concentrating the features. Convolution Layer extracts complex features, reducing spatial dimensions while preserving the channel count. Non-linear activation ensures that the model can learn complex relationships without changing the output dimensions. The second pooling layer further reduces the spatial dimensions for more generalized and lower-dimensional representation. At the end, the feature maps are flattened and passed to fully connected layers. The output progressively reduces to 1x32, likely for classification or regression tasks.

3. Performance Metrics

Various metrics are used to evaluate the performance of machine learning and deep learning algorithms in classification problems. These metrics help compare the accuracy and effectiveness of different models, enabling the selection of the best-performing one. Commonly used evaluation metrics are calculated based on a table known as the "confusion matrix." The confusion matrix is a visual tool that summarizes the performance of a classification model. In this table, the columns represent the predicted classes, while the rows indicate the actual classes. An example of a confusion matrix for a binary classification problem is shown in Table 3.

Table 3. *Two-class confusion matrix*

		PREDICTED CLASS	
		POSITIVE	NEGATIVE
ACTUAL CLASS	POSITIVE	True Positive (TP)	False Negative (FN)
	NEGATIVE	False Positive (FP)	True Negative (TN)

True Positive (TP) refers to the correct prediction of positive examples as positive, meaning the model accurately identifies the positive class. False Negative (FN) refers to the incorrect prediction of positive examples as negative, where the model mistakenly classifies a positive instance as negative. False Positive (FP) refers to the incorrect prediction of negative examples as positive, meaning the model incorrectly classifies a negative instance as positive. True Negative (TN) refers to the correct prediction of negative examples as negative, where the model accurately identifies the negative class. These four terms are crucial for evaluating the performance of classification models and are used in calculating metrics such as accuracy, precision, recall and F1-Score. All the evaluation indices are defined in Table 4.

Table 4. *Performance Metrics.*

Metric	Formula
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	$\frac{TP}{TP + FP}$
Recall	$\frac{TP}{TP + FN}$
F1-Score	$\frac{2xPrecisionxRecall}{Precision\ x\ Recall}$

Accuracy represents the proportion of correctly classified instances out of all instances, but it may not be sufficient in cases of class imbalance. Precision measures the proportion of positive predictions that are actually correct, making it crucial when minimizing false positives is important. Recall evaluates how many actual positive instances are correctly identified, being essential when missing positive cases has severe consequences. The F1 Score, as the harmonic mean of precision and recall, provides a balanced metric and is particularly useful in imbalanced classification problems.

4. Results and Analysis

The proposed CNN was implemented and trained on a system equipped with an Intel Xeon E5-1603 v3 2.80GHz (8-core) CPU and 16 GB of RAM. The experimental environment is presented in Table 5.

Table 5. *Experimental environment.*

Hardware Environment		Software Environment	
Memory	16GB	System	Windows 10 Pro
CPU	Intel Xeon E5- 1603 v3 2.80GHz (8 core)	Environment configuration	Python 3.7.3, Keras 2.13.0

The training configuration included a batch size of 64, indicating the number of images processed in each training step. The model using the Adam optimization algorithm, and the cross-entropy loss function was trained for 200 iterations, with a batch size of 64 and learning rate of 1 × 10^{-2}. The parameter configuration is shown in Table 6. This setup was chosen to balance computational efficiency and model performance. The specified parameters were optimized to achieve effective convergence while minimizing overfitting. These parameters played a crucial role in ensuring the stability and accuracy of the training process.

Table 6. *Training parameters.*

Training Parameters	Values	Definations
Batch Size	64	Number of pictures per training
Learning Rate	1×10^{-2}	Initial learning rate
Epoch	200	Training iteration times

In the Figure3, the obtained loss graph clearly illustrates how the training and validation losses vary with the number of epochs. At the beginning of the training process, the loss value starts at approximately 80% and rapidly decreases as the number of epochs increases, eventually stabilizing below 5%. This indicates that the model progressively learns the patterns in the dataset and reduces its errors over time. Notably, during the first 25 epochs, both training and validation losses show a sharp decline. The training loss drops from around 80% to approximately 10% in a short time, indicating that the model is in the initial phase of learning the fundamental patterns in the dataset and significantly reducing its errors. Similarly, the validation loss also demonstrates a downward trend, suggesting that the model performs well not only on the training data but also on the validation data.

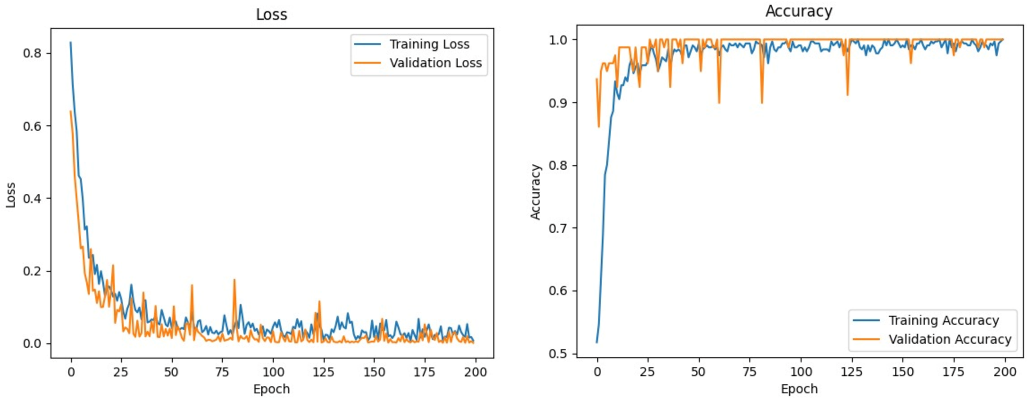


Figure 3. *The model was trained with a training dataset and test datasets: (a) Loss value; (b) Accuracy value.*

As the number of epochs increases, slight fluctuations can be observed in both training and test losses. Although the test loss occasionally exhibits brief spikes, the overall trend remains downward, eventually falling below 5%. These fluctuations indicate that the model encounters challenges with certain examples in the test set. Such increases in test loss could arise due to the diversity of the dataset or limitations in the model's generalization ability. The fact that the training and test loss values remain close to each other demonstrates that the model does not exhibit overfitting. The simultaneous decrease in both training and validation losses during the training process indicates that the model has effectively learned the patterns in the training data and successfully reflected this knowledge in the test data. Given the absence of signs of overfitting, it can be concluded that the model has strong generalization capabilities.

Table 7. Presents the precision, recall, F1-score, and accuracy of CNN method for classifying wood knot defect images.

Table 7. *The evaluation index values of the network.*

Metrics	Training	Test
Accuracy	0.9810	0.9760
F1 score	0.9812	0.9760
Precision	0.9812	0.9690
Recall	0.9812	0.9710

Upon analyzing the Figure 4, it is evident that instances belonging to the true class "Knotty" are classified as "Knotty" with 100% accuracy, demonstrating the model's exceptional performance in recognizing the "Knotty" class. However, 1.10% of the instances labeled as "Knotty" are misclassified as "Knotless," indicating a minor but noteworthy error in distinguishing certain "Knotty" instances. This observation underscores the existence of a minimal error margin. Furthermore, the graph reveals that no instances of the true class "Knotless" (0%) are incorrectly classified as "Knotty," signifying the model's robustness in avoiding false positive classifications for the "Knotty" class. Lastly, the model achieves a classification accuracy of 98.90% for the "Knotless" instances, highlighting its strong capability to correctly identify examples of the "Knotless" class.

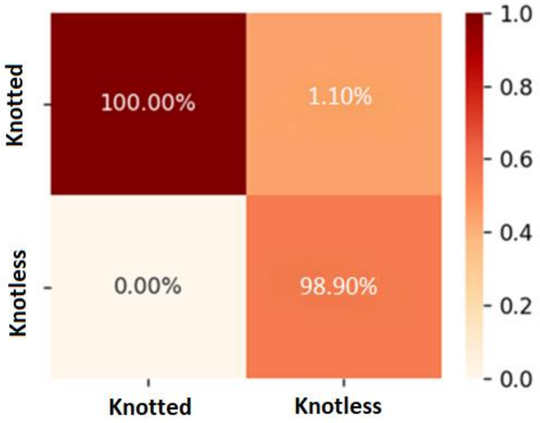


Figure 4. *Wood knots classification results.*

5. Conclusions

In summary, a neural network model, CNN, was proposed to quickly and accurately identify wood knot defects. By extracting structural defect features, optimizing training parameters, and improving datasets and images, the network achieved an accuracy of 99.90%. Experimental results showed that CNN reached a high recognition rate of 99.90% on the training dataset and a low training loss of 1.30% on the validation dataset during the process of identifying 3000 different wood knot defects. The overall accuracy reached 98.60%, and the loss curve and accuracy curve exhibited small fluctuation ranges when CNN was applied to the test dataset.

Moreover, this method does not require extensive image preprocessing or feature extraction when detecting various wood defects and demonstrates high efficiency and recognition accuracy during both the training and testing stages. This indicates that the collected wood knot defects can be accurately and quickly identified using the proposed CNN method. Based on the above analysis, the proposed CNN parameters have potential applications in wood non-destructive testing and wood defect detection.

References

[1] Görgün H V, “Budak tipleri ve değerlendirme farklılıkları,” *Artvin Çoruh Üniversitesi Orman Fakültesi Dergisi*, 24(1) (2023) 96-105; doi:10.17474/artvinofd.1177307.

[2] As N, Dündar T, Büyüksarı Ü, “Budakların Odunun Fiziksel ve Mekanik Özellikleri Üzerine Etkileri”, *Journal of the Faculty of Forestry Istanbul University, 58(2) (2008) 1-13; https://doi.org/10.17099/jffiu.76055*.

[3] Doğu D, Koç H, As N, Atik C, Aksu B, Erdinler S, “Türkiye’de Yetişen Endüstriyel Öneme Sahip Ağaçların Temel Kimlik Bilgileri ve Kullanıma Yönelik Genel Değerlendirme”, *Journal of the Faculty of Forestry Istanbul University,* 51(2) (2014) 69-84; https://doi.org/10.17099/jffiu.33874.

[4] Özkan S, “Kayın (Fagus Orientalis L.) Kerestesinde Eğilme Özelliklerinin Tahribatsiz Yöntemle Tespiti”, Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü , Isparta, 2012.

[5] Yılmaz, M, Şahin, H, Yıldız, A, “Sectoral Application Analysis of Studies Made with Deep Learning Models”, *Electronic Letters on Science & Engineering*,17(2) (2021) 126-140.

[6] Özgür S. B., “Algoritmalar, Yapay Zeka, Makine Öğrenmesi, Derin Öğrenme ve Uygulamaları: Beşeri Fayda Üretiminin Yazılımlar Tarafından Karşılanması”, *Ekonomi ve Yönetim Araştırmaları Dergisi*, 10(1) (2021) 1-29.

[7] Eker R, Alkiş KC, Uçar Z, Aydın A, “Ormancılıkta makine öğrenmesi kullanımı”, *Turkish Journal of Forestry | Türkiye Ormancılık Dergisi*, 24(2) (2023), 150-177; doi: 10.18182/tjf.1282768.

[8] Çetiner H, Çetiner İ, “Classification of Cataract Disease with a DenseNet201 Based Deep Learning Model”, *Journal of the Institute of Science and Technology,* 12(3) (2022) 1264-1276; doi:10.21597/jist.1098718.

[9] Gurkan, C, Kozalioglu, S, Palandoken, M, “Real Time Mask Detection, Social Distance and Crowd Analysis using Convolutional Neural Networks and YOLO Architecture Designs”, *Academic Perspective Procedia,* 4(1) (2021) 195-204, doi: 10.33793/acperpro.04.01.29.

[10] Elkıran, H, “OCC-OPENCV Kütüphanesi için Blok Tabanlı Programlama Aracı,”, *Yüksek Lisans Tezi, İstanbul Sabahattin Zaim Üniversitesi Fen Bilimleri Enstitüsü*, 2020.

[11] Shukla N, Fricklas K, “Machıne Learnig with Tensorflow”, (2nd Ed.), Manning, USA,2018.

[12] Primandani Arsi and Retno Waluyo, “Analisis Sentimen Wacana Pemindahan Ibu Kota Indonesia Menggunakan Algoritma Support Vector Machine (SVM)”, *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, 8(1) (2021) 147-156; doi: 10.25126/jtiik.202183944.

[13] Baita, A, Yoga P, Cahyono N “Analisis Sentimen Mengenai Vaksin Sinovac Menggunakan Algoritma Support Vector Machine (SVM) dan K-Nearest Neighbor (KNN)”, *Information System Journel (INFOS)*, 4(2) (2021) 42-46; https://doi.org/10.24076/infosjournal.2021v4i2.687.

[14] Çavuşlu MA, Becerikli Y, Karakuzu C, “Levenberg-Marquardt Algoritması ile YSA Eğitiminin Donanımsal Gerçeklenmesi Hardware Implementation of Neural Network Training with Levenberg-Marquardt Algorithm,” *Türkiye Bilişim Vakfi Bilgisayar Bilimleri Ve Mühendisliği Dergisi,* 5(1) (2016) 1.

[15] Kayalı N Z, Omurca İS, “Konvolüsyonel Sinir Ağları (CNN) ile Çin Sayı Örüntülerinin Sınıflandırması,” *Journal of Computer Science*,Sep. IDAP 2021(1) (2021) 184 – 191; doi: 10.53070/bbd.989668.

[16] Aalami N, “Hierarchical Convolutional Neural Networks for Fashion Image Classification”, *Expert Systems with Applications,* 116(1) (2019) 328-339; doi: 10.1016/j.eswa.2018.09.022.

[17] Samtaş G, Gülesin M, “Sayısal Görüntü İşleme ve Farklı Alanlardaki Uygulamaları”, *Electronic Journal of Vocatinal Collages*, 2(1) (2011) 85 - 97.

[18] Ide H, Kurita T, “Improvement of Learning for CNN with ReLU Activation by Sparse Regularization,” in *Proceedings of the International Joint Conference on Neural Networks,* Anchorage, AK, USA, 2017, 2684-2691; doi: 10.1109/IJCNN.2017.7966185.

[19] Cengil E, Çınar A, “A New Approach for Image Classification: Convolutional Neural Network,” *European Journal of Technic EJT*, 6(2) (2016) 96 - 103.

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

Vol. 4 (No. 2), pp. 117-123, 2024
doi: 10.54569/aair.1608702
Published online: Dec 28, 2024

ALTO-assisted Peer Selection in Bitcoin P2P Network

Cihat ÇETİNKAYA [1,*,]

[1] Department of Software Engineering, Faculty of Engineering, Muğla Sıtkı Koçman University, Turkiye;

Abstract

Blockchain-based applications rely on a decentralized structure wherein the transactions are recorded on a public ledger that is maintained by every node in the peer-to-peer (P2P) network. The transactions and blocks are propagated using a multi-hop broadcast and verified by every node in the network. Application Layer Traffic Optimization (ALTO), on the other hand, is a network protocol developed and maintained by the Internet Engineering Task Force (IETF) to provide network related information to the P2P applications to increase their performance. In this study, a novel peer selection method based on the network information provided by ALTO protocol is proposed to decrease the block propagation delay of the Bitcoin P2P network. The simulations show that the proposed peer selection method can effectively decrease the block propagation time and fork rate compared to Bitcoin's random peer selection and region-based peer selection methods.

***Keywords**: P2P network, Blockchain, Traffic Optimization, Peer Selection, Bitcoin.*

1. Introduction

Blockchain technology emerged in 2008 with the development of the Bitcoin [1] cryptocurrency by a group of researchers using the nickname Satoshi and has since attracted attention from both academia and industry. Due to its distributed architecture, blockchains are used in smart contracts, internet of things (IoT), non-fungible tokens (NFTs), healthcare, logistics, and personnel digital security, as well as cryptocurrency.

In P2P applications, which also form the basis of blockchain-based systems, one of the factors affecting performance is the peer selection process in which the nodes in the P2P network select the peers with whom they will exchange data [2]. In addition to the fact that peer selection is usually done randomly, in some P2P applications, criteria such as the geographical distance between nodes, the upload/download bandwidth of the nodes, and the chunks of data held by the nodes also guide the peer selection process. However, both peer selection processes do not consider *(i)* the topology of the network on which the applications run and *(ii)* the localization of network traffic. While peer selection without network topology information usually reduces the performance of the P2P application, peer selection without considering network traffic information causes a cost for the economies of Internet Service Providers (ISPs). Therefore, it is crucial to consider both network topology and network traffic information when peer selection is performed in P2P applications.

In the literature, studies that provide such network-related information to P2P applications are classified into two different groups [3]. In the first group, network-related information is estimated and provided to the nodes by running a distributed application at the application layer [4, 5]. In the second group of studies, network-related information is provided by ISPs that own the network [6-8]. When the performances of the studies in both groups are analyzed on P2P applications, it is seen that the ISS-based approaches in the second group provide a more effective peer selection for P2P applications [3]. Therefore, the IETF (Internet Engineering Task Force) group introduced the ALTO protocol [9] to provide network-related information to the peers that run on P2P applications. One of the objectives of the ALTO protocol is to design and define the ALTO service that provides the necessary network-related information to the nodes running on P2P applications to perform better-than-random peer selection.

In this study, an ALTO-assisted peer selection method is proposed for blockchain-based systems. In the proposed method, peer selection is based on a multi-objective optimization model and aims to select peers that will reduce the block propagation delay by using network information obtained from the ALTO server. In this paper, the peer selection method is implemented on the Bitcoin P2P network since it is both a public blockchain, consists of hundreds of thousands of nodes deployed in many autonomous systems around the world, and has the highest commercial value. However, the peer selection method proposed in this study can also be applied to other alternative public blockchains such as Litecoin and Dogecoin with little or almost no modification.

The rest of this paper is as follows. Section 2. provides background information about Bitcoin, ALTO protocol and summarizes related works. Section 3. presents the proposed peer selection method, while Section 4. describes the simulation study and presents the results and comparative analysis. The conclusion and future work discussed in Section 5.

*Corresponding author
cihat.cetinkaya@mu.edu.tr

Received: 27/Dec/2024; Accepted: 28/Dec/2024.

2. Background and Related Works

According to the Bitcoin protocol, a new node joining the network first performs peer discovery mechanism since it does not yet have any information about the other nodes in the network. In the first phase of peer discovery, the node obtains information about nodes in the network by sending queries to DNS seeds that are hardcoded in the Bitcoin reference software. Then, the node tries to establish a connection by sending a *version* message to the nodes in the node list randomly obtained from DNS seeds (**Figure 1**). If the remote node sends *verack* message, the node adds the remote node as outgoing peer and the remote node adds the new node as incoming peer. The nodes also exchange information about other nodes they discover on the network by sending *addr* and *getaddr* messages to each other (**Figure 2**). By default, Bitcoin implements a total of 125 peer connections, 117 of which are incoming connections and 8 are outgoing connections. In our previous work [10], we investigated the optimum number of outgoing connections through simulations and found out that the optimum number for outgoing connections is 8-10 which is almost same with the default value of Bitcoin.

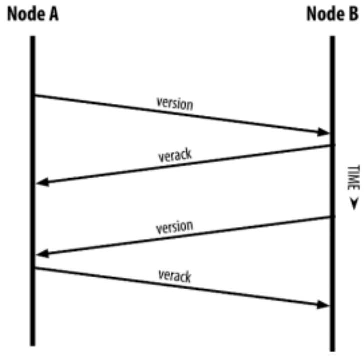


Figure 1. *Message timeline of connection establishment between nodes.*

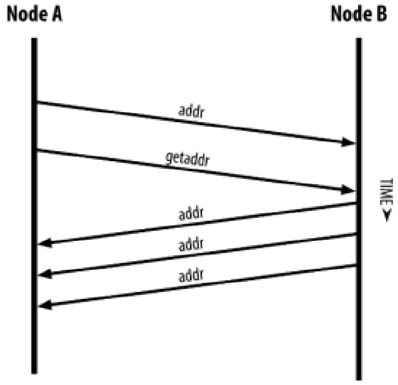
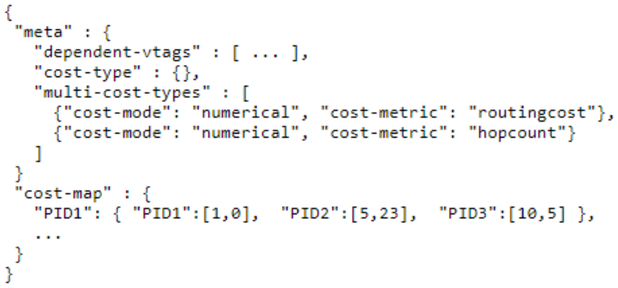


Figure 2. Message timeline of exchanging discovered nodes

The main purpose of the ALTO protocol, which was standardized by the IETF in 2014 as RFC 7285, is to define ALTO services that provide the network-related information to the applications running on the network so that the nodes in the applications can perform better than random peer selection. Upload and download bandwidth capacity of nodes, packet routing cost due to ISPs policy, topological hop count, end-to-end delay, traffic quota can be given as network-related information that is served by ALTO protocol. According to ALTO protocol, the ALTO server is responsible for delivering the ALTO service where ALTO client queries ALTO server with different ALTO queries. The multi-cost map service, which is one of the services defined in the ALTO protocol, enables multiple cost metric to be served by making a single query/response transaction. **Figure 3**. presents an example of multi-cost map that is received by an ALTO client. There are 2 different cost metrics in the map, *routingcost* and *hopcount*, both of which are numerical. According to the cost information given in the map, the *routingcost* between PID1 and PID2 is 5 while the *hopcount* is 23. In the proposed peer selection method, the node that joins the P2P network acts as an ALTO client and queries the ALTO server and receives multi-cost map for the candidate nodes in the network. The details are given in Section 3.



```
{
  "meta" : {
    "dependent-vtags" : [ ... ],
    "cost-type" : {},
    "multi-cost-types" : [
      {"cost-mode": "numerical", "cost-metric": "routingcost"},
      {"cost-mode": "numerical", "cost-metric": "hopcount"}
    ]
  }
  "cost-map" : {
    "PID1": { "PID1":[1,0],  "PID2":[5,23],  "PID3":[10,5] },
    ...
  }
}
```

Figure 3. *Example of a multi-cost map received by ALTO client after querying ALTO server* [11]

Several peer selection methods have been proposed for blockchain-based systems. In [12], the nodes in the network are clustered with respect to physical distance between nodes and the peer selection is performed within cluster. In [13-15] the closest nodes based on the geographical distance are considered in the peer selection process. In [16], the peers are selected based on the ping latencies between nodes. In [17, 18], similarly aimed to select peers with low delay according to the protocol messages that nodes received from peers. In [19], the authors propose a region-based peer which is based on regional information of nodes. Previous studies performed peer selection without having any information about the real network on which they run. The novelty of the proposed study is that it uses up-to-date fine-grained network information obtained from ALTO services.

.

3. Proposed Peer Selection Method

In the proposed peer selection method, when a new peer joins the Bitcoin P2P network and after getting the initial node list from the DNS seeds starts the peer discovery process. Differing from the default peer discovery process given in Section-II, in the proposed peer selection method the node does not send *verack* messages to the other peers in the network. Instead, the peer keeps discovering the nodes in the network by sending *getaddr* messages to the other nodes in the network. After completion of this step, the node has obtained information (e.g. IP address and protocol version) of several nodes in the network and keeps their information in a node list. In the second step, the node queries the ALTO server using ALTO multi-cost service with the IP addresses of the nodes in the node list and receives the end-to-end delay, upload and download bandwidth capacities as the cost variables of the nodes. The node applies a certain threshold value to each of the delay, upload and download bandwidth values of the nodes in the node list and creates a candidate peer list from nodes that do not exceed the threshold value for all three values.

In the third step, the node applies a multi-objective optimization model aiming to find a peer that minimizes the delay while maximizing the upload and bandwidth capacities. Let N represents the nodes in the candidate peer list. For each $n \in N$; $dl_{n,}$ $up_n$and dw_n denotes the delay, upload and bandwidth values of the node n, respectively. A row vector for each node is constructed and given in Eq.(1) as follows:

$$\overrightarrow{N_{(n)}} = [dl_n, 1/up_n\,, 1/dw_n] \tag{1}$$

In Eq.(1), reciprocal values of upload and download bandwidth are used. Thus, the optimization model given in Eq.(3) aims to minimize all cost variables. Since each cost variable may have different effects on the performance of the peer, a weight vector of $\vec{w}$ is assigned to objective variables and weighted objective variables are calculated in Eq.(2) by inner product of $\vec{w}$ and $\overrightarrow{N_{(n)}}$.

$$\Psi_n = \langle \vec{w}, \overrightarrow{N_{(n)}} \rangle \tag{2}$$

Since the aim of the optimization model is to find a node that minimizes all objective variables, an utopia point υ is defined that represents the optimal solution. The optimization model is given as follows:

$$\begin{aligned} &\underset{\text{n}}{minimize} \quad \|\Psi_n, \upsilon\| \\ &subject\ to \quad n \in N \end{aligned} \tag{3}$$

The optimization model given in Eq.(3) is solved using an exhaustive search and returns a node n_i whose objective variables are closest to utopia point υ among other nodes in $\overrightarrow{N_{(n)}}$. Then, n_i is removed from the candidate peer list and the node tries to establish an outgoing connection with n_i by sending a version message as discussed in Section 2. The third step of the peer selection process is repeated until the node successfully connects to at most 6 outgoing peers. The rest of the peers are randomly selected from the node list populated in the first step of the peer selection process. By selecting a subset of peers randomly makes the proposed peer selection method resilient to eclipse attacks [20].

4. Simulation Study

First, the simulation environment based on Simblock Bitcoin P2P simulator [21] was set up. Then the proposed work was tested with different weights given in Section 3. Last, the performance of the proposed work was compared with Bitcoin’s default peer selection method and region-based peer selection like method presented in [19] from the literature.

4.1. Simulation Setup

Simblock is a discrete-event simulator that can simulate Bitcoin P2P network with the exact same parameters that Bitcoin P2P network had in 2015 and 2019. Since its release, Simblock has been extensively used [22] by Blockchain researchers and developers. Since the default parameters of Simblock are out of date and do not reflect the current characteristics of the Bitcoin P2P network, the up-to-date parameters were gathered for Bitcoin P2P network and these parameters were passed to Simblock.

Geographical distribution of the nodes used in the simulations are given in **Table 1**. The values presented in **Table 1** were obtained by averaging the unique nodes discovered in Bitcoin P2P network using Bitnodes API [23] from July 1, 2024, to July 31, 2024. In the simulations, the number of nodes varied as 500, 1000, 2000 and 4000, and the nodes were randomly located in the network regions according to the rates given in **Table 1**. The download/upload bandwidths of the nodes were calculated using country-based values obtained from the testmy.net [24] website. The latencies between the network regions were retrieved from Verizon [25].

The other Bitcoin-related parameters used in the simulations are presented in **Table 2**. The end block height parameter, which indicates the number of blocks to be mined during the simulations, is calculated as the total number of blocks mined in the Bitcoin network between July 1, 2024, and July 31, 2024, and the average block size parameter is calculated as the arithmetic average of the total sizes of the blocks mined in the same period.

Table 1. *Geographical distribution of nodes*

Region	Rate
North America	18.9 %
Europe	59.7 %
South America	4.3 %
Asia Pacific	13.7 %
Japan	1.6 %
Australia	1.8 %

Table 2. *Simblock parameters used in simulations*

Parameter	**Value**
# of nodes	[500, 1000, 2000, 4000]
Average block size	1.69 MB
Compact block size	13 KB
End block height	4713
Block generation interval	10 mins

4.2. Performance Evaluation

After setting up the simulation environment, the proposed peer selection method was tested using different weights $\vec{w}$ given in the Section 3. Each test was conducted 30 times and average block propagation delay and fork rate values were reported. It was found that the best result obtained for the proposed selection method was when the weight vector $\vec{w}$ was assigned as $\langle 0.5,0.25,0.25\rangle$ where the elements of the vector denote delay, upload bandwidth and download bandwidth, respectively.

To evaluate the performance of the proposed peer selection method, the results were compared with Bitcoin's default peer selection method and region-based like method. In Bitcoin's default peer selection method, the peers were selected randomly. In region-based like method, 6 out of 8 peers were selected within the same region as node's where remaining 2 peers were selected randomly from outside of the node's region.

Both peer selection methods were tested 30 times with the same simulation parameters used to test the proposed method and the average results were reported.

Figure 4, **Figure 5** and **Figure 6** present the average block propagation delay, block propagation delay to reach 50% of nodes and, block propagation delay to reach 90% of nodes, respectively. It can be seen that the proposed peer selection method outperforms the default peer selection of Bitcoin and region-based like peer selection by reducing the block propagation time in general. This indicates that using fine-grained network-related data in peer selection process plays an important role in Bitcoin P2P network.

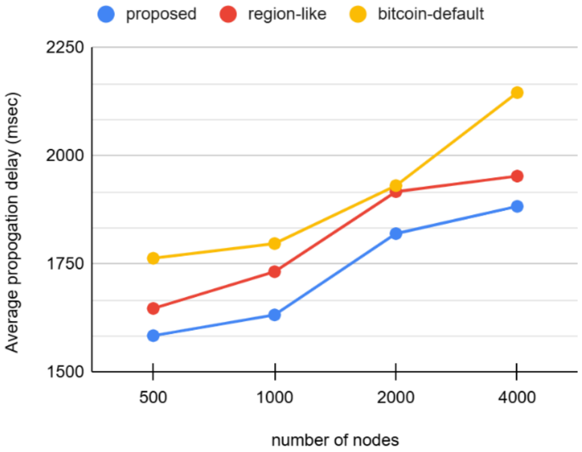


Figure 4. *Average block propagation delay*

Table 3 presents the block propagation delay where the block reaches every node in the network. It is observed from **Table 3**. that the time required for the block to reach all nodes in the network is the lowest in the proposed method. However, it is seen that the values obtained are close to each other in all peer selection methods. This is because there are nodes in the network with low bandwidth capacity and high end-to-end delay.

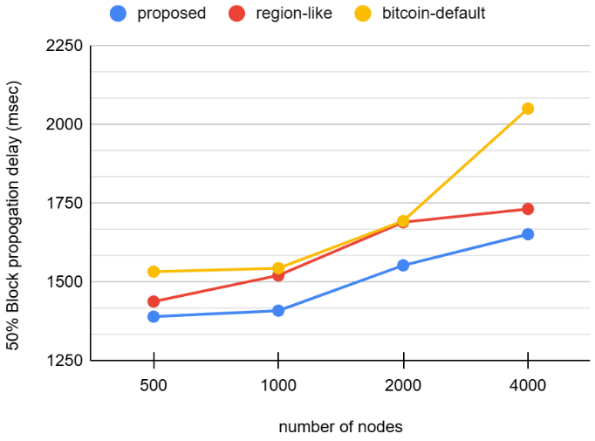


Figure 5. *Block propagation delay to reach 50% of nodes*

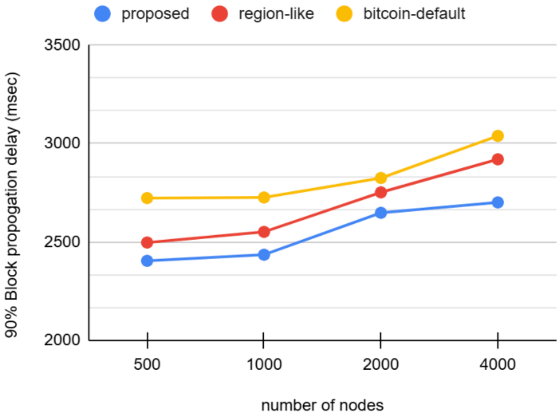


Figure 6. *Block propagation delay to reach 90% of nodes*

Table 3. *Block propagation delays reaching all nodes in the network. (msecs)*

	Number of nodes			
Method	500	1000	2000	4000
proposed	7873	8232	9951	10808
region-like	8118	8559	10182	10980
bitcoin-default	8284	8749	10331	11222

The average fork number of the peer selection methods are given in **Table 4**. As in block propagation delay, the proposed method has a lower fork number than the other peer selection methods for all the tests that were conducted with different number of nodes. In addition, as the number of nodes in the network increases, the chances of nodes finding good peers also increases, so the number of forks decreases in all methods. All the results obtained with the simulations show that the peer selection made by obtaining detailed information about the network thanks to the ALTO protocol plays an important role in reducing the block propagation delay and thus increasing the security of the blockchain by minimizing the possibility of forks.

Table 4. *Average number of forks in the blockchain.*

	Number of nodes			
Method	500	1000	2000	4000
proposed	13.9	13.6	13.5	12.8
region-like	14.9	14.1	14.0	13.3
bitcoin-default	16.6	14.9	14.1	13.8

5. Conclusion

In this paper, a novel peer selection method for Bitcoin P2P network was proposed under the guidance of ALTO protocol. The proposed method aims to select the optimum peers that minimize the block propagation delay by considering the upload/download bandwidth capacity of the candidate nodes and the network delay between the node and the candidate nodes, which are the network-related information obtained by the node from the ALTO server. The simulation results show that ALTO-assisted peer selection outperforms default random peer selection of Bitcoin and region-based peer selection. It is also shown that both the upload and download bandwidth capacities of nodes affect the block propagation delay. As a future work, we plan to propose a reinforcement learning-based peer selection method.

Declaration of interest

The authors declare that there is no conflict of interest.

Acknowledgements

This work is funded by the Scientific and Technological Research Council of Turkey (TUBITAK) Electric, Electronic and Informatics Research Group (EEEAG) under grant 121E401.

References

[1] Nakamoto, S. “Bitcoin: A Peer-to-Peer Electronic Cash System”, 2008, [Online] Available: https://bitcoin.org/bitcoin.pdf (accessed: December 01, 2024).

[2] Shen X, Yu H, Buford J, Akon M. “Handbook of Peer-to-Peer Networking”, New York, Springer, 2010.

[3] Gurbani VK, Hilt V, Rimac I, Tomsu M, Marocco E. "A survey of research on the application-layer traffic optimization problem and the need for layer cooperation”, IEEE Communications Magazine, 47, 107-112, 2009.

[4] Costa M, Castro M, Rowstron A, Key P. “PIC: Practical Internet coordinates for distance estimation”, in Proceedings of International Conference on Distributed Systems, 2003.

[5] Dabek F, Cox R, Kaashoek F, Morris R. “ Vivaldi: A Decentralized Network Coordinate System”, in Proceedings of ACM SIGCOMM, 2003, 15-26.

[6] Saucez D, Donnet B, Bonaventure O. “Implementation and Preliminary Evaluation of an ISP-Driven Informed Path Selection”, in Proceedings of. ACM CoNEXT, 2007,1-2.

[7] Aggarwal V, Feldmann A, Scheideler C. “Can ISPs and P2P systems co-operate for improved performance?”, ACM SIGCOMM Computer Communications Review (CCR), 37(3), 29-40, 2007.

[8] Xie H, Yang YR, Krishnamurthy A, Liu Y, Silberschatz A. “P4P: Provider Portal for (P2P) Applications”, in Proceedings of ACM SIGCOMM, 2008, 351-362.

[9] Alimi R, Penno R, Yang Y, Kiesel S, Previdi S, Roome W, Shalunov S, Woundy R. “Application-Layer Traffic Optimization (ALTO) Protocol”, 2014, [Online], Available: https://datatracker.ietf.org/doc/rfc7285/ (accessed: December 01, 2024).

[10] Cetinkaya C. “A Study on the Impact of Connection Number Parameter of Nodes on the Performance of Bitcoin Peer-to-Peer Network”, 5th International Conference on Data Science and Applications, 2022, 131-134.

[11] Randriamasy S, Wendy R, Schwan N. “Multi-Cost Application-Layer Traffic Optimization (ALTO)”, 2017, [Online], Available: https://datatracker.ietf.org/doc/rfc8189/ (accessed: December 01, 2024).

[12] Fadhil M, Owenson G, Adda M.“A Bitcoin Model for Evaluation of Clustering to Improve Propagation Delay in Bitcoin Network”, in Proceedings of IEEE Intl Conference on Computational Science and Engineering, 2016.

[13] Fadhil M, Owenson G, Adda M. “Locality based approach to improve propagation delay on the Bitcoin peer-to-peer network”, in Proceedings of the IFIP/IEEE International Symposium on Integrated Network and Service Management, 2017, 556-559.

[14] Park S, Im S, Seol Y, Paek J. “Nodes in the Bitcoin Network: Comparative Measurement Study and Survey”, IEEE Access, 7, 57009-57022, 2019.

[15] Sudhan A, Nene M. “Peer Selection Techniques for Enhanced Transaction Propagation in Bitcoin Peer-to-Peer Network”, in Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, 2019, 679-684.

[16] Sallal M, Owenson G, Adda M. “Proximity Awareness Approach to Enhance Propagation Delay on the Bitcoin Peer-to-Peer Network”, in Proceedings of the International Conference on Distributed Computing Systems, 2017, 2411-2416.

[17] Wang K, Kim H. “FastChain: Scaling blockchain system with informed neighbor selection”, in Proceedings of the 2nd EEE International Conference on Blockchain, 2019, 376-383.

[18] Aoki Y, Shudo K. “Proximity neighbor selection in blockchain networks”, in Proceedings of the 2nd IEEE International Conference on Blockchain, 2019, 52-58.

[19] Matsuura H, Goto Y, Sao H. “Region-based Neighbor Selection in Blockchain Networks”, in Proceeding of the IEEE International Conference on Blockchain, 2021, 21-28.

[20] Heilman E, Kendler A, Zohar A, Goldberg S. “Eclipse attacks on Bitcoin’s peer-to-peer network”, USENIX Security Symposium, 2015, 129–144.

[21] Aoki Y, Otsuki K, Kaneko T, Banno R, Shudo K. “Simblock: A Blockchain Network Simulator”, in Proceedings of IEEE Conference on Computer Communications Workshops, 2019, 325-329.

[22] Shudo K, Hasegawa T, Sakurai A, Banno R. "Blockchain Network Studies Enabled by SimBlock," 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates, 2023, pp. 1-2.

[23] Global Bitcoin nodes distribution, [Online] Available: https://bitnodes.io/api/ (accessed: December 01, 2024).

[24] Internet Speed Test, [Online], Available: https://testmy.net (accessed: December 01, 2024).

[25] Verizon Network Performance, [Online] Available: https://verizon.com (accessed: December 01, 2024).