Computer Science and Interdisciplinary Research Journal

Volume 1, Issue 2, 2024

ISSN: 3033-1218



CSIR Journal

https://csirjournal.com

Computer Science and Interdisciplinary Research Journal [CSIR Journal]

ISSN: 3033-1218
Volume 1, Issue 2, 2024.

Published by Science, Education, Technologies Association
Plovdiv, Bulgaria

Licensed under CC BY-NC-ND 4.0.

The views expressed in the articles are those of the authors and do not necessarily reflect the views of the publisher or editorial team.

Editorial Team

Editors-in-Chief

Section COMPUTER SCIENCE

Prof. Asya Stoyanova-Doycheva, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Section MATHEMATICS

Assoc. Prof. Mariyan Milev, Ph.D.
Sofia University St. Kliment Ohridski, Sofia, Bulgaria

Section EDUCATION

Prof. Todorka Glushkova, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Section INTERDISCIPLINARY RESEARCH

Assoc. Prof. Stanka Hadzhikoleva, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Editorial Board

Prof. Aldo Tagliani, Ph.D.
Università di Trento, Trento, Italy

Prof. Igor Britchenko, Ph.D.
University of the National Education Commission, Krakow, Poland

Prof. Emil Hadzhikolev, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Prof. Miglena Koleva, Ph.D.
University of Rousse, Ruse, Bulgaria

Assoc. Prof. Senthil Kumar, Ph.D.
Mepco Schlenk Engineering College, Sivakasi, India

Assoc. Prof. Margarita Terziyska, Ph.D.
University of Food Technology, Plovdiv, Bulgaria

Assoc. Prof. Stoyan Cheresharov, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Assoc. Prof. Velika Kuneva, Ph.D.
Agricultural University, Plovdiv, Bulgaria

Assoc. Prof. Silvia Gaftandzhieva, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Assoc. Prof. Hristina Kulina, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Assoc. Prof. Mikhail Kolev, Ph.D.
University of Architecture, Civil Engineering and Geodesy, Sofia, Bulgaria

Assoc. Prof. Hristo Hristov, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Shih-Hao Chang, Ph.D.
National Taipei University of Technology, Taiwan

Gordana Rakić, Ph.D.
University of Novi Sad, Serbia

Kostadin Yotov, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria



English Proofreader

Assoc. Prof. Vanya Ivanova, Ph.D.
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria



Technical Assistance

Ph.D. student Maria Gorgorova
University of Plovdiv Paisii Hilendarski, Plovdiv, Bulgaria

Table of contents

Research and Application of Deep Learning Models for Detecting Brain Abnormalities Based on CT Images 2

Dinh Cong Tung, Mai Van Linh, Dimitar Borisov

Numerical Analysis of the Magnetic Moment in S/F/S-Josephson Junction 10

Elis Dieva

Classification of News Using Artificial Intelligence Methods 17

Georgi Gustinov, Emil Hadzhikolev, Stanka Hadzhikoleva, Stoyan Cheresharov

Opportunities to Achieve Cyber Security Through Blockchain and Smart Contracts 25

Margarita Terziyska, Zhelyazko Terziyski, Valentina Nikolova-Alexieva, Katina Valeva

Computer Science and Interdisciplinary Research Journal
Volume 1, Issue 2, ISSN: 3033-1218, DOI: 10.70862/CSIR.2024.0101-06

Research and Application of Deep Learning Models for Detecting Brain Abnormalities Based on CT Images

Dinh Cong Tung[1], Mai Van Linh[2], Dimitar Borisov[3]

[1] Faculty of Information Technology, University of Transport and Communications, Hanoi, Vietnam
[2] Faculty of Information Technology, East Asia University of Technology, Hanoi, Vietnam
[3] University of Chemical Technology and Metallurgy, Sofia, Bulgaria

[1] tungdc@utc.edu.vn, [2] linhmv@eaut.edu.vn, [3] shtain@uctm.edu

Abstract – This paper studies a method to improve the accuracy of a deep learning model for detecting brain abnormalities based on computed tomography images. The process begins with image preprocessing using the Histogram Equation algorithm and Gabor filter. Then, features are extracted from the fully connected layer of the AlexNet model. To optimize feature extraction, we use the MIFS algorithm to identify the most important features. Finally, the SVM machine learning model is deployed to detect brain abnormalities. To confirm the effectiveness of the proposed model, we compare it with other popular deep learning models, including AlexNet. Experimental results show that our method achieves an accuracy of over 93%, higher than pure deep learning models.

Keywords – CT images, AlexNet, MIFS, SVM.

1. Introduction

Early detection of brain abnormalities is essential. This not only helps identify brain diseases but also helps plan appropriate treatment, minimizing the risk of serious complications and death. A simple and popular method to identify brain-related diseases is computed tomography. Each type of disease leaves abnormal signs on the images, and based on these signs, researchers have developed methods to apply advanced technology to detect abnormalities early, thereby reducing the burden on the healthcare system. Recently, deep learning algorithms have been widely used because of their ability to detect abnormalities in the brain quickly and accurately. Image recognition algorithms based on convolutional neural networks are capable of learning and extracting complex image features effectively. Therefore, many researchers have applied CNNs to detect brain abnormalities from medical images.

In the paper [1], the authors proposed a deep learning model with the ability to diagnose Alzheimer's syndrome and schizophrenia through patient time-series EEG data. From the synthesis of EEG time-series data, the authors converted it into a brain region connectivity matrix using three methods (Granger relation, Pearson correlation, Spearman correlation) and then used it as a dataset for the deep learning model with optimal algorithms. The model proposed by the authors achieved a maximum accuracy of 92% when diagnosing Alzheimer's syndrome and 52% when diagnosing schizophrenia with the corresponding matrix formation methods of Pearson and Granger. In the paper [2], the authors studied the feasibility of applying deep learning to diagnose fetal brain abnormalities using ultrasound images. With a hospital image dataset of 10.251 normal brain images and 2.529 abnormal brain images, through data enrichment and training, comparing the conclusions with previous diagnoses of doctors and experts, the studied model achieved an accuracy of 96.3%. In the article [3], the authors proposed a deep learning model capable of supporting

DOI: https://doi.org/10.70862/CSIR.2024.0101-06

Corresponding author: Dimitar Borisov,
University of Chemical Technology and Metallurgy
Email: shtain@uctm.edu

Received: 4 September 2024.
Revised: 18 November 2024.
Accepted: 27 November 2024.
Published: 28 November 2024.

© 2024 Dinh Cong Tung, Mai Van Linh, Dimitar Borisov. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The article is published with Open Access at http://csirjournal.com/

the recognition of undetected abnormalities from MRI scans of newborns. Using the input image preprocessing method, training a 3D automatic decoding structure to analyze brain features, reapply the analyzed features, and highlight abnormalities, the research team built a model to analyze and recognize abnormalities with an AUC of 83%. The deep neural model "ASD-SAENet" for diagnosing bipolar depression disorder proposed by the authors of the article [4] uses the SAE algorithm to optimize the ability to separate features used for the classification process. Using the dataset of fMRI images "ABIDE", the model achieved an accuracy of 70.8%. The authors of the article [5] proposed a deep learning model using EEG signal data processed by a combination of CNN and LSTM models to identify EEG rhythm frequencies similar to bipolar depression disorder, achieving an accuracy of 96.43%. To solve the risk of cerebral hemorrhage, the authors of the article [6] suggested a model to assist in identifying cerebral hemorrhage using CT scans of patients. Using the MobileNetV2 architecture trained on the "RSNA Intracranial Hemorrhage" dataset, CT scans indicating potential brain hemorrhage risks are flagged and processed through the Hounsfield Unit scale and the DBSCAN method to emphasize risk areas and reduce noise. This model achieved an AUC of 99.1%. In a separate study, the authors of article [7] proposed a brain tumor diagnosis system based on MRI images, utilizing a deep convolutional neural network optimized with an ADAS method. The process involves simplifying the input images by removing unnecessary details, which are then used to train the deep neural network to identify risk factors and classify the images. This model achieved an accuracy of 94%. The paper [8] applied trained deep learning strategies such as VGG16 and ResNet50 to classify brain MRI images using multi-class classifications with groups of Normal, Stroke, Low-grade Neuroma, and High-grade Neuroma. The authors used different classifiers such as SoftMax, SVM-RBF, and SVM-Cubic to compare the results between them. The authors of the paper [9] conducted a feasibility test of abnormality detection in fetal brain ultrasound images using deep learning algorithms by training the model with 12780 real images from the hospital database. The results of the project can also accurately localize the abnormal detection area. The paper [10] proposed a ResNet-based deep learning model capable of classifying brain MRI images to support doctors in reducing the volume of images that need to be examined in depth. Data enrichment methods, learning rate optimization, and appropriate tuning were applied in the model-building process. In 613 MRI images, this model achieved 100% accuracy according to the authors, and is ready to be tested with larger data volumes. In order to build a deep learning model to detect brain tumors and classify brain tumors, the authors of the article [11] proposed a model to identify and classify brain tumors from MRI images using deep learning methods. The model applied transfer learning techniques and combined VGG16 with the "23-layer CNN" block to improve the accuracy of brain tumor classification. The proposed model successfully achieved 97.8% accuracy. The authors of the article [12] proposed three deep learning methods to analyze the impact of local features and deep features on detecting abnormalities in brain MRI images. The first method uses a deep autoencoder using directional bit planes to differentiate and learn local and directional features. The second method uses a separable scaling convolutional neural network to extract deep and shallow features in the image. The final method is a combination of DBP-DAE and DSRCN. The results are comparable to the popular deep learning algorithm. A hybrid deep learning model between a deep neural network and a Random Forest classifier is proposed by the authors of the paper [13]. The hybrid approach helps the model compete with simple DNN and DNN+SVM models, achieving training AUC of 94% and validation of 87%.

2. Motivation

In this paper, we present a novel method to improve the accuracy of detecting brain abnormalities from computed tomography images. The collected data is characterized by a large amount of random noise, which can degrade the image quality and affect the analysis process. Therefore, the first step is to perform image preprocessing to improve the input quality. We use the Histogram Equalization technique to balance the brightness and enhance the contrast of the image, helping to highlight important structures. Gabor filters are then applied to enhance edge features, helping to emphasize details and complex textures in the image. After preprocessing, we proceed to extract features using the AlexNet model. From the fully connected layer of AlexNet, we obtain 4096 features, each representing a different aspect of the image data. These features contain rich information in the data. However, not all of these features are equally valuable in distinguishing between normal and abnormal brain states. Therefore, to optimize the classification process and reduce the computational load, we apply the Mutual Information Feature Selection (MIFS) algorithm. MIFS helps to remove redundant or irrelevant features, retaining only the most important and discriminating features. With the selected optimal feature set, we use the Support Vector Machine (SVM) machine learning model to classify normal and abnormal brain states.

Experimental results show that our method achieves an accuracy of 93.68%, which is significantly higher than the pure AlexNet model and other deep learning models such as VGG-16 and DenseNet-121. Figure 1 illustrates an overview of the brain abnormality detection process that we propose.

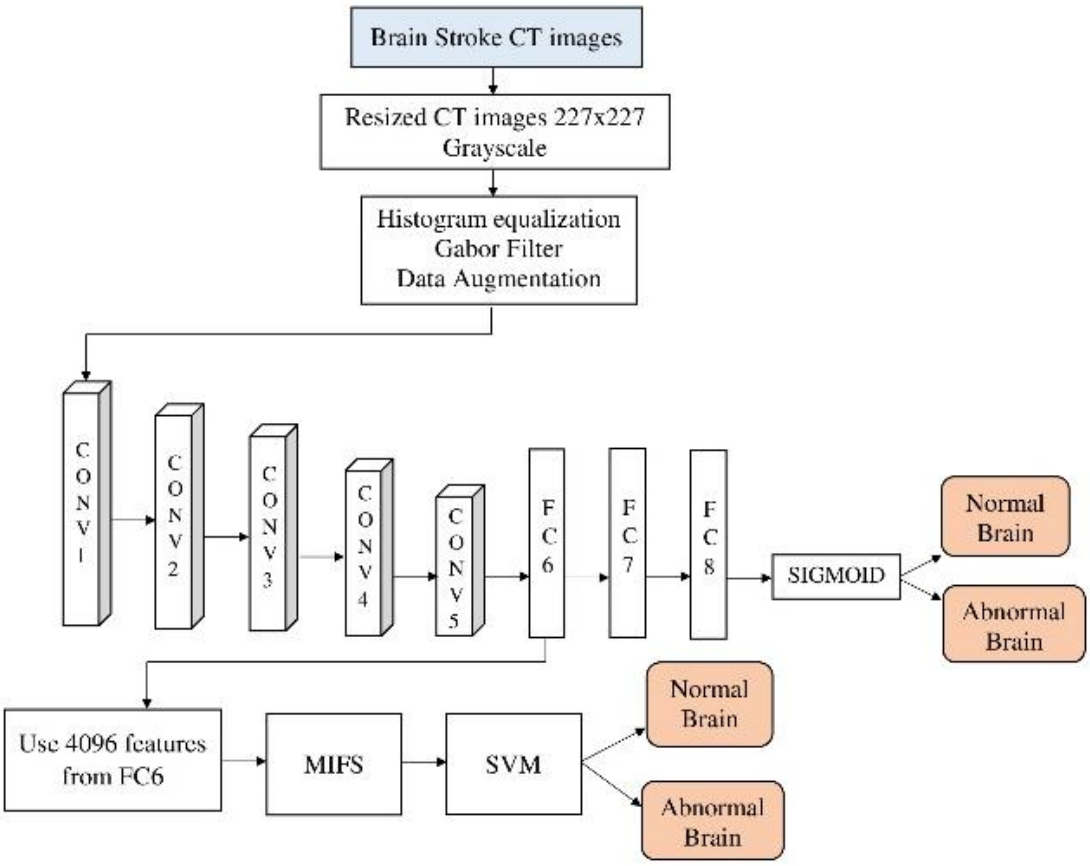


Figure 1. Overall processing workflow in the paper

3. Methodology

3.1. Histogram Equalization

Histogram Equalization is an image preprocessing method used in our paper to enhance the contrast of computed tomography images, in order to more accurately detect brain abnormalities. This technique redistributes the intensity values of the image so that the entire intensity range is used more evenly, helping to highlight the details in the image [14]. The Histogram Equalization algorithm can be described by formula (1).

$$H': I'(x,y) = H'(I(x,y)) \qquad (1)$$

where, $H'(i) = \sum_{j=0}^{i} H(j)$, is the Histogram normalized to $[0, 255]$, $H(i)$ is the Histogram of the original image, $I(x,y)$ is the original image, $I'(x,y)$ is the Histogram normalized image. This technique provides clearer and more informative input data, which helps to improve the feature extraction ability and the performance of the next classification steps in the anomaly detection process.

3.2. Gabor Filter

Gabor Filter is an image processing technique used in our study to extract spatial and frequency features from CT images, which improves the detection of brain abnormalities. Gabor filter simulates the responses of the human visual system by combining a Gaussian function with a harmonic wave function, allowing it to capture local features such as edges, corners, and textures with sensitivity to frequency and orientation [15], as shown in the formula (2).

$$g(x, y, \lambda, \theta, \varphi, \sigma, \gamma) = e\left(\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right) e(i(2\pi\frac{x}{\lambda} + \varphi)) \qquad (2)$$

where, $x' = x\cos\Theta + y\sin\Theta$ and $y' = -x\sin\Theta + y\cos\Theta$, λ denotes the wavelength of the sinusoidal coefficient, θ denotes the direction of the normal of the Gabor function, ϕ is the phase shift, σ is the standard deviation of the Gaussian envelope, and γ is the phase shift of the Gabor function. In this paper, we extract Gabor features in 16 directions. By applying Gabor Filter on CT images, we can extract features related to direction and frequency, especially details

such as the structure and border of the brain. These features are used as input for the next steps in the feature selection process.

3.3. AlexNet Model

AlexNet [16] is a deep learning model used in our study to extract features from computed tomography images, and is also a model used to detect brain abnormalities. The structure of AlexNet consists of five convolutional layers and three fully connected layers. AlexNet uses techniques such as ReLU, dropout, and local normalization to improve performance. The convolutional layers consist of 96 11x11 filters in the first layer, followed by layers with 256 5x5 filters, 384 3x3 filters, and 256 3x3 filters, all interleaved with ReLU and pooling layers. The fully connected layers consist of two layers of 4096 neurons and an output layer of 2 neurons, using a sigmoid function to classify normal and abnormal brains. The output from the first fully connected layer is a feature vector of 4096 values, which is used as input for feature selection using MIFS and then classification using SVM. Therefore, AlexNet not only helps detect brain abnormalities, but also provides useful features for subsequent processing steps, contributing to improving the accuracy of the classification system.

3.4. MIFS algorithm

Mutual Information Feature Selection (MIFS) [17] is a feature selection technique used in our model to optimize the input feature set, extracted from the AlexNet model. MIFS works on the concept of mutual information, which measures the degree of dependence between variables. The goal of MIFS is to select the features with the highest discriminative power, while minimizing the information redundancy between the features, which is described by formula (3).

$$J(S,f) = I(f;c) - \beta \sum_{s \in S} I(f;s) \qquad (3)$$

where, $J(S,f)$ is the value of feature f when considering the set of selected features S. $I(f;c)$ is the mutual information between feature f and the target variable. By optimizing this objective function, MIFS helps to select a set of features with the highest representativeness and least redundancy, which helps to improve the accuracy and efficiency of the data before using the SVM model to classify brain abnormalities from CT images.

3.5. SVM machine learning model

Support Vector Machine (SVM) [18] is a machine learning algorithm used in our model to classify brain abnormalities from preprocessed and feature-extracted CT image data. SVM works by searching for a hyperplane in the feature space, with the goal of maximizing the margin between data classes, in this case between normal and abnormal brain states. Specifically, SVM determines the hyperplane through formula (4).

$$w.x + b = 0 \qquad (4)$$

where, w is the weight vector, x is the feature vector of the data sample, b is the bias correction coefficient. The objective function of the SVM to be optimized is $\frac{1}{2}||w||^2$ with the constraint $y_i(w.x_i + b) \geq 1 - \delta_i$, where y_i is the class label, x_i is the data point and δ_i is the slack variable allowing some points to violate the boundary. The combination of SVM with the features selected through the MIFS algorithm ensures that our model achieves high accuracy in detecting brain abnormalities.

4. Results

Data in the paper

In this paper, we use the Afridi Rahman dataset, which consists of 1900 CT images, including 950 normal brain images, and 950 abnormal brain images of stroke [19]. These images are taken in cross-section, each image representing a different depth of the brain. Figure 2 illustrates some of the data in the paper, with the first row depicting images of normal brains, and the second row depicting abnormal brain images of stroke on the same CT scan slice.

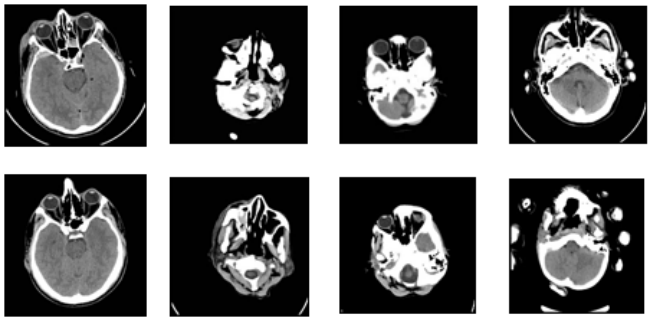


Figure 2. Some data collected in the paper

The images are preprocessed using white balance, Gabor filter and data enrichment techniques as presented in Section 2. After preprocessing, we split the data into two parts, 80% for training and 20% for testing, then put them into the AlexNet model and MIFS algorithm for feature extraction, then classified with the SVM machine learning model.

5. Discussion

Experimental results

Our proposed method was implemented and trained on a computer with a Core i7-10700 2.9GHz

configuration, 64GB RAM, Windows 10 operating system, and Python 3.6 development environment. The training parameters for the models include Learning Rate = 1e-3, batch size = 64, and number of epochs = 100. We conducted experiments with four deep learning models including the proposed model, VGG-16, AlexNet, and DenseNet. The accuracy of each model during training is shown in Figure 3. The results show that our method has outstanding performance. Specifically, in Figure 3a, the accuracy of the proposed model reaches a stable level from the 20th epoch, with the difference between the training and testing values being insignificant, and the Test values fluctuating little, demonstrating the good convergence ability of the model. The VGG-16, AlexNet, and DenseNet models, shown in Figures 3b, 3c, and 3d, respectively, show that the achieved accuracy is 92.1%, 90%, and 72.63%, respectively. The proposed model not only achieves the highest accuracy but also maintains stability and efficiency during training.

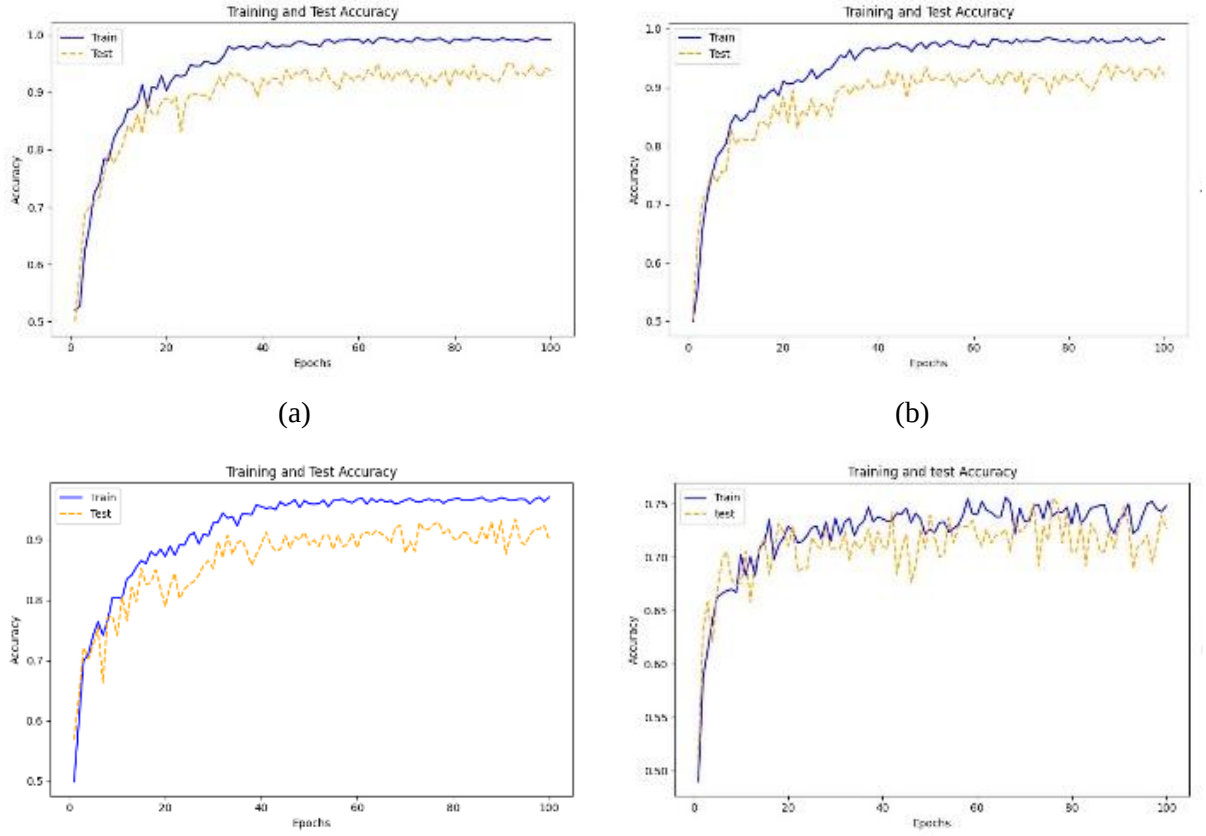
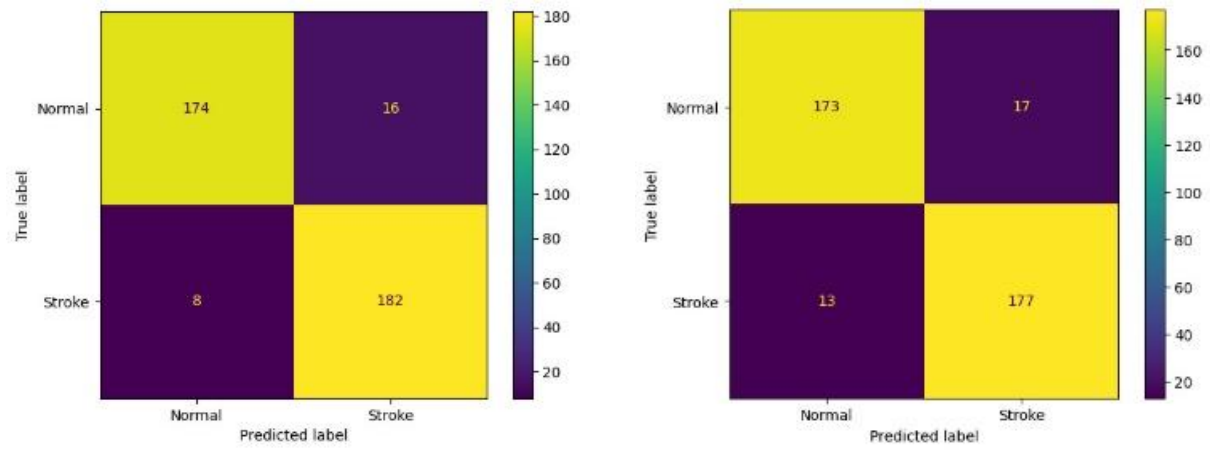


Figure 3. Training accuracy of four models

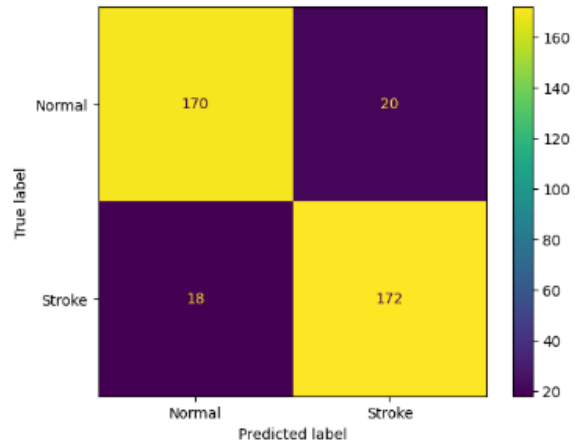
To provide a more intuitive view of the prediction accuracy on the test dataset in the proposed model, we used the Confusion matrix. In this matrix, the vertical axis represents the actual label, which includes two classes: Normal Brain and Abnormal Brain, the horizontal axis represents the model's predicted label, which also corresponds to these two classes. The Confusion matrix for the proposed model is presented in Figure 4a, showing the distribution of correct and incorrect predictions. Similarly, the Confusion matrices for the VGG-16, AlexNet, and DenseNet models are presented in Figures 4b, 4c, and 4d, respectively.

To further demonstrate the effectiveness of the proposed method, we conducted statistical and comparison analyses of six important metrics during the training of the models, including Accuracy, Precision, Sensitivity, Specificity, G-mean, and F1-Score. These metrics provide a comprehensive view of the model's classification ability and are the basis for evaluating the superiority of the proposed method over other deep learning models. Specifically, Accuracy reflects the proportion of correctly predicted cases for both normal and abnormal brain classes on the total data of the test set, calculated by formula (5).

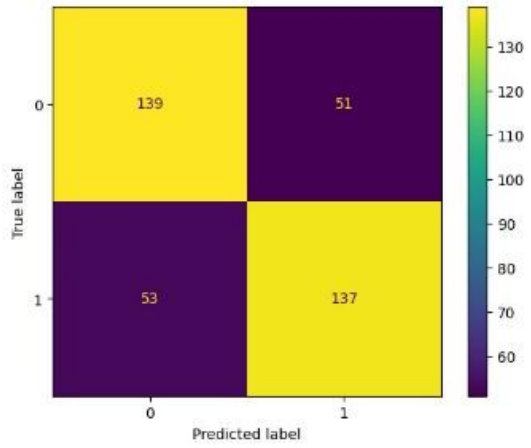


(a)

(b)



(c)



(d)

Figure 4. Confusion Matrix of Four Models

Precision measures the accuracy in predicting abnormal brain cases, calculated as the ratio of the number of correct predictions as abnormal to the total number of abnormal predictions, according to formula (6). Sensitivity (also known as Recall) evaluates the model's ability to detect brain abnormalities, calculated as the ratio of the number of correct predictions as abnormal to the total number of actual abnormal cases, according to formula (7). Specificity measures the ability to correctly identify normal brain cases, calculated as the ratio of the number of correct predictions as normal to the total number of actual normal cases, according to the formula (8). The G-mean value is a comprehensive performance measure that balances Sensitivity and Specificity, representing the model's ability to not only correctly detect brain abnormalities but also correctly classify normal brain cases. G-mean is calculated according to formula (9) as the geometric mean of Sensitivity and Specificity. Finally, F1-Score is the harmonic average of Precision and Sensitivity, which provides an overall measure of the model's accuracy considering both accuracy and sensitivity, which helps evaluate the model in handling difficult-to-classify cases, calculated by formula (10). The results of comparing these metrics between the proposed method and other deep learning models, including VGG-16, AlexNet, and DenseNet, are presented in Table 1. From this analysis, we find that the proposed method achieves the highest accuracy, while maintaining a good balance between Sensitivity and Specificity, as evidenced by the G-mean and F1-Score values. This demonstrates that our method outperforms other deep learning models in detecting and classifying brain abnormalities, while minimizing classification errors.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Sensitivity = \frac{TP}{TP + FN} \quad (7)$$

$$Specificity = \frac{TN}{TN + FP} \quad (8)$$

$$G - mean = \sqrt{Sensitivity * Specificity} \quad (9)$$

$$F - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (10)$$

Table 1. Statistical table of post-training values of four deep learning models

Model	Proposed model	VGG16	AlexNet	DenseNet
Accuracy	93.68%	92.10%	90.0%	72.63%
Precision	95.60%	93.01%	90.42%	72.39%
Sensitivity	91.57%	91.05%	89.47%	73.15%
Specificity	95.78%	93.15%	90.52%	72.10%
G-mean	93.66%	92.09%	89.99%	72.62%
F1-Score	93.54%	92.02%	89.94%	72.77%

6. Conclusion

In this paper, we present a novel architecture for brain anomaly detection based on computed tomography (CT) images. To improve the image quality and highlight the brain features, we apply preprocessing methods including Histogram Equalization and Gabor Filter. The AlexNet deep learning model is not only used to detect the anomaly with an accuracy of 90% but also plays an important role in extracting 4096 features from the dataset. These features are then filtered and optimized by the MIFS algorithm, to select the most prominent features. The final classification is performed using the SVM machine learning model. Experimental results show that the proposed model achieves an accuracy of 93.68%, which outperforms pure deep learning models. However, the current study is still unable to classify different types of brain abnormalities, and the images used are CT scans instead of MRI. While CT imaging speeds up the diagnostic process, MRI provides greater reliability. Therefore, in future studies, we plan to collect data from MRI and develop and refine the deep learning model to improve the accuracy and detection of other types of brain abnormalities. The goal is to improve diagnostic efficiency, and support earlier and more accurate detection of brain abnormalities, thereby contributing to the advancement of medicine.

Acknowledgements

This research is funded by University of Transport and Communications (UTC) under grant number T2024-CN-004.

References

[1] Alves, C. L., Pineda, A. M., Roster, K., Thielemann, C., & Rodrigues, F. A. (2022). EEG functional connectivity and deep learning for automatic diagnosis of brain disorders: Alzheimer's disease and schizophrenia. Journal of Physics: complexity, 3(2), 025001. https://doi.org/10.1088/2632-072X/ac5f8d

[2] Xie, H. N., Wang, N., He, M., Zhang, L. H., Cai, H. M., et al. (2020). Using deep-learning algorithms to classify fetal brain ultrasound images as normal or abnormal. Ultrasound in Obstetrics & Gynecology, 56(4), 579-587. https://doi.org/10.1002/uog.21967

[3] Raad, J. D., Chinnam, R. B., Arslanturk, S., Tan, S., Jeong, J. W., & Mody, S. (2023). Unsupervised abnormality detection in neonatal MRI brain scans using deep learning. Scientific Reports, 13(1), 11489. https://doi.org/10.1038/s41598-023-38430-0

[4] Almuqhim, F., & Saeed, F. (2021). ASD-SAENet: a sparse autoencoder, and deep-neural network model for detecting autism spectrum disorder (ASD) using fMRI data. Frontiers in Computational Neuroscience, 15, 654315. https://doi.org/10.3389/fncom.2021.654315

[5] Al-Ezzi, A., Yahya, N., Kamel, N., Faye, I., Alsaih, K., & Gunaseli, E. (2021). Severity assessment of social anxiety disorder using deep learning models on brain effective connectivity. IEEE Access, 9, 86899-86913. https://doi.org/10.1109/ACCESS.2021.3089358

[6] Luong, K. G., Duong, H. N., Van, C. M., Thi, T. H. H., Nguyen, T. T., Thoai, N., & Thi, T. T. T. (2021). A computer-aided detection to intracranial hemorrhage by using deep learning: a case study. In Computational Intelligence Methods for Green Technology and Sustainable Development: Proceedings of the International Conference GTSD2020 5 (pp. 27-38). Springer International Publishing. https://doi.org/10.1007/978-3-030-62324-1_3

[7] Han-Trong, T., Nguyen Van, H., Nguyen Thi Thanh, H., Tran Anh, V., Nguyen Tuan, D., & Vu Dang, L. (2022). An efficient method for diagnosing brain tumors based on MRI images using deep convolutional neural networks. Applied Computational Intelligence and Soft Computing, 2022(1), 2092985. https://doi.org/10.1155/2022/2092985

[8] Kadry, S., Nam, Y., Rauf, H. T., Rajinikanth, V., & Lawal, I. A. (2021). Automated detection of brain abnormality using deep-learning-scheme: a study. In 2021 Seventh international conference on bio signals, images, and instrumentation (ICBSII) (pp. 1-5). IEEE. https://doi.org/10.1109/ICBSII51839.2021.9445122

[9] Xie, H. N., Wang, N., He, M., Zhang, L. H., Cai, H. M., et al. (2020). Using deep-learning algorithms to classify fetal brain ultrasound images as normal or abnormal. Ultrasound in Obstetrics & Gynecology, 56(4), 579-587. https://doi.org/10.1002/uog.21967

[10] Talo, M., Baloglu, U. B., Yıldırım, Ö., & Acharya, U. R. (2019). Application of deep transfer learning for automated brain abnormality classification using MR images. Cognitive Systems Research, 54, 176-188. https://doi.org/10.1016/j.cogsys.2018.12.007

[11] Khan, M. S. I., Rahman, A., Debnath, T., Karim, M. R., Nasir, M. K., et al. (2022). Accurate brain tumor detection using deep convolutional neural network.

Computational and Structural Biotechnology Journal, 20, 4733-4745. https://doi.org/10.1016/j.csbj.2022.08.039

[12] Einy, S., Saygin, H., Hivehch, H., & Dorostkar Navaei, Y. (2022). Local and deep features based convolutional neural network frameworks for brain MRI anomaly detection. Complexity, 2022(1), 3081748. https://doi.org/10.1155/2022/3081748

[13] Shinde, K., & Thakare, A. (2021). Deep hybrid learning method for classification of fetal brain abnormalities. In 2021 International Conference on Artificial Intelligence and Machine Vision (AIMV) (pp. 1-6). IEEE. https://doi.org/10.1109/AIMV53313.2021.9670994

[14] Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., et al. (1987). Adaptive histogram equalization and its variations. Computer vision, graphics, and image processing, 39(3), 355-368. https://doi.org/10.1016/S0734-189X(87)80186-X

[15] Gabor, D. (1946). Theory of communication. Journal of the Institution of Electrical Engineers, 93(26), 429-457. https://doi.org/10.1049/ji-3-2.1946.0076

[16] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.

[17] Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence, 27(8), 1226-1238. https://doi.org/10.1109/TPAMI.2005.159

[18] Cortes, C., Vapnik, V. (1995) Support-vector networks. Machine Learning, 20, 273-297. https://doi.org/10.1007/BF00994018

[19] Afridi, R., Stroke, B. (2021). CT Image Dataset.

Computer Science and Interdisciplinary Research Journal
Volume 1, Issue 2, 2024, ISSN: 3033-1218, DOI: 10.70862/CSIR.2024.0102-07

Numerical Analysis of the Magnetic Moment in S/F/S-Josephson Junction

Elis Dieva

Paisii Hilendarski University of Plovdiv, 24 Tzar Asen, 4000 Plovdiv, Bulgaria

dievaelis@gmail.com

Abstract – One particularly intriguing object in superconductivity is the Josephson junction, which is an integral part of quantum computer hardware. Analysing this object requires solving differential equations. Such problems often cannot be solved with analytical methods; therefore, numerical methods and computational mathematics must be utilized. This article explores a fundamental model of the Josephson junction. The model presents the spatial coordinates of the magnetic moment as unknown functions dependent on time in a system of three ordinary differential equations with initial conditions. Comparative analysis of various methods is conducted for different sets of parameters. Optimal numerical methods for these particular instances of the problem are found. Visual results are also presented.

Keywords – numerical analysis, Josephson junction, magnetic moment.

1. Introduction

Modern science and technology are based on the modelling and analysis of complex systems representing physical, chemical, economic, and biological processes. These systems, which are defined by motion in space over time are often described by differential equations. In this context the Cauchy problem for systems of ordinary differential equations (ODEs) is one of the fundamental problems providing a framework for studying the dynamic behaviour of different systems [1]. Moreover, to present reliable results, they must meet conditions related to their accuracy. Implicit numerical methods known for their ability to handle stiff problems due to their stability and increased accuracy play an important role in numerical analysis.

Most commonly, a Josephson junction (Fig. 1) is composed of two superconductors separated by a thin layer of dielectric or ferromagnetic material [2]. It allows the passage of a superconducting current without voltage and energy loss.



Figure 1. Ferromagnetic Josephson junction (S/F/S)

When a current flows through a Josephson junction, it generates a magnetic field in the immediate vicinity of the junction [3]. The magnetic moment in the Josephson junction arises from the formation of currents known as Josephson currents which can create a magnetic field around the junction [4]. The magnetic moment is a physical quantity that describes the magnetic properties of particles, bodies, or systems. It is related to the distribution of magnetic charge or the movement of charged particles.

φ_0 and an S/F/S (superconductor/ferromagnet/ superconductor) Josephson junctions differ primarily in the mechanisms driving their behaviour [5, 6].

A φ_0-junction arises from broken inversion or time-reversal symmetry, typically due to spin-orbit coupling and an applied magnetic field, leading to a phase offset. In contrast, an S/F/S junction relies on the magnetic exchange field in the ferromagnetic layer, which induces oscillatory superconducting correlations and can stabilise ground states with phase differences of 0 or π. However, an S/F/S junction can exhibit φ_0-like behaviour when additional symmetry-breaking effects, such as noncollinear magnetization or spin-orbit coupling, are introduced. These effects

DOI: https://doi.org/10.70862/CSIR.2024.0102-07

Corresponding author: Elis Dieva,
University of Plovdiv Paisii Hilendarski
Email: dievaelis@gmail.com

Received: 4 December 2024.
Revised: 12 January 2025.
Accepted: 12 January 2025.
Published: 13 January 2025.

BY-NC-ND © 2024 Elis Dieva. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The article is published with Open Access at http://csirjournal.com/

effectively merge the characteristics of both systems. This is the reason why in some cases the terms can be interchangeable.

In summary, the magnetic moment is an important concept in physics that describes how particles and systems interact with magnetic fields and how they create magnetic phenomena. The precession of the magnetic moment also plays an important role in the field of spintronics.

The magnetic moment in this S/F/S model of the Josephson junction can be described by a system of ordinary differential equations that requires initial conditions to be solved [7]. Through this Cauchy problem, the solutions that reflect the behaviour of the physical system are determined, taking into account external influences and parameters.

The primary mathematical model considers the phase difference of the Josephson junction as a linear function of time.

2. Problem Formulation

Let us assess the fundamental mathematical model of a Josephson junction, described by a system of three ODEs [7, 8].

$$\frac{dm_x}{dt} = \frac{\omega_F}{1+(M\alpha)^2}(-m_z m_y + Grm_z sin(\omega t - rm_y) - \alpha[m_x m_z^2 + Grm_x m_y sin(\omega t - rm_y)])$$

$$\frac{dm_y}{dt} = \frac{\omega_F}{1+(M\alpha)^2}(m_x m_z - \alpha[m_y m_z - Gr(m_z^2 + m_x^2) sin(\omega t - rm_y)])$$

$$\frac{dm_z}{dt} = \frac{\omega_F}{1+(M\alpha)^2}(-Grm_x sin(\omega t - rm_y) - \alpha[Grm_y m_z sin(\omega t - rm_y) - m_z(m_y^2 + m_x^2)])$$

where the initial conditions are
$m_x(0) = 0,\ m_y(0) = 0,\ m_z(0) = 1.$

The unknown functions here (m_x, m_y and m_z) are the spatial coordinates of the magnetic moment, represented as functions of time. As can be seen, the problem is multi-parameter with the following parameters:

- ω_F: resonance frequency of the ferromagnet;
- α: damping parameter;
- G: parameter of phase difference coupling;
- r: spin-orbit coupling parameter;
- ω: ratio of the phase difference to time;
- M: norm of the magnetic moment.

3. Numerical Approach

For the numerical solving of the system in this investigation, numerous program modules have been developed using the computational software Wolfram Mathematica [9, 10]. They implement a wide variety of numerical methods such as the Euler method, Runge-Kutta in their explicit and implicit forms of different orders, as well as the Adams methods in their explicit and implicit forms [11, 12].

Gauss-Legendre methods, which have been used in this paper, are a specific subset of implicit Runge-Kutta methods derived from the collocation of differential equations at Gauss-Legendre quadrature points. This choice makes Gauss-Legendre methods capable of achieving high-order accuracy. The Adams methods included in this paper are chosen so that the number of the previous steps used for computation is identical. This is in order to use the implicit method in combination with the explicit in the predictor-corrector manner [13].

Table 1. Summary of the used methods

Method Name	Notation	Global Error of the Method	Characteristics
Euler	Euler	$\mathcal{O}$(h)	Single step
Runge-Kutta 2nd Order	RK2	$\mathcal{O}$(h²)	Explicit, two intermediate steps
Runge-Kutta 4th Order	RK4	$\mathcal{O}$(h⁴)	Explicit, four intermediate steps
Gauss-Legendre 4th Order	GL4	$\mathcal{O}$(h⁴)	Implicit, increased stability, two steps
Gauss-Legendre 6th Order	GL6	$\mathcal{O}$(h⁶)	Implicit, increased stability, three steps
Explicit Adams 4th Order	EA4	$\mathcal{O}$(h⁴)	Uses 4 previous derivatives
Implicit Adams 4th Order	IA4	$\mathcal{O}$(h⁴)	Uses interpolation for refinement

A summary of the used methods can be seen in Table 1. Implicit numerical methods, renowned for their stability and improved accuracy when dealing with stiff problems, are a focal point in this numerical analysis. The presented work represents an extension of the methods and visualization approaches proposed in [14].

4. Numerical Results

Let us consider the efficiency of various numerical methods with the following parameters: $G = 20, r = 0.1, \alpha = 0.1, \omega = 1, \omega_F = 1, t = 50$. Table 2 shows the expected global theoretical errors T_{err}, the actual error value obtained from the calculations (A_{err}), and the computation time $CT(s)$ for different step sizes h.

The study shows that the best methods are those that apply additional cycles to refine results on a local level (GL4 and GL6). These methods achieve accuracy on the order of 14 decimal places, even with larger step sizes (the step size of the uniform grid used is denoted by h), unlike explicit methods, which work well with smaller step sizes but increase the computation time ($CT(s)$) in seconds.

Table 2. Comparison of the numerical methods with different steps for the parameters $G = 20, r = 0.1, \alpha = 0.1, \omega = 1, \omega_F = 1, t = 50$

Method	h=0.1			h=0.01			h=0.001		
	T_{err}	A_{err}	$CT(s)$	T_{err}	A_{err}	$CT(s)$	T_{err}	A_{err}	$CT(s)$
Euler	10^{-1}	2.9	0.4	10^{-2}	3.5×10^{-1}	0.64	10^{-3}	8.7×10^{-2}	14
RK2	10^{-2}	2.9×10^{-2}	0.1	10^{-4}	2.8×10^{-5}	1.2	10^{-6}	2.8×10^{-8}	19
RK4	10^{-4}	6×10^{-5}	0.2	10^{-8}	1.4×10^{-9}	2	10^{-12}	6.4×10^{-14}	34
GL4	10^{-4}	1.8×10^{-10}	1	10^{-8}	1.7×10^{-15}	11	10^{-12}	6.3×10^{-16}	60
GL6	10^{-6}	1.28×10^{-14}	2	10^{-12}	3.3×10^{-16}	16	10^{-18}	4.4×10^{-15}	119
EA4	10^{-4}	9×10^{-3}	0.1	10^{-8}	2×10^{-7}	2	10^{-12}	1.3×10^{-11}	115
IA4	10^{-4}	4×10^{-4}	0.3	10^{-8}	1.2×10^{-8}	4	10^{-12}	9.2×10^{-13}	167

The comparison of the values of the magnetic moment coordinates found for the set of parameters used in the table can be seen in Fig. 2 through Fig. 7, where the horizontal lines represent the magnetic norm. Also, it can be observed that after an initial imbalance, the system settles into its new state.

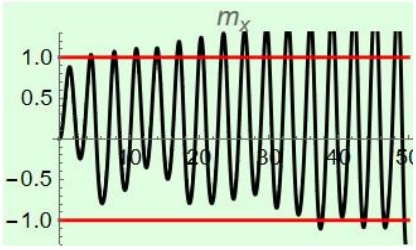


Figure 2. Values of the spatial m_x-coordinate using the Euler method, where h=0.01

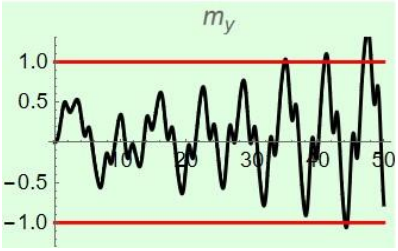


Figure 3. Values of the spatial m_y-coordinate using the Euler method, where h=0.01

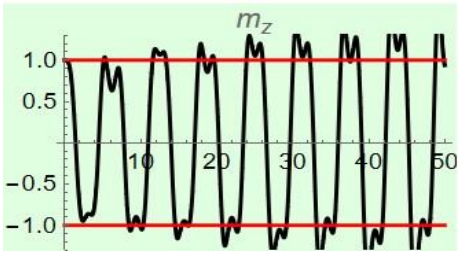


Figure 4. Values of the spatial m_z-coordinate using the Euler method, where h=0.01

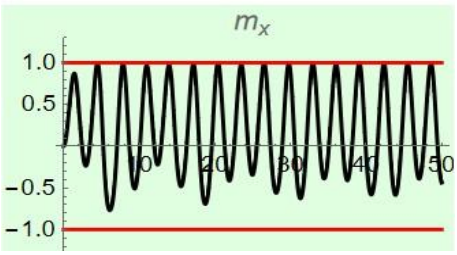


Figure 5. Values of the spatial m_x-coordinate using the Runge-Kutta method, where h=0.01

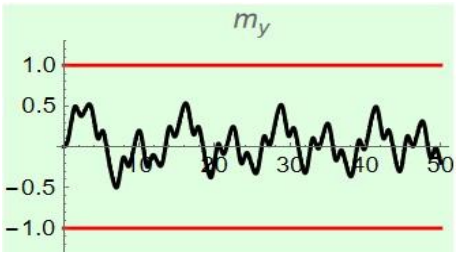


Figure 6. Values of the spatial m_y-coordinate using the Runge-Kutta method, where h=0.01

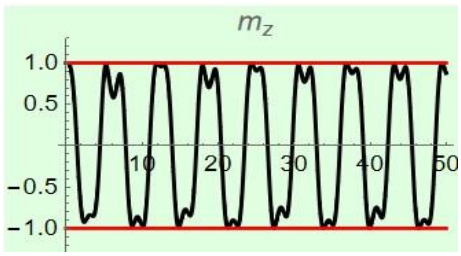


Figure 7. Values of the spatial m_z-coordinate using the Runge-Kutta method, where h=0.01

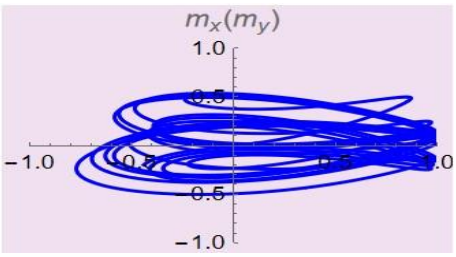


Figure 8. Precession of the magnetic moment in the $m_x(m_y)$ plane

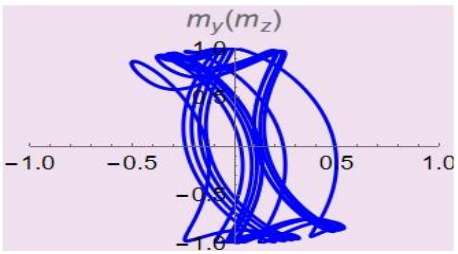


Figure 9. Precession of the magnetic moment in the $m_y(m_z)$ plane

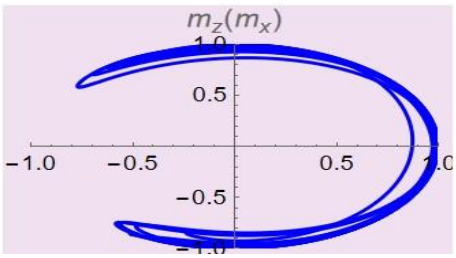


Figure 10. Precession of the magnetic moment in the $m_z(m_x)$ plane

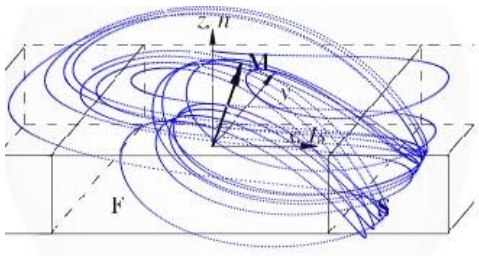


Figure 11. Precession of the magnetic moment in space

The precession of the magnetic moment for the parameters $G = 20, r = 0.1, \alpha = 0.1, \omega = 1, \omega_F = 1, t = 50$ where h=0.01 using the Runge-Kutta method is showcased in Fig. 8 through Fig. 11. Let us examine the numerical results with a different set of parameters. A solution with the following set of parameters $G = 90, r = 0.1,\ \alpha = 0.1, \omega = 0.8,\ \omega_F = 0.8, t = 50$ is presented. After performing the calculations using all methods, we obtain the following results, shown in Table 3.

In this case, it is clear that the chosen set of parameters creates a stiff problem [8,14]. This presents a challenge for numerical methods due to the rapid accumulation of local errors. Trivial methods like the Euler method and EA4 result in the so-called overflow – an inability to continue calculations due to the computer's limitations and computations with excessively large numbers. It can also be seen that even the implicit methods do not meet the expected order of accuracy. However, such cases allow us to assess which methods are more stable when dealing with stiffness. Once again, GL4 and GL6 show the best results in computational accuracy in relation to computational time compared to the other methods used. Also greater accuracy is obtained with a larger step size and relatively less processing capacity. These cases also provide a basis for exploring other, more efficient numerical methods.

In Fig. 12 through Fig. 14 we see that the behaviour of the system seems periodic and shows no signs of settling in the designated period of time. The parameters used are as in Table 3. Also, the rapid changes of the values of the magnetic moment coordinates explain the inability of the trivial methods to cope with the accumulation of local errors. Figure 15. through Fig. 18 show the magnetic moment in the planes and in space.

The precession of the magnetic moment for the parameters $G = 20, r = 0.1, \alpha = 0.1, \omega = 1, \omega_F = 1, t = 50$ where h=0.01 are showcased in Fig. 8 through Fig. 10.

Table 3. Comparison of the numerical methods with different steps for the parameters $G = 90, r = 0.1, \alpha = 0.1, \omega = 0.8, \omega_F = 0.8, t = 50$

Method	h=0.1			h=0.01			h=0.001		
	T_{err}	A_{err}	$CT(s)$	T_{err}	A_{err}	$CT(s)$	T_{err}	A_{err}	$CT(s)$
Euler	10^{-1}	–	–	10^{-2}	6	0.75	10^{-3}	1.4	15
RK2	10^{-2}	7	0.1	10^{-4}	1.7×10^{-2}	1.7	10^{-6}	1.9×10^{-5}	21
RK4	10^{-4}	0.29	0.2	10^{-8}	2.4×10^{-5}	2.3	10^{-12}	1.9×10^{-10}	34
GL4	10^{-4}	3.5×10^{-8}	1.9	10^{-8}	2.6×10^{-12}	10	10^{-12}	3×10^{-11}	88
GL6	10^{-6}	6.7×10^{-12}	3	10^{-12}	1.1×10^{-15}	20	10^{-18}	8.8×10^{-14}	137
EA4	10^{-4}	–	–	10^{-8}	2.7×10^{-4}	2	10^{-12}	2.2×10^{-8}	120
IA4	10^{-4}	2.5×10^{-2}	0.7	10^{-8}	3.3×10^{-5}	7	10^{-12}	3.3×10^{-10}	112

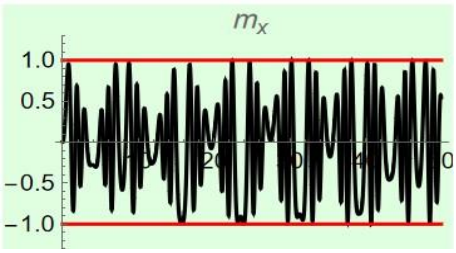


Figure 12. Values of the spatial m_x-coordinate using the Runge-Kutta method, where h=0.01

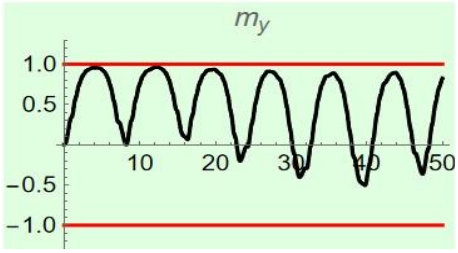


Figure 13. Values of the spatial m_y-coordinate using the Runge-Kutta method, where h=0.01

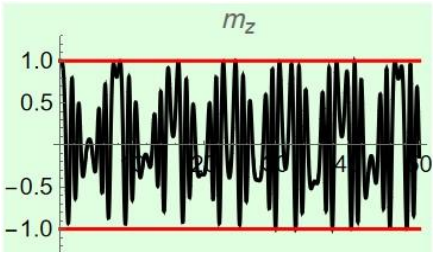


Figure 14. Values of the spatial m_z-coordinate using the Runge-Kutta method, where h=0.01

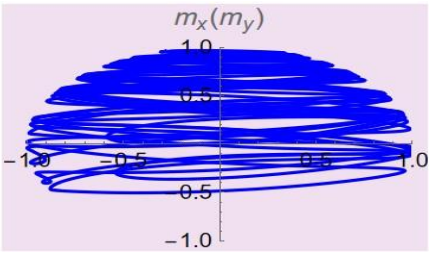


Figure 15. Precession of the magnetic moment in the $m_x(m_y)$ plane

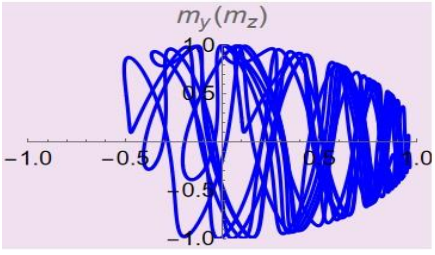


Figure 16. Precession of the magnetic moment in the $m_y(m_z)$ plane

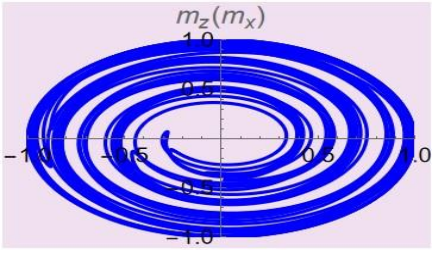


Figure 17. Precession of the magnetic moment in the $m_z(m_x)$ plane

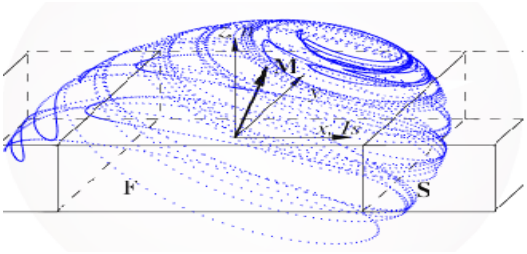


Figure 18. Precession of the magnetic moment in space

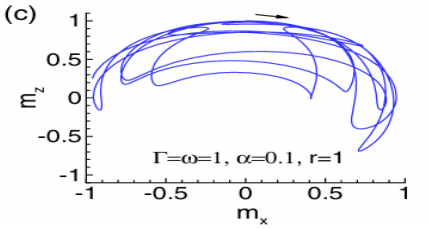


Figure 19. Physical Review Letters 123, 169901(E) (2019)

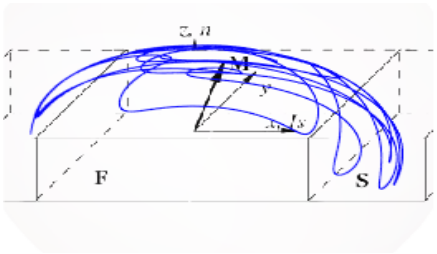


Figure 20. Our results for the identical set of parameters

Verification

This paper has conducted a comparison with other studies and has demonstrated consistent results, further validating the findings. As illustrated in the verification figure provided (Fig. 19 and Fig. 20), the numerical behaviour for the parameters $G = 1, r = 1, \alpha = 0.1, \omega = 1, \omega_F = 1$ observed aligns with those reported in previous research [16].

5. Conclusion

The findings presented in this paper contribute to advancing the understanding of parameter influence on the dynamics of Josephson junction systems. We explored two different sets of parameters to demonstrate the stiffness of the problem under varied conditions. The results underscore the limitations of simpler explicit methods. Conversely, advanced implicit methods achieve high levels of accuracy. By systematically analysing the behaviour of different methods under varying conditions, this research offers valuable insights into the computational challenges associated with modelling complex physical phenomena in Josephson junctions.

Furthermore, this work highlights the importance of numerical strategies in the spintronics field, which plays a critical role in advancing quantum computing. A deeper exploration of these mechanisms is essential for improving the precision and efficiency of models that underlie quantum technologies. Future work may involve refining these methods and exploring more complex junction models with additional physical phenomena.

Future research could involve the implementation of various implicit methods that incorporate adaptive step sizes and stopping criteria to mitigate numerical overflow issues. By optimising the step size dynamically, improvements in computational efficiency and result precision are anticipated. Future investigations may involve magnetic moments with x-coordinate (long JJ) – in such a case we would have to solve a system of partial differential equations [17]. A more general model consider a second harmonic in the current-phase relation - the model includes a double sine-Gordon equation [18].

Acknowledgements

Special thanks are extended to Pavlina Atanasova and Stanka Hadzhikoleva for their support, valuable advice, constructive criticism, and insightful comments.

References

[1] Borzì, A. (2020). *Modelling with ordinary differential equations: a comprehensive approach*. Chapman and Hall/CRC. https://doi.org/10.1201/9781351190398

[2] Bobkov, G. A., Bobkova, I. V., & Bobkov, A. M. (2024). Controllable magnetic states in chains of coupled φ₀-Josephson junctions with ferromagnetic weak links. *Physical Review B*, *109*(5), 054523. https://doi.org/10.1103/PhysRevB.109.054523

[3] Nashaat, M., Kovalenko, E., & Shukrinov, Y. M. (2024). Buzdin, Shapiro, and chimera steps in φ₀-Josephson junctions. II. Bifurcation, switching, and hysteresis. *Physical Review B*, *110*(2), 024510. https://doi.org/10.1103/PhysRevB.110.024510

[4] Buzdin, A. (2008). Direct coupling between magnetism and superconducting current in the Josephson φ₀-junction. *Physical review letters*, *101*(10), 107005. https://doi.org/10.1103/PhysRevLett.101.107005

[5] Atanasova, P. K. H., Boyadjiev, T. L., Shukrinov, Y. U. M., Zemlyanaya, E. V., & Seidel, P. (2010). Influence of Josephson current second harmonic on stability of magnetic flux in long junctions. In Journal of Physics: Conference Series (Vol. 248, No. 1, p. 012044). IOP Publishing. https://doi.org/10.1088/1742-6596/248/1/012044

[6] Atanasova, P. K., Zemlyanaya, E. V., & Shukrinov, Y. M. (2012). Interconnection between static regimes in the LJJs described by the double sine-Gordon equation. In *Journal of Physics: Conference Series* (Vol. 393, No. 1, p. 012023). IOP Publishing. https://doi.org/10.1088/1742-6596/393/1/012023

[7] Shukrinov, Y. M., Rahmonov, I. R., Sengupta, K., & Buzdin, A. (2017). Magnetization reversal by

superconducting current in φ Josephson junctions. *Applied Physics Letters*, *110*(18). https://doi.org/10.1063/1.4983090

[8] Atanasova, P. K., Panayotova, S. A., Zemlyanaya, E. V., Shukrinov, Y. M., & Rahmonov, I. R. (2019). Numerical simulation of the stiff system of equations within the spintronic model. In *Numerical Methods and Applications: 9th International Conference, NMA 2018, Borovets, Bulgaria, August 20-24, 2018, Revised Selected Papers 9* (pp. 301-308). Springer International Publishing. https://doi.org/10.1007/978-3-030-10692-8_33

[9] Atanasova, P., Panayotova, S., Shukrinov, Y., Rahmonov, I., & Zemlyanaya, E. (2018). User Software for Numerical Study of Josephson Junction with Magnetic Momenta. In *EPJ Web of Conferences* (Vol. 173, p. 05002). EDP Sciences. https://doi.org/10.1051/epjconf/201817305002

[10] Wolfram, "Official website of Wolfram Mathematica," Wolfram, Jul. 5, 2024. [Online]. Available: https://www.wolfram.com/mathematica/.

[11] Boyanov, B. & Semerdzhiev, H. (1988) Numerical Methods, Publ. Paisii Hilendarski University of Plovdiv.

[12] Butcher, J. C. (1964). Implicit runge-kutta processes. *Mathematics of computation*, *18*(85), 50-64. https://doi.org/10.2307/2003405

[13] Hairer, E., Nørsett, S. P. & Wanner, G. (1987) *Solving Ordinary Differential Equations I*. Springer-Verlag. https://doi.org/10.1007/978-3-662-12607-3

[14] Atanasova, P. K., Panayotova, S. A., Zemlyanaya, E. V., Shukrinov, Y. M., & Rahmonov, I. R. (2019). Numerical simulation of the stiff system of equations within the spintronic model. In *Numerical Methods and Applications: 9th International Conference, NMA 2018, Borovets, Bulgaria, August 20-24, 2018, Revised Selected Papers 9* (pp. 301-308). Springer International Publishing. https://doi.org/10.1007/978-3-030-10692-8_33

[15] Hall, G. G., & Watt, J. M. (1976). *Modern numerical methods for ordinary differential equations*. Oxford, U.K.: Clarendon Press.

[16] Konschelle, F., & Buzdin, A. (2009). Magnetic moment manipulation by a Josephson current. *Physical Review Letters*, *102*(1), 017001. https://doi.org/10.1103/PhysRevLett.102.017001

[17] Atanasova, P. K., Boyadjiev, T. L., Shukrinov, Y. M., & Zemlyanaya, E. V. (2011). Numerical modeling of long Josephson junctions in the frame of double sine-Gordon equation. *Mathematical Models and Computer Simulations*, *3*, 389-398. https://doi.org/10.1134/S2070048211030033

[18] Atanasova, P. & Georgiev, V. (2023) Numerical solving of the sine-Gordon equation. In *Proceedings of the International Scientific Conference "Informatics, Mathematics, Education and Their Applications"* (IMEA 2023), Pamporovo, Bulgaria, Nov. 29–Dec. 1, 2023, Plovdiv University Press, pp. 23–30, ISBN: 978-619-7663-79-2.

Computer Science and Interdisciplinary Research Journal
Volume 1, Issue 2, 2024. ISSN: 3033-1218, DOI: 10.70862/CSIR.2024.0102-08

Classification of News Using Artificial Intelligence Methods

Georgi Gustinov[1], Emil Hadzhikolev[2], Stanka Hadzhikoleva[3], Stoyan Cheresharov[4]

University of Plovdiv Paisii Hilendarski, 236, Bulgaria Blvd, 4027 Plovdiv, Bulgaria

[1] lz3gt@uni-plovdiv.bg, [2] hadjikolev@uni-plovdiv.bg,
[3] stankah@uni-plovdiv.bg, [4] cheresharov@uni-plovdiv.bg

Abstract – Many organizations use analysis and forecasting systems that rely on human labor for data collection and subsequent processing. The use of artificial intelligence algorithms improves the efficiency of these systems by reducing the risk of suboptimal decisions and erroneous judgments. For such systems to be useful, their costs must not exceed the generated revenue.

This article presents experiments on news classification using intelligent methods as part of a larger system for automated forecasting of critical event. A model and experimental framework for a software system utilizing artificial intelligence methods are proposed. A prototype has been developed to experiment with the Multinomial Naive Bayes and Support Vector Machines algorithms, combined with the vectorizers CountVectorizer, HashingVectorizer, and TfidfVectorizer. The tests were conducted using a parallel computer consisting of four Raspberry Pi 4 nodes. The best results were achieved with the Multinomial Naive Bayes method and the CountVectorizer.

Keywords – Critical Events, Forecasting, Classification, Categorization, Parallel Processing.

DOI: https://doi.org/10.70862/CSIR.2024.0102-08

Corresponding author: Emil Hadzhikolev,
University of Plovdiv Paisii Hilendarski
Email: hadjikolev@uni-plovdiv.bg

Received: 12 January 2025.
Revised: 24 January 2025.
Accepted: 18 February 2025.
Published: 22 February 2025.

© 2024 Georgi Gustinov, Emil Hadzhikolev, Stanka Hadzhikoleva & Stoyan Cheresharov. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The article is published with Open Access at http://csirjournal.com/

1. Introduction

Innovations in manufacturing, healthcare, energy, transportation, commercial services, and governance in recent years have led to a continuous increase in global data generation. According to Statista, the volume of data created, captured, copied, and consumed worldwide is projected to reach 394 zettabytes by 2028, which is approximately four times more than in 2022 [1]. The accumulation of large volumes of historical data on past processes and events enables more detailed study and analysis. Predicting critical events in a given field is becoming increasingly accessible thanks to the capabilities of advanced data analysis methods, including those powered by artificial intelligence.

A critical event represents a significant turning point that often occurs unexpectedly and has the potential to lead to serious negative consequences. Such events can disrupt or even completely unsettle the normal course of routine human or business activities. Numerous authors have studied critical events across various domains.

Sornette investigates the collapse of financial markets by examining the causes behind the formation of financial "bubbles" throughout history [2]. The author draws on studies of other historical precedents, such as the "tulip mania" in the Netherlands, the Great Crash of October 1929, Black Monday in 1987, and others. Longin applies extreme value theory in finance and its methods to explain critical events in capital markets [3]. Critical analyses related to modeling Extremal Events in Financial Phenomena have been conducted by Focardi and Fabozzi in [4].

The issue of incidents in healthcare remains a pressing topic. The Australian Commission on Safety and Quality in Health Care published the Incident Management Guide, where it defines an incident as an “event or circumstance that resulted, or could have resulted, in unintended or unnecessary harm to a patient or consumer; or a complaint, loss or damage”

[5]. Woloshynowych et al. conducted a literature review on the methods used for investigating incidents in healthcare. They developed guidelines for analyzing critical incidents in the hospital sector, mental health, and primary care [6]. Working in the same field, Walshe defines a critical event as an instance where a patient received poor-quality care [7]. Seys et al. expanded research on adverse events in healthcare, focusing on their impact on healthcare workers, who are often considered second victims [8]. Baldisseri et al. examined critical events from the perspective of intensive care units, defining such events as cases involving natural disasters, biological (pandemic) and chemical disasters, radiation, anthropogenic (terrorist and criminal acts) disasters, and others [9].

Kadri and colleagues examine critical events caused by natural disasters or technological reasons. They study their impact on critical infrastructure (industrial and residential complexes, transportation infrastructure, etc.), while also considering subsequent cascading effects [10]. In a similar way, Mittelstädt et al. focus on the cascading effects between multiple infrastructures, such as the power grid, communication networks, and road networks. They develop a comprehensive visual analytics system that integrates separate visualizations of different network infrastructures with social media analysis and on-site mobile analytics [11]. Nivolianitou and Synodinou identify factors that contribute to effective emergency management, including natural disasters and critical incidents. They highlight the importance of prevention, coordination, adherence to procedures, and clarity of roles [12]. Mijalkovic and Cvetković explore and analyze the vulnerability of critical infrastructure during major natural disasters. They present various approaches and measures to minimize damages and ensure resilience during natural disasters [13].

Critical events are also a subject of interest in other domains. For instance, Berzin and De Marco study critical events in human development during the transition to adulthood and their relationship with poverty [14]. Road traffic conflicts as critical events are explored in the work of Bagdadi [15]. He proposes a method for assessing the safety-critical events based on risk evaluation of crashes and the assessment of potential consequences. Various techniques are usually employed for analyzing critical events, including regression, logistic regression, cluster analysis, neural networks, decision trees, and others [16].

Critical events have various forms and can be triggered by different causes, including natural, technological, human, social, and others. Natural factors include earthquakes, floods, hurricanes, volcanoes, and wildfires. Technological factors can involve industrial accidents, transportation incidents, cyberattacks, and more. The human factor encompasses criminal and terrorist activities, human errors, and conflicts. Social factors can include epidemics, population migration, overpopulation, and others.

The management of critical events generally involves four stages:

- *Prevention* – identifying potential risks and critical events and taking measures to reduce the likelihood of their occurrence.
- *Preparation* – developing strategies and action plans, training employees, and ensuring the availability of necessary resources to be used in the event of a critical incident.
- *Response* – providing a rapid and effective reaction to the event to minimize damage.
- *Recovery* – restoring normal conditions after the crisis has ended, analyzing the causes and consequences, and improving current strategies and action plans.

Early identification of critical events and a rapid and effective response are essential for successful crisis management. One possible approach to achieving this is through the analysis of real-time data collected from the internet.

As part of the present study, a module for news classification has been developed and analyzed. Its purpose is to classify news content by topics and is part of a larger software system for early identification of critical events. Independent news from the United States Geological Survey (USGS) [17] is used as input data. Through a classifier, incoming news is distributed into thematic categories. Then, the information must be analyzed for specific terms, based on which the importance and type of the critical event are determined. Machine learning methods are used for classification, which increases processing speed and minimizes human intervention in the system.

The present study is inspired by the publication of Zowalla et al. [18]. Their team developed a crawler for the acquisition of the German Health Web, which includes all health-related web content from Germany, Austria, and Switzerland. Using this crawler, they collected and analyzed health-related articles from public internet sources. To classify the publications, they employed the Support Vector Machine (SVM) algorithm, which is highly suitable for text categorization [19]. Zowalla and colleagues found that over 80% of the target data could be successfully retrieved from the internet [18], which provides a strong basis for its use as a primary data source. The choice of the news section of the USGS is driven by its specific structure, independent of the news flow in established news agencies. Typically, USGS news is

used as a primary source in the media space and is republished or cited.

The aim of the current experiment is:

- To create an automatic classifier for articles from multiple media streams;
- To investigate the classification effectiveness, measured using a statistical method;
- To study the flexibility of the model by modifying their parameters.

2. Models and Methods

2.1. Model of a Critical Events System

Every organization that works with information requires an analytical system for its processing. Systems based on the human factor are inert, and even if they are effective and successful initially, at some point, they lead to suboptimal decisions [20] and bias [21]. The main objective of the proposed automated system with artificial intelligence is to optimize activities and eliminate errors and inaccuracies caused by human intervention. The system consists of the following modules (Fig. 1):

1. Data collection from the internet: The data collection module aims to periodically extract internet content from specified web services according to a predefined schedule. Each extracted result must be saved in a text file.

2. Processing of collected data: This module is responsible for parsing the data, where the text is split into words, and they are stored in appropriate structures for further processing.

3. Data storage: The data is stored on scalable file systems in relational or non-relational databases for subsequent extraction and analysis.

4. Data analysis: Data analysis can be performed using various artificial intelligence methods, and for real-time analysis and classification of information, there are suitable libraries for natural language processing. The goal is to classify the output words according to their belonging to specific linguistic sets, resulting in classified individual words or phrases ranked by importance. Another responsibility of this module is the identification of potential critical events, which is based on expressions with a high significance coefficient or recurrence in the input data.

5. Decision making: This module aims to generate signals for potential critical events, which are to be sent to systems specialized in processing signals of the respective type.

6. Visualization: The results for critical events recorded in the graph database must be visualized for users in an appropriate manner.

After conducting research, the technologies presented in Table 1 were selected as suitable for the initial experiments.

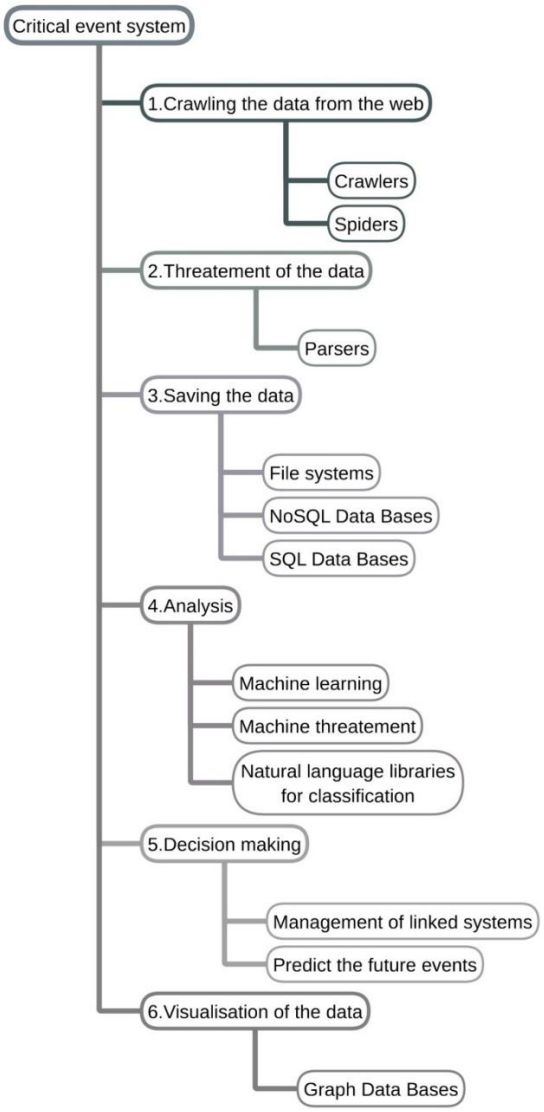


Figure 1. Model of a system for identifying, analyzing, and processing critical events.

Table 1. Modules and suitable technologies for their implementation.

Module	Technologies
Data collection	Crawler – Apache Nutch
Data processing	Parser – Apache Tika
Data storage	File storage – Apache Hadoop
Data analysis	Spark, NLP, NLTK
Decision-making	System Functions
Visualization	Standard Visualization Tools for Neo4j on Hadoop

2.2. Data Collection and Preparation

In the initial experiments, the news from the USGS website (page https://www.usgs.gov/news/news-releases/all-news-releases) was collected manually. The subsequent categorization and filtering of the content were performed automatically. Separating meaningful content from repetitive phrases used for advertisements and links to other websites is crucial

for the categorization of the news. The steps in the text processing workflow are as follows:

1. Open the news page and select the desired category of news from the Topic section.
2. In the Year section, select a specific year and sort the articles in descending order.
3. Download all news articles from the selected year and save them in text format.
4. Edit the saved files by removing all unnecessary links or text located before the Breadcrumb tag, as well as after the tags ### or Get Our News, depending on the content.
5. Store the cleaned files without extensions, assigning them sequential numerical names in the current dataset.
6. Once the addition of news articles to the dataset is complete, compress the dataset in ZIP format.
7. The resulting compressed dataset is used as input for the classifier executed in the research environment Jupyter [22].

The technique used for text content analysis and classification is Bag of Words (BoW). This technique represents a document as a BoW vector, where each element corresponds to the normalized count of occurrences of a given word in the document [23, 24]. The counting process requires exact matching, which is treated as a strict match between the words in the text and the predefined words or expressions. The main disadvantage of this approach is the inability to perform higher-order semantic analysis; consequently, in the current implementation, the system cannot recognize nuances in the analyzed documents. Despite these limitations, this method was chosen due to its speed. Experiments were conducted using three types of vectorizers — CountVectorizer, HashingVectorizer, and TfidfVectorizer - available in the scikit-learn library [25, 26].

For the creation of the initial dataset, news from USGS in the form of HTML documents [17] was used, categorized into Biology, Climate, and Energy. The process of creating a word vector corresponding to a single news item involves several stages (Fig. 2):

1. Initial processing of the text document: At this stage, duplicate titles and links to similar articles are removed from the text files.

2. Applying a regular expression (REGEX): This removes prepositions, punctuation marks, whitespaces, and others.

3. Tokenizing: By applying the regular expression, a list of key words in the text is generated.

4. Removing stop words: Optionally, parasitic connecting words with no weight in the current text (known as stop words) such as if, then, also, are, have, etc., can be removed.

5. Applying a Vectorizer: This step provides a vectorized numerical representation of the word list.

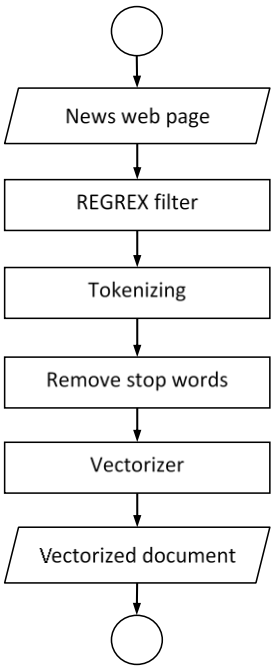


Figure 2. General process of creating a vectorized document.

Different vectorizers use different algorithms to determine the significance of words, which is subsequently important for machine learning classification methods.

2.3. Classification Methods

The classification of vectorized documents is performed using the algorithms Multinomial Naive Bayes [27, 28] and Support Vector Machines [29] with varying kernels. These algorithms operate on word vectors and calculate the percentage ratio of specific expressions in the text relative to all expressions within it. Based on these calculations, the category to which the currently processed document belongs is determined.

The input data for training and testing the models is split into a training and test dataset in a 75:25 ratio.

The training dataset X_{train} consists of document vectors and their corresponding target categories and can be represented as:

$$X_{train} = \{(DocumentVector_i, TargetCategory_i)\}_{i=1}^{n}, n \in N.$$

Similarly, the test dataset X_{test} also contains document vectors and their corresponding real categories, represented as:

$$X_{test} = \{(DocumentVector_j, RealCategory_j)\}_{j=1}^{m}, m \in N.$$

The model M is trained on the training dataset X_{train}. After training, the model is applied to the document vectors in the test dataset to evaluate its performance. During testing, the model predicts a category for each vector in the test dataset:

$$PredictedCategory_j = M(DocumentVector_j),$$
$$j = 1, \dots, m.$$

Various evaluation methods compare the real and predicted categories – $RealCategory_j$ and $PredictedCategory_j$ – for each document vector and its original news content.

The overall classification process includes the parallel training of different classification methods on vectorized documents (Figure 3).

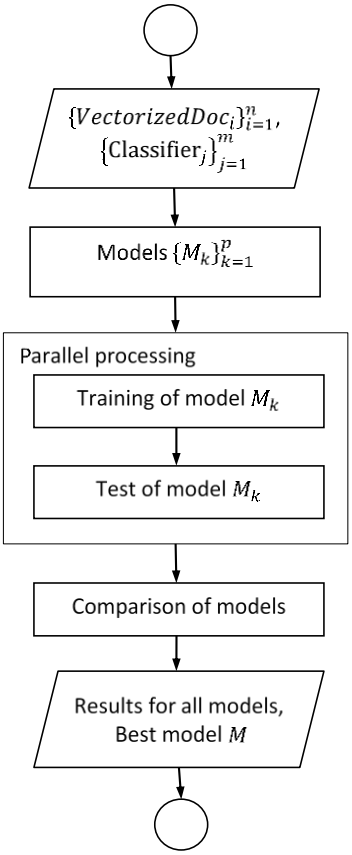


Figure 3. General diagram of the process for training, testing, and evaluating classification models.

In the mathematical model, the total number of vectorized documents is denoted as a set of n elements, and the classifiers are denoted as a set of m elements:

$$VectorizedDocuments = \{VectorizedDocument_i\}_{i=1}^{n}$$
$$Classifiers = \{Classifier_j\}_{j=1}^{m}, n, m \in N.$$

The maximum number of training models, combining classifiers and vectorized documents, is $n \times m$:

$$Models = \{M_k\}_{k=1}^{p} \subseteq \{M_{i,j}\}_{i=1,j=1}^{i=n,j=m}, n, m, p \in N.$$

To identify the most effective model, all models can be trained in parallel, or experiments can be conducted with only p selected models. Each of the selected models is trained and tested sequentially. Important steps in the process include presenting the performance of all models and comparing them to determine the model with the best results.

2.4. Hardware and Software Tools

The research utilized the following tools and systems:

- *Parallel Computer*: A cluster consisting of four Raspberry Pi 4 nodes with quad-core processors and 8 GB of RAM [30], designed to minimize power consumption (approximately 4x2.5 watts compared to 200–400 watts for a standard desktop computer) while providing acceptable performance.
- *Operating system* Ubuntu 22 Server [31].
- *Remote research environment Jupyter* [22].
- *Programming language Python 3* [32].

To accelerate computations when processing large datasets, IPython Interactive Computing [33] was employed. This setup utilizes a parallel cluster of Python 3 instances distributed across each node (four instances, one for each core). While this configuration was tested for performance evaluation, it was not deemed necessary for the specific case. However, the need for further exploration of this parallel system as a cost-effective option for 24/7 parallel text analysis was noted.

The parallel computer was equipped with a HADOOP distributed file system [34] and the SPARK analytics engine [35]. These software systems will be utilized for data storage and analysis when the volume of processed information is expanded.

3. Experiments and Results

The training data includes a total of 177 news articles, collected manually and divided into a ratio of 75% for training and 25% for testing. The news articles include a title, date, content, and a concluding section. The distribution of the news articles by category and group is presented in Table 2.

Table 2. Number of articles by categories and data groups

Category	Total Articles	Training Articles	Test Articles
Biology	62	47	15
Climate	40	30	10
Energy	75	55	20

In the initial experiments, the vectorizers CountVectorizer, HashingVectorizer, and TfidfVectorizer were tested in combination with the algorithms Multinomial Naive Bayes and Support Vector Machines with a linear kernel. To extract meaningful words, the expression REGEX = ur"\b\w\w+\b" was used, which identifies whole words with at least two characters in the text. Based on the obtained results (Table 3), the following combinations were selected as the most suitable for further experiments:

- Multinomial Naive Bayes with CountVectorizer;
- Support Vector Machines with a linear kernel combined with TfidfVectorizer.

A cross-validation coefficient of $K = 6$ was chosen to ensure balanced use of the data for training and testing.

Table 3. Precision using the regular expression REGEX = ur"\b\w\w+\b" for different combinations of algorithms and vectorizers.

Algorithm	Count Vectorizer	Hashing Vectorizer	Tfidf Vectorizer
MNBayes	0.712	0.441	0.621
SVM (Linear Kernel)	0.689	0.593	0.717

In subsequent experiments, the REGEX: ur"\b[a-z0-9_-.]+[a-z][a-z0-9_-.]+\b" was tested, where strings containing special characters such as hyphens (-), dots (.), digits (0-9), etc., are also considered as words. Additionally, frequently repeated parasitic words (stop words) [36] such as “and”, “is”, “are”, “then”, etc., were removed. The results are presented in Table 4.

Based on the previous studies, experiments were conducted with several variants of the classifiers (Table 5). It is observed that the removal of stop words has a positive impact on the precision of the classifiers.

Table 4. Precision of selected algorithms using REGEX = ur"\b[a-z0-9_-.]+[a-z][a-z0-9_-.]+\b" and stop words.

Algorithm	REGEX Only	REGEX + Stop Words
MNBayes + CountVectorizer	0.718	0.701
SVM (Linear Kernel) + TfidfVectorizer	0.706	0.717

Table 5. Precision of selected classifiers.

Classifier	Training Dataset	Test Dataset
CountVectorizer, MNBayes, REGEX, alpha, stop words	0.955	0.778
CountVectorizer, MNBayes	0.955	0.800
TfidfVectorizer, SVM Linear, REGEX, stop words	0.947	0.756
TfidfVectorizer, SVM Rbf, REGEX, stop words	0.947	0.689
TfidfVectorizer, SVM Linear	0.932	0.756

The confusion matrix for Multinomial Naive Bayes + CountVectorizer on the test dataset (Table 6) shows an accuracy of 0.80. The confusion matrix for SVM Linear Kernel + TfidfVectorizer on the test dataset (Table 7) shows an accuracy of 0.756.

Table 6. Confusion matrix for Multinomial Naive Bayes + CountVectorizer on the test dataset.

	Biology	Climate	Energy
Biology	14	0	1
Climate	2	6	2
Energy	0	4	16

Table 7. Confusion matrix for SVM Linear Kernel + TfidfVectorizer on the test dataset.

	Biology	Climate	Energy
Biology	14	0	1
Climate	5	3	2
Energy	1	2	17

In both cases, classification errors in the Climate category can be attributed to the smaller amount of training data available for this category.

The best classifier in the current experiments is the CountVectorizer + Multinomial Naive Bayes model, which achieved an accuracy of 0.95 on the training dataset and 0.80 on the test dataset.

The evaluation metrics – Precision, Recall, and F1-Score – for assessing the quality of the classification model are presented in Table 8.

Table 8. Evaluation of the CountVectorizer + Multinomial Naive Bayes Classifier.

Category	Precision	Recall	F1-Score
Biology	0.88	0.93	0.90
Climate	0.60	0.60	0.60
Energy	0.84	0.80	0.82

4. Conclusion

This study presents a news classification module, which is part of a system for predicting critical events. The hardware and software tools used for its implementation are described. Experiments were conducted to utilize artificial intelligence algorithms for classifying news into different categories. The most effective model, achieving 80% accuracy, was obtained using the combination of the Multinomial Naive Bayes method and the CountVectorizer. Improved model performance can be expected with larger test datasets. The use of Large Language Models for semantic analysis and prediction will be the subject of future experiments.

References

[1] Statista (2024) Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2023, with forecasts from 2024 to 2028, Retrieved from

https://www.statista.com/statistics/871513/worldwide-data-created/

[2] Sornette, D. (2009). *Why stock markets crash: critical events in complex financial systems*. Princeton university press. https://doi.org/10.1515/9781400829552

[3] Longin, F. (Ed.). (2016). *Extreme events in finance: A handbook of extreme value theory and its applications*. John Wiley & Sons. http://dx.doi.org/10.1002/9781118650318

[4] Focardi, S. M., & Fabozzi, F. J. (2003). Fat tails, scaling, and stable laws: a critical look at modeling extremal events in financial phenomena. *The Journal of Risk Finance*, *5*(1), 5-26. https://doi.org/10.1108/eb022976

[5] Australian Commission on Safety and Quality in Health Care. (n.d.). *Incident management guide*. Retrieved from https://www.safetyandquality.gov.au/publications-and-resources/resource-library/incident-management-guide

[6] Woloshynowych, M., Rogers, S., Taylor-Adams, S., & Vincent, C. (2005). The investigation and analysis of critical incidents and adverse events in healthcare. http://dx.doi.org/10.3310/hta9190

[7] Walshe, K. (2000). Adverse events in health care: issues in measurement. *BMJ Quality & Safety*, *9*(1), 47-52. https://doi.org/10.1136/qhc.9.1.47

[8] Seys, D., Scott, S., Wu, A., Van Gerven, E., Vleugels, A., Euwema, M., ... & Vanhaecht, K. (2013). Supporting involved health care professionals (second victims) following an adverse health event: a literature review. *International journal of nursing studies*, *50*(5), 678-687. https://doi.org/10.1016/j.ijnurstu.2012.07.006

[9] Baldisseri, M., Reed, M. J., & Wax, R. (Eds.). (2019). *Intensive Care Unit in Disaster, An Issue of Critical Care Clinics* (Vol. 35, No. 4). Elsevier Health Sciences.

[10] Kadri, F., Birregah, B., & Châtelet, E. (2014). The impact of natural disasters on critical infrastructures: A domino effect-based study. *Journal of Homeland Security and Emergency Management*, *11*(2), 217-241. https://doi.org/10.1515/jhsem-2012-0077

[11] Mittelstädt, S., Wang, X., Eaglin, T., Thom, D., Keim, D., Tolone, W., & Ribarsky, W. (2015). An integrated in-situ approach to impacts from natural disasters on critical infrastructures. In *2015 48th Hawaii International Conference on System Sciences* (pp. 1118-1127). IEEE. https://doi.org/10.1109/HICSS.2015.136

[12] Nivolianitou, Z., & Synodinou, B. (2011). Towards emergency management of natural disasters and critical accidents: The Greek experience. *Journal of environmental management*, *92*(10), 2657-2665. https://doi.org/10.1016/j.jenvman.2011.06.003

[13] Mijalković, S., & Cvetković, V. (2013). Vulnerability of critical infrastructure by natural disasters. *national critical infrastructure protection–regional perspective, University of Belgrade– Faculty of Security Studies, Institute for Corporative Security Studies in Ljubljana, Belgrade*, 91-102.

[14] Berzin, S. C., & De Marco, A. C. (2010). Understanding the impact of poverty on critical events in emerging adulthood. *Youth & Society*, *42*(2), 278-300. https://doi.org/10.1177/0044118X09351909

[15] Bagdadi, O. (2013). Estimation of the severity of safety critical events. *Accident Analysis & Prevention*, *50*, 167-174. https://doi.org/10.1016/j.aap.2012.04.007

[16] Acito, F. (2023). Predictive analytics with KNIME. *Analytics for citizen data scientists. Switzerland: Springer*. https://doi.org/10.1007/978-3-031-45630-5

[17] U.S. Geological Survey. (n.d.). *All news releases*. Retrieved from https://www.usgs.gov/news/news-releases/all-news-releases

[18] Zowalla, R., Wetter, T., & Pfeifer, D. (2020). Crawling the german health web: Exploratory study and graph analysis. *Journal of medical Internet research*, *22*(7), e17853. https://doi.org/10.2196/17853

[19] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning* (pp. 137-142). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/BFb0026683

[20] Alós-Ferrer, C., Hügelschäfer, S., & Li, J. (2016). Inertia and decision making. *Frontiers in psychology*, *7*, 169. https://doi.org/10.3389/fpsyg.2016.00169

[21] Samadi, A. H., Panahi, M., & Raanaei, A. (2024). The roots of cognitive inertia: an introduction to institutional changes. In *Institutional Inertia: Theory and Evidence* (pp. 133-153). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-51175-2_6

[22] Project Jupyter. (n.d.). *Project Jupyter*. Retrieved from https://jupyter.org

[23] Zhao, R., & Mao, K. (2017). Fuzzy bag-of-words model for document representation. *IEEE transactions on fuzzy systems*, *26*(2), 794-804. https://doi.org/10.1109/TFUZZ.2017.2690222

[24] Wu, L., Hoi, S. C., & Yu, N. (2010). Semantics-preserving bag-of-words models and applications. *IEEE Transactions on Image Processing*, *19*(7), 1908-1920. https://doi.org/10.1109/TIP.2010.2045169

[25] Scikit-learn. (n.d.). *Scikit-learn: Machine learning in Python — scikit-learn 1.5.1 documentation*. Retrieved from https://scikit-learn.org/

[26] Scikit-learn. (n.d.). *Sklearn.feature_extraction*. Retrieved from https://scikit-learn.org/stable/api/sklearn.feature_extraction.html

[27] Scikit-learn. (n.d.). *1.9. Naive Bayes*. Retrieved from https://scikit-learn.org/stable/modules/naive_bayes.html

[28] Wickramasinghe, I., & Kalutarage, H. (2021). Naive Bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation. *Soft Computing*, *25*(3), 2277-2293.

[29] Scikit-learn. (n.d.). *1.4. Support Vector Machines*. Retrieved from https://scikit-learn.org/stable/modules/svm.html

[30] Raspberry Pi Ltd. (n.d.). *Raspberry Pi 4 Model B specifications*. Retrieved from https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[31] Ubuntu. (n.d.). *Install Ubuntu on a Raspberry Pi*. Retrieved from https://ubuntu.com/download/raspberry-pi

[32] Python Software Foundation. (n.d.). *Welcome to Python.org*. Retrieved from https://www.python.org/

[33] IPython Development Team. (n.d.). *Using IPython for parallel computing — IPython 3.2.1 documentation*. Retrieved from https://ipython.org/ipython-doc/stable/parallel/index.html

[34] Apache Software Foundation. (n.d.). *Apache Hadoop*. Retrieved from https://hadoop.apache.org/

[35] Apache Software Foundation. (n.d.). *Apache Spark - Unified engine for large-scale data analytics*. Retrieved from https://spark.apache.org/

[36] xldrkp. (n.d.). *English Stopwords*. Retrieved from https://gist.github.com/xldrkp/4a3b1a33f10d37bedbe0068f2b4482e8

Computer Science and Interdisciplinary Research Journal
Volume 1, Issue 2, 2024. ISSN: 3033-1218, DOI: 10.70862/CSIR.2024.0102-09

Opportunities to Achieve Cyber Security Through Blockchain and Smart Contracts

Margarita Terziyska [1], Zhelyazko Terziyski [2],
Valentina Nikolova-Alexieva [3], Katina Valeva [4]

[1, 3, 4] University of Food Technologies, 4002 Plovdiv, Bulgaria
[2] Trakia University, 6000 Stara Zagora, Bulgaria

[1] mterziyska@uft-plovdiv.bg, [2] zheyazko.terziyski@trakia-uni.bg,
[3] v_aleksieva@uft-plovdiv.bg, [4] k_valeva@uft-plovdiv.bg

Abstract – In the context of growing threats in the digital space, cyber security has established itself as a critical component for protecting information systems. With the increasing reliance on digital technology and the Internet, the risks of cyber-attacks and unauthorized access to sensitive data have greatly increased. This highlights the need to develop advanced security mechanisms that ensure the overall safety of systems. Traditional security methods often fail to cope with the complexity of modern threats. Organizations are therefore exploring new technologies, such as blockchain and artificial intelligence, that offer higher levels of security. Blockchain provides innovative solutions that decentralize identity management and reduce dependence on centralized servers. This paper aims to present a methodology combining two blockchain authentication methods – MetaMask Authentication and Smart Contract-Based Authentication, for implementation and analysis, ensuring overcoming challenges in real applications.

Keywords – Cybersecurity, Blockchain, Smart Contracts, Authentication, Decentralized Applications.

DOI: https://doi.org/10.70862/CSIR.2024.0102-09

Corresponding author: Margarita Terziyska, University of Food Technologies
Email: mterziyska@uft-plovdiv.bg

Received: 18 August 2024.
Revised: 24 February 2025.
Accepted: 26 February 2025.
Published: 26 February 2025.

© 2024 Margarita Terziyska, Zhelyazko Terziyski, Valentina Nikolova-Alexieva & Katina Valeva. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The article is published with Open Access at http://csirjournal.com/

1. Introduction

The Internet has undergone a significant transformation in recent decades. What started as static web pages has evolved into a web of intelligent and immersive experiences. From the beginning of the 1990s to today, the evolution of the web has passed through four stages and puts us on the threshold of the fifth - Web 5.0. [1].

Web 1.0, is called the static web. This stage is characterized by the appearance of static sites without interactive features. The information provided in them can only be read by users.

Web 2.0, is called the social network. This web generation has been developing since the beginning of the 21st century. It is characterized by the emergence of social media and the generation of content by users who can interact, collaborate, or share information, photos, links, or videos.

Web 3.0 is a stage of development known as the Semantic Web. This network brings semantic integration and data interoperability. It is characterized by the application of artificial intelligence and machine learning that understand and interpret human language. Technological developments contribute to the ability of websites to provide more personalized experiences for users.

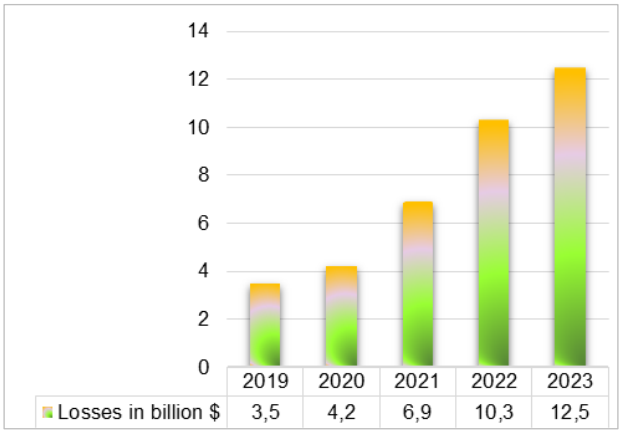
Web 4.0 is associated with a transition from physical to digital technologies. It is in the early stages of development. The aspiration in this stage is through the use of AI, big data, semantic web social networking sites, etc., to develop the Internet to be easier and more convenient to use, more efficient, and personalized for the user.

Web 5.0 is under development. It is seen as a concept that promises to change the way we use and perceive the Internet. At the core of Web 5.0 are modern information technologies such as machine learning, artificial intelligence, and blockchain, which

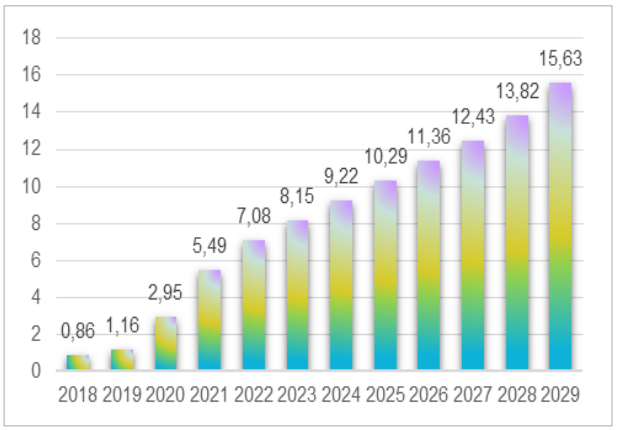
introduce decentralization, increase personal data security, and improve Internet services' sustainability. This is of utmost importance in the era of digitization and global connectivity when cyber security has become critical.

Information that is stored and transmitted in digital form is becoming increasingly valuable. This makes it a prime target for cybercrimes. Data of companies, government agencies, and individual users are constantly under threat from hackers, malware, phishing attacks, and other forms of cyberattacks.

According to the FBI's latest IC3 Internet Crime Report, 880418 complaints were reported in 2023. [2]. The data shows that there has been an annual increase in complaints over the past five years. Despite the reported decline in hacker attack complaints in 2022 compared to the previous year 2021, the results outline a significant increase that is forecast to continue in the future. At the same time, the financial losses of the victims are increasing annually (Fig. 1a), reaching a record value of 12.5 billion dollars in 2023 [3].



(**a**)



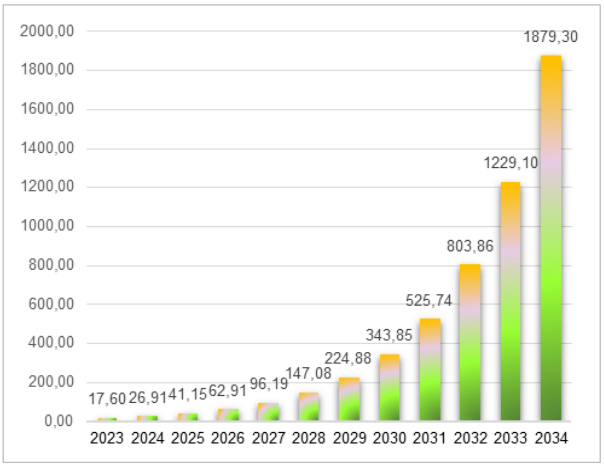
(**b**)

Figure 1. Cybercrime losses and investment: **(a)** Cybercrime losses reported in the last five years; **(b)** Cybercrime defense investment market trend, in trillions of dollars.

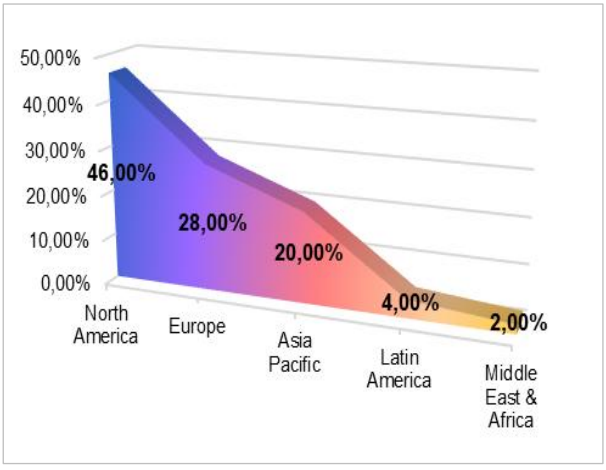
Global cybercrime defence spending is expected to reach $15.63 trillion by 2029 (Fig. 1b) [4]. As digital assets and connectivity increase, cyber threats are also evolving and becoming more sophisticated. Traditional cybersecurity methods, such as antivirus programs and firewalls, are often insufficient to deal with modern attacks. Ransomware attacks, identity theft, DDoS (Distributed Denial of Service) attacks, and attacks against critical infrastructures are just some of the challenges that require new and more effective protection methods [5, 6].

That is why organizations are increasingly turning to new approaches such as blockchain technology and artificial intelligence, which offer increased security and resilience against attacks [4].

Research shows that the blockchain technology market size is projected to grow at a steady steady pace through 2034 (Fig. 2a) In the past year 2023, the global blockchain technology market was valued at USD 17.60 billion. It is expected to reach about US$ 1,879.30 billion at a compound annual growth rate (CAGR) of 52.9% by 2034. [8].



(**a**)



(**b**)

Figure 2. Blockchain technology market and geographic regions distribution: **(a)** Blockchain technology market status and trend, in billion dollars; **(b)** Market share distribution for blockchain technologies by geographic regions, 2023.

The distribution of the blockchain technology market size by geographical regions shows (Fig. 2b) that North America has the largest share for 2023, with 46%, followed [8] by the European market, with 28%. The Asia Pacific region accounts for 20% of the market. The last places are Latin America, with a 4% share, and the Middle East and Africa, with a 2% share.

Blockchain technology, which is often associated with cryptocurrencies, has gained immense popularity recently. It is now widely applied outside of the cryptocurrency context, forming the basis for many applications, including those to improve cybersecurity. Research shows that blockchain can play a key role in protecting data and preventing fraud. A systematic review of the current research on blockchain technology can be found in the [9], which highlights the major challenges and opportunities in the field.

Smart contracts are self-executing programs that are stored and executed on the blockchain. They automate and guarantee the fulfilment of contractual terms without the need for intermediaries. This makes them suitable for building applications in cybersecurity, where they can be used to automate incident response, manage identities, and provide decentralized authentication.

Smart contracts have the potential to transform multiple industries by increasing efficiency and security [10]. This is further confirmed by the extremely diverse applications that are being developed based on blockchain technology. For example, blockchain-based multi-factor authentication provides a higher level of security and integration with various IoT devices [11]. Multi-factor authentication includes additional layers of security by combining traditional passwords with biometrics or other factors, making it more difficult to compromise. The authentication and authorization scheme in mobile cloud environments uses smart contracts to manage access and update user rights while minimizing computational and communication costs [12].

Another application of blockchain technology is systems like Dock.io's Web3 ID, which uses Decentralized Identities (DIDs) and Verifiable Credentials (VCs) to authenticate users without revealing personal data. Such systems improve the user experience by eliminating the need for multiple passwords and accounts. Platforms such as uPort and Sovrin offer stand-alone identity solutions where users securely manage and share their credentials [13]. This concept can be seen as the future of identity management in the digital age.

Blockchain-based solutions often use advanced cryptographic techniques, such as zero-knowledge proofs, which allow users to prove their claims without revealing actual data [10]. This increases the privacy and security of authentication. Crypto wallets such as MetaMask allow users to use their Ethereum accounts for authentication by signing nonce messages with their private keys [14]. This is a secure method of authentication without the need for passwords.

Secure authentication can also be achieved through smart contracts. In his paper [15], Buterin demonstrates how Ethereum smart contracts offer automated authentication and access management solutions using role-based access control (RBAC). This provides better security and reduces the risk of human error.

Blockchain technology finds application in digital identity management in various industries, such as healthcare, education, and logistics. For example, BurstIQ uses blockchain to manage healthcare data, ensuring its integrity and easy transfer between healthcare institutions. This approach ensures a high level of personal data protection and facilitates the exchange of information between the various participants in the healthcare ecosystem.

The use of non-fungible tokens (NFTs) for authentication is a relatively new concept, but it is rapidly gaining popularity due to the advantages it offers in terms of security and control over personal data. NFTs are unique digital assets that use blockchain technology to verify their authenticity and ownership. NFTs are used to verify ownership and access to exclusive digital resources and services [16].

In the literature [17-19], some authors propose different models that use NFTs as unique identifiers for users. A comprehensive review dedicated to NFT can be found in the article [20].

The literature review shows that blockchain offers a range of innovative authentication solutions that can improve the security and efficiency of various systems. By integrating these technologies, organizations can achieve better data protection and more secure identification processes. Among the main challenges are complexity and cost, especially for smaller organizations. This shortcoming applies to all the scientific publications cited above. The authors of this article sincerely believe that the root of these problems is the lack of understanding and, therefore, the limited use of blockchain technology, which is primarily associated with cryptocurrencies. Therefore, the main purpose of this article is to provide basic information about blockchain technology and smart contracts, as well as to demonstrate the step-by-step creation of simple authentication applications called MetaMask Authentication and Smart Contract-Based Authentication.

2. Blockchain Technology and Smart Contracts

Blockchain is a decentralized and distributed digital ledger that securely and transparently records transactions across multiple computers. A detailed overview of the architecture, consensus mechanisms, and future trends in blockchain technology is provided by Zheng et al. [21]. Each transaction is recorded in a block that is linked to the previous block by a cryptographic hash, forming a chain of blocks (blockchain) [22]. The key characteristics of blockchain are decentralization, transparency, and immutability. De-centralization refers to the fact that blockchain operates without a central authority, with data stored and managed by a network of participants (nodes), eliminating single points of failure and increasing security [23]. Each network participant has access to the entire transaction ledger, which provides a high degree of transparency [24]. Once recorded, the data in the blockchain cannot be changed or deleted without compromising the integrity of all subsequent blocks, ensuring their security and reliability [25].

There are several types of blockchain networks – public, private, and consortium.

- Public blockchain is open to anyone who wants to join and participate in the network, like Bitcoin and Ethereum [23];
- A private blockchain is controlled by a single organization that determines who can access and participate in the network and is often used in corporate environments [24];
- Consortium blockchain is a hybrid model where management is performed by a group of organizations, providing partial decentralization and increased security [25].

Blockchain applications are software solutions that use decentralized and secure blockchain technology to manage, store, and exchange data. These applications take advantage of blockchain features, including transparency, immutability, decentralization, and high security, to provide innovative solutions across industries and fields. Currently, blockchain applications have been developed in various fields of human life, such as voting and e-governance (Voatz, Follow My Vote), healthcare (Medicalchain, BurstIQ), food industry in supply chain management (VeChain, IBM Food Trust), digital identity management (Civic, uPort, Sovrin), real estate sales (Propy, RealT), entertainment and media by providing copyright management and direct payments to content creators (Audius, Mycelia), transaction finance with cryptocurrency, international transfers, and asset management (Bitcoin, Ethereum, Ripple), etc.

3. The Process of Creating Blockchain Applications

Regardless of the area in which the blockchain application will be used, its creation goes through the stages shown in Fig. 3.

In the first stage of developing a blockchain application, the main needs and problems that the application must address are identified. This may include specific business needs, user requirements, or technical challenges. Market research is also conducted to determine the competitive landscape and identify existing solutions and their limitations. The target users of the application are identified and how their needs can be met through blockchain technology is assessed. It is essential to define success metrics that will be used to measure the performance of the application after its implementation.

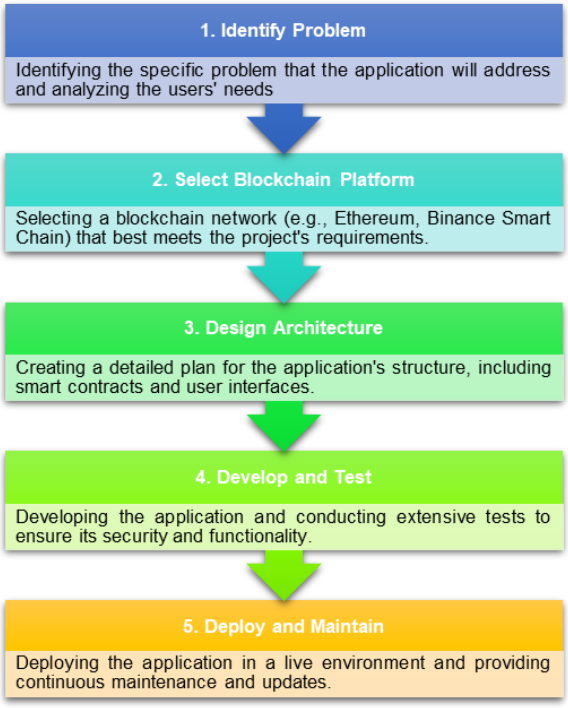


Figure 3. Stages of Blockchain Application Development.

The second stage focuses on choosing an appropriate blockchain platform. Choosing the right blockchain platform is a critical stage that can significantly affect the success of the project. This choice should be based on the specific requirements and goals of the application. Key factors to consider include performance (scalability, transaction speed, and network capacity), security, presence of an active developer community, and compatibility with required tools and protocols. Scalability refers to the platform's ability to handle an increasing volume of transactions. Transaction speed, which is the time it takes to process transactions, is a key factor for user-centric applications. Network capacity is the total

number of transactions the network can process per second (TPS). Security measures of the blockchain platform refer to its protection against vulnerabilities and attacks and the type of consensus algorithms (Proof of Work, Proof of Stake, etc.) that affect the level of security and efficiency. The requirement for an active developer community ensures that there are more resources, tools and libraries, documentation, and support materials available to developers that can make the development process easier and faster. Compatibility is primarily related to the platform's ability to integrate with existing systems and technologies, as well as the required protocols and standards for the application.

In the second stage, a blockchain platform is selected. Some of the most popular blockchain platforms are:

- Ethereum is a public, open platform that allows developers to create and execute smart contracts without the involvement of third parties. It is known for its ability to support smart contracts and decentralized applications (dApps). Ethereum has the largest developer community and broad support, which means more resources and educational materials are available (Ethereum White Paper). The platform offers numerous development tools and libraries that can speed up the development process (Cointelegraph). However, Ethereum requires payments to make transactions and execute smart contracts through fees known as "gas". These fees are paid in ether (ETH), the platform's cryptocurrency. Gas is needed to offset the computing resources used by the network to validate and execute transactions and smart contracts;
- Hyperledger Fabric is a private blockchain project supported by the Linux Foundation designed for enterprise solutions. It offers a modular architecture that allows various components, such as consensus and membership, to be plug-in systems. It is suitable for applications requiring high confidentiality and data control;
- EOS.IO is a blockchain platform that focuses on performance and scalability. It uses a Delegated Proof of Stake (DPoS) consensus algorithm that enables high transaction speeds. This platform can process thousands of transactions per second and is suitable for large decentralized applications;
- Binance Smart Chain (BSC) is a blockchain platform created by the Binance crypto-currency exchange. It is compatible with Ethereum, allowing easy migration of dApps from Ethereum to BSC. It supports smart contracts and decentralized applications and offers lower transaction fees compared to Ethereum;
- Cardano is a blockchain platform developed with a scientific approach and partnerships with academic institutions, focusing on research and evidence. It uses the Ouroboros consensus algorithm, which is a proof-of-stake (PoS) system. The platform was built with an emphasis on security and scalability
- Polkadot is a blockchain platform that allows different blockchain chains to interact and exchange information with each other. It uses a system of parallel chains called parachains. It supports multiple parachains, which improves performance and scalability.

When developing blockchain applications (e.g. on the Ethereum platform), costs can be reduced by using testnets. They allow free development and testing of smart contracts using test ether that has no real value. Test ether can be obtained from so-called faucets that distribute free test coins. Rinkeby, Ropsten, Kovan, and Goerli are some of the better-known test networks.

It is also possible to use local blockchain or layer 2 solutions, which are important components for developing and optimizing blockchain applications. They provide opportunities to improve performance, reduce costs, and test applications in a more controlled environment. Some popular native blockchains include Ganache, which allows developers to simulate a blockchain network on their computer and is suitable for testing and developing Ethereum smart contracts; Hardhat, which is an Ethereum development environment that includes a built-in local blockchain for testing and is suitable for testing complex smart contracts and integrations in decentralized applications; Hyperledger Fabric Local Network, which is a local blockchain network for enterprise solutions; and Parity Substrate, a modular framework for building custom blockchains. Among the better-known Tier 2 solutions are Polygon, which provides scalable and low-cost transactions for Ethereum and is suitable for applications requiring high transaction throughputs, such as games and decentralized finance (DeFi) platforms; Arbitrum, which is used in DeFi platforms and applications requiring fast data processing, as it allows the use of optimistic aggregations to improve scalability; and Optimism, which is suitable for applications that want to reduce transaction fees and improve the user experience.

The third stage of developing a blockchain application is dedicated to designing its architecture. This includes defining the structure of the application, including the front and back ends, the user interface design, and determining how the blockchain

components will interact with other parts of the system. It is also necessary to define smart contracts, i.e., to develop their logic and determine what functions they will perform and how they will interact with the blockchain.

During the fourth stage, application components are created and integrated using appropriate technologies and programming languages. Functional testing is conducted followed by stress testing to ensure that the system can handle workloads, performance analysis, and user testing. The most commonly used programming languages for the development of blockchain applications are listed in Table 1.

Table 1. The most commonly used programming languages for the development of blockchain applications.

Programming language	Description
Solidity	Used for programming when creating blockchain applications and writing smart contracts in Ethereum. It has a JavaScript-like syntax and is specifically de-signed to run on the Ethereum Virtual Machine (EVM). It supports inheritance, libraries, and user-defined complex types, and has a large developer community
Vyper	Alternative to Solidity. It has a stricter and simpler syntax compared to Solidity, which makes it more secure and reduces the risk of vulnerabilities. It also has fewer features but is easier to audit
Rust	Systems programming language known for its speed and security. It is widely used to develop smart contracts on platforms such as Solana and Polkadot
Golang (Go)	Golang is a powerful and efficient language used to write smart contracts and modules in Hyperledger Fabric
JavaScript/ TypeScript	Widely used to create a front-end for decentralized applications. Their main advantages include a huge ecosystem of libraries and tools for building user interfaces, as well as easy integration with blockchain through libraries such as Web3.js and Ethers.js
Python	Multipurpose language used for automation, data analysis and blockchain prototyping. It is easy to learn and use, with powerful support for blockchain integration libraries. Its main applications are related to blockchain application prototyping, blockchain data analysis and automation

When developing blockchain applications, some of the most commonly used tools and libraries are Truffle Suite, Hardhat, Web3.js, Ethers.js, and Brownie (Table 2).

Table 2. The most commonly used tools and libraries for the development of blockchain applications.

Tool/ library	Description
Truffle Suite	Complete framework for developing Ethereum applications, including tools for compiling, deploying, and testing smart contracts. Its main components are the Ganache platform and Drizzle - a state management library for dApps
Hardhat	Hardhat, as mentioned earlier, is a professional Ethereum development environment offering powerful tools for debugging and testing smart contracts
Web3.js	JavaScript library that enables interaction with the Ethereum blockchain from front-end applications. It is used to perform transactions, read data from the blockchain, and interact with smart contracts, allowing easy integration with dApps
Ethers.js	Lightweight library for interacting with Ethereum, known for its simplicity and powerful features. It provides tools for creating wallets, managing transactions, and interacting with smart contracts
Brownie	Python-based framework for developing and testing Ethereum smart contracts. Supports automated testing and deployment management. It is used to develop and test Ethereum smart contracts using Python

In the final fifth stage, the completed blockchain application is deployed on the chosen blockchain platform with the necessary configurations. Application performance monitoring is done to promptly detect and resolve any issues. It is recommended to offer updates and improvements to the blockchain application, as well as provide support to users in case of questions or problems.

4. Blockchain Authentication

4.1. Crypto wallet-based authentication

Crypto wallets are digital tools that allow users to store, manage, and transact with cryptocurrencies. They provide an interface for accessing blockchain networks while ensuring the security of private keys necessary for signing transactions. Crypto wallets can be categorized into two main types: "hot wallets" and "cold wallets," depending on whether they are connected to the internet.

Hot wallets are connected to the internet, offering easier access and convenience for daily transactions. Examples of such wallets include MetaMask, Trust Wallet, and Coin-base Wallet. These wallets are popular due to their ease of use and integration with decentralized applications, particularly within the Ethereum ecosystem.

Cold wallets, on the other hand, are not connected to the internet and offer a higher level of security,

reducing the risk of hacking attacks. They are preferred for storing larger amounts of cryptocurrency over the long term. Popular examples of cold wallets include Ledger Nano S and Trezor. These hardware devices store private keys offline, providing an additional layer of protection against cyber-attacks.

The use of crypto wallets is essential for securing cryptocurrencies and managing digital assets. According to research by Satoshi Nakamoto, the creator of Bitcoin, crypto wallets play a crucial role in decentralized security by providing control over private keys and eliminating the need for trust in third parties [22].

For the created example, the MetaMask crypto wallet was selected due to its wide-spread use and ease of integration with dApps.

MetaMaks is a tool for interacting with decentralized applications on blockchain networks such as Ethereum. It provides a simple interface for managing Ethereum ac-counts, sending and receiving cryptocurrencies, and signing transactions and messages directly from the browser. MetaMask operates as a browser extension, allowing users to interact with the blockchain without the need to install full nodes or specialized software. A full node refers to a computer that fully validates transactions and blocks in the block-chain. Currently, MetaMask is compatible with some of the most popular browsers, including Google Chrome, Mozilla Firefox, Brave, Microsoft Edge, and others. It should be noted that automatic installation of MetaMask as a browser extension is not possible due to security and privacy concerns. Browsers do not allow websites or scripts to install ex-tensions automatically without the user's consent.

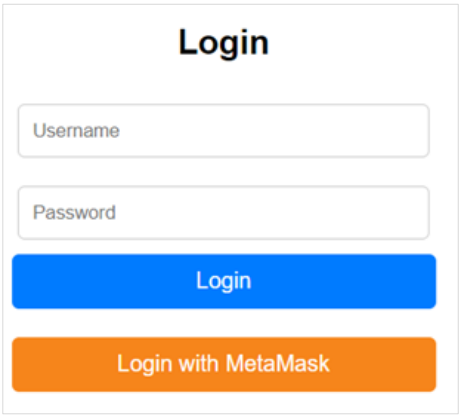
Once MetaMask is installed, it needs to be configured. This involves creating a new account or importing an existing one. Creating a new account requires setting a strong password, as well as a 12-word secret phrase that serves as a backup for account recovery. In case of loss of access to MetaMask or your computer, the wallet can be restored using this phrase. After the secret phrase is confirmed, the setup process is complete, and the wallet becomes ready for use. This includes receiving and sending cryptocurrencies, signing transactions, and interacting with decentralized applications.

To demonstrate the integration process, a standard login form was created, similar to those used by most websites or web-based information systems. This example form is depicted in Fig. 4a.

The HTML code behind the login form is shown in Fig. 4b. The JavaScript function checks if MetaMask is installed and requests permission to access the user's MetaMask account, which can then be used to log in to the system.

Clicking on the "Login" and "Login with MetaMask" buttons initiates two distinct processes, as they use different methods for user authentication.

When clicking the "Login" button, the user enters their username and password into the corresponding fields and submits the form to the server for processing. Basic client-side validations, such as checking for empty fields or minimum password length, may be performed. Once the form passes all client-side validations, the data (username and password) is sent to the server via an HTTP POST request.



(**a**)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.
    <title>MetaMask Authentication - Primer 1</title>
</head>
<body>
    <div class="container">
        <h2>Login</h2>
        <form>
            <input type="text" placeholder="Username" required><br>
            <input type="password" placeholder="Password" required><br>
            <input type="submit" value="Login">
        </form>
        <div class="metamask-btn" onclick="connectMetaMask()">Login wit
MetaMask</div>
    </div>

    <script>
        function connectMetaMask() {
            if (typeof window.ethereum !== 'undefined') {
                console.log('MetaMask is installed!');
                window.ethereum.request({ method: 'eth_requestAccounts'
                .then(accounts => {
                    console.log('Connected account:', accounts[0]);
                    alert('Connected account: ' + accounts[0]);
                })
                .catch(error => {
                    console.error('Error connecting to MetaMask:', erro
                    alert('Error connecting to MetaMask');
                });
            } else {
                alert('MetaMask is not installed. Please install MetaMa
and try again.');
            }
        }
    </script>
</body>
</html>
```

(**b**)

Figure 4. Standard login form and basic HTML code for a MetaMask integration and authentication options: **(a)** Standard login form with MetaMask integration; **(b)** Basic HTML code for a login form with standard and MetaMask authentication options.

The server first checks whether the username exists in the database. If the username exists, the server then verifies whether the entered password matches the hashed pass-word stored in the database. If both the username and password are valid, the server creates a session for the user and redirects them to the secure area of the website.

The user is provided with a session identifier (session ID), which is used for identification in subsequent requests to the server. If the username and password are correct, the user will be able to log into the system and gain access to protected resources. If an error occurs (e.g., incorrect password or non-existent username), the user will see an error message.

When clicking the "Login with MetaMask" button, a JavaScript function is triggered, which checks whether MetaMask is installed and available in the browser. The JavaScript code verifies whether window. ethereum is defined in the browser, indicating the presence of MetaMask. If MetaMask is not available, the user is prompted to install MetaMask. If MetaMask is available, the browser sends a request for account access using Ethereum. request({ method: 'eth_requestAccounts' }). MetaMask then displays a pop-up window where the user must confirm or deny access to their Ethereum account. If the user grants access, MetaMask returns a list of Ethereum account addresses associated with the application. Typically, the first account in the list is the user's primary address, which will be used for authentication. The application may send a message to the user for signing with their private key stored in MetaMask. This signature serves as proof of ownership of the account. The signed message is sent back to the server, where it is verified whether the signature corresponds to the account address (this step of the process is optional).

To validate the signature generated by the user via MetaMask, the server must utilize appropriate cryptographic libraries that allow verification of the signature's authenticity against the sent message and the user's public key. The validation process involves sever-al key steps.

First, the server receives the signature, the user's address, and the original message that was signed.

Second, then, using a cryptographic library such as Ethereum-util in Node.js, the server extracts the public key from the signature. With this key, the server can verify whether the signature was created by the private key associated with the address provided by the user. If the signature is valid, the server can confidently assume that the user who presented the signature indeed owns the corresponding Ethereum account.

Third, the server then creates a session for the user and redirects them to the secure area of the website. If the authentication process is successful, the user will be logged into the system without the need to enter a username and password, providing a more secure method of authentication. If there is an error during the connection process or if the user denies access to MetaMask, the process is terminated, and the user remains unauthenticated.

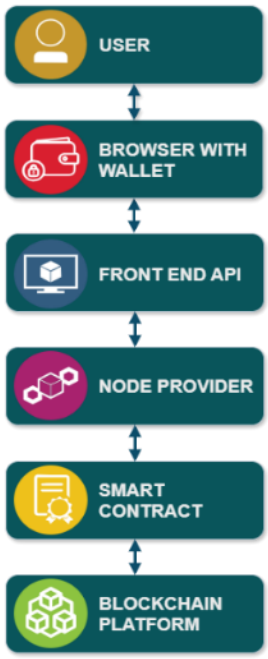
During the authentication process via MetaMask, various errors may occur, which must be properly handled to ensure the system's reliability and a good user experience. One of the most common errors is the user's refusal to grant access to their cryptographic key or to sign the message. In this case, the browser or the MetaMask extension itself generates an exception that must be caught by the application. Upon catching such an exception, the server or the client-side application may provide an appropriate error message to the user, explaining why the authentication cannot proceed. For instance, a message such as "Authentication was denied. Please try again and provide the necessary permissions." may inform the user of the actions they need to take. Additionally, the system should be prepared to handle other potential errors, such as network issues, browser incompatibility, or failures in sending data to the server, with all these situations being addressed through appropriate recovery mechanisms or error messages.

4.2. Smart Contract-Based Authentication

Reliable authentication can also be implemented using smart contracts. These are automated, self-executing programs that exist within a blockchain network and are triggered when predefined conditions are met. They are designed to facilitate, execute, and ensure trust in contractual agreements without the need for intermediaries. Authentication based on smart contracts provides a reliable and secure method for managing user identities and access control in a decentralized environment. These contracts contain the logic necessary for authenticating entities, ensuring that only authorized users have access to specific resources.

For this paper, a Smart Contract-Based Authentication application was developed, and its working diagram is presented in Fig. 5a. The authentication process is initiated by the user, who connects to a web application containing a login form. This application may be part of a decentralized application or an information system using blockchain technology for managing user identities. Simultaneously, the user must have a crypto wallet. For this specific application, MetaMask, a popular browser extension, was used. When attempting to log in, the web application establishes a connection with the crypto wallet through a library such as Web3.js or Ethers.js (in this case, Web3.js was used). The crypto wallet manages the user's cryptographic keys and

provides secure access to their crypto accounts. The user enters their password to unlock access to their account.



(**a**)

```
CONTRACT Authentication:

// Declare a variable to store registered users
DECLARE mapping(address => bool) registeredUsers

// Function to register a user
FUNCTION registration():
    IF registeredUsers[CALLER_ADDRESS] == false THEN:
        // Mark the current address as registered
        registeredUsers[CALLER_ADDRESS] = true
        RETURN "Registration successful"
    ELSE:
        RETURN "User already registered"

// Function to check user authentication
FUNCTION isAuthenticated(userAddress):
    // Returns whether the given address is registered
    RETURN registeredUsers[userAddress]

// Constructor (if necessary)
FUNCTION constructor():
    // Any necessary initial settings for the contract
    INITIALIZE any necessary variables or states

END CONTRACT
```

(**b**)

Figure 5. Smart Contract for Decentralized Applications working diagram and pseudocode: **(a)** Workflow of Smart Contract-Based Authentication; **(b)** Pseudocode of the Authentication Smart Contract for Decentralized Applications.

During the authentication process via MetaMask, various errors may occur, which must be properly handled to ensure the system's reliability and a good user experience. One of the most common errors is the user's refusal to grant access to their cryptographic key or to sign the message. In this case, the browser or the MetaMask extension itself generates an exception that must be caught by the application. Upon catching such an exception, the server or the client-side application may provide an appropriate error message to the user, explaining why the authentication cannot proceed. For instance, a message such as "Authentication was denied. Please try again and provide the necessary permissions." may inform the user of the actions they need to take. Additionally, the system should be prepared to handle other potential errors, such as network issues, browser incompatibility, or failures in sending data to the server, with all these situations being addressed through appropriate recovery mechanisms or error messages.

Once the user is authenticated in the crypto wallet, the Web3.js library acts as an in-termediary between the web application and the blockchain network. It sends a request to the crypto wallet containing the necessary information for authentication. The crypto wallet receives the request and presents the user with details about the transaction or message that needs to be signed. The user can confirm or reject the signing request. If the user confirms, the crypto wallet uses the user's private key to sign the data, which serves as proof of their identity and authorization to perform the requested action.

The signed data is returned to the library, which then forwards it to the node provider. The node provider (for example Infura, Alchemy, QuickNode, etc.) provides access to the blockchain network and sends the signed transaction to the relevant blockchain node for processing. When the signed transaction reaches the blockchain network, it is processed and passed to the smart contract deployed on the blockchain. The smart contract verifies the signature to ensure its validity and that it was executed by an authorized user. If the signature is valid, the smart contract executes the prescribed actions, such as granting access to protected resources or completing the transaction.

After the smart contract completes the requested action, the result is returned along the same path—from the blockchain network, through the node provider, to the Web3.js or Ethers.js library, and finally to the web application in the browser. The web application informs the user of successful authentication and grants access to the corresponding resources or services. This process demonstrates the interaction between the user, the crypto wallet, the blockchain interaction library, the node provider, and the smart contract, all of which play a key role in ensuring the security and efficiency of the system.

One of the critical stages is the development and deployment of the smart contract—the most crucial part of this type of cybersecurity application. The contracts are programmed, most commonly in Solidity. For this example, a smart contract in Solidity was also developed, and its pseudocode is presented

below (Fig. 5b). The smart contract for authentication functions as a basic system for registering and authenticating users within a blockchain environment. Its primary task is to provide a secure method for access management using decentralized infrastructure.

When executed, users have the option to register their Ethereum addresses directly on the blockchain via the registration() function. This function checks whether a given address is already registered. If the address is not found in the registry, it is added, and the user receives a confirmation of successful registration. In cases where the address already exists in the system, the function informs the user that they are already registered.

In addition to registration, the smart contract offers an isAuthenticated() function for verifying user authenticity. This function allows the application to check whether a given address is registered, returning a result that verifies the user's status. This way, the system can determine whether a user has the right to access certain resources or services within the decentralized application.

The smart contract may also contain a constructor() function, which is optional. It is executed only once—at the moment the contract is created—and cannot be called again thereafter. It is most commonly used for initializing variables but can also be used to set the address of the owner or administrator of the contract, define certain trusted addresses such as those of counterparties or other smart contracts with which the current contract will interact, automatically distribute a certain number of tokens to different addresses upon contract deployment, and more. If there are no specific tasks that need to be performed during contract deployment, the constructor() function can be omitted.

Lastly, while some node providers offer paid services, Ganache is used as a development tool in this study. Ganache allows developers to create a local blockchain network for developing and testing smart contracts and decentralized applications.

4.3. Comparative Analysis of Authentication Security

The comparison of authentication security between crypto wallets, particularly MetaMask, and smart contracts reveals different aspects of protection that are crucial for the reliability of blockchain-based systems. MetaMask, as a crypto wallet and browser extension, provides users with a convenient and secure way to manage their cryptographic keys and sign transactions directly from the browser. A key aspect of MetaMask's security is the local storage of keys on the user's device, which reduces the risk of centralized attacks. The user's involvement in approving each transaction adds a layer of protection, as every action requires explicit user consent. However, the security of Meta-Mask is directly linked to the security of the device on which it is installed, which may be vulnerable to physical or software attacks.

On the other hand, smart contracts offer a significantly different approach to authentication security. Smart contracts are automated, self-executing programs that operate within blockchain networks and provide decentralized management of identities and access. One of the primary advantages of smart contracts is their immutability after deployment, ensuring that once the rules and logic are recorded, they cannot be altered or manipulated. The cryptographic security of smart contracts provides reliable protection against data forgery, using proven algorithms for verifying digital signatures and identities. However, smart contracts are not immune to vulnerabilities; they can be exposed to specific risks such as re-entrancy attacks and logical errors, which can be exploited if the contract is not properly tested and audited.

In terms of authentication reliability, MetaMask offers a high level of security based on the local storage of keys and the need for user participation. This makes it a suitable solution for scenarios where user flexibility and control are necessary. However, Meta-Mask relies on the security of the device, which can be a vulnerability in certain contexts. Smart contracts, on the other hand, provide strong protection through decentralization and automation, making them highly suitable for systems requiring immutability and minimization of human involvement in the authentication process. Nevertheless, the security of smart contracts is directly tied to the quality of their code and development practices. Errors in the code can lead to serious vulnerabilities, requiring careful testing and auditing before deployment.

It can be said that the MetaMask wallet is more suitable for situations requiring active user involvement and flexibility, while smart contracts offer greater security and automation in decentralized applications. It is important to note that the security of these technologies depends not only on their built-in mechanisms but also on their proper implementation and maintenance in a real-world environment.

The security of smart contracts is critical to the reliability of blockchain-based authentication systems. Although smart contracts offer powerful possibilities for automation and decentralized management, they are also exposed to specific vulnerabilities that can be exploited if not properly addressed. One of the most well-known vulnerabilities is the re-entrancy attack, in which a malicious contract can re-invoke the original contract's function before the initial transaction is completed. This can allow the attacker to bypass identity verification logic,

potentially leading to unauthorized access to resources or the execution of unauthorized actions. For example, if a contract includes an authentication function that checks whether a user is authorized and then performs an action such as granting access or completing a transaction, a re-entrancy attack could allow multiple invocations of this function before the initial operation is completed. This can disrupt the authentication process and compromise system security.

To protect smart contracts from such attacks, it is essential to follow best practices in development and deployment. An effective strategy is to ensure that the contract's state is updated before making external calls or sending funds. This means that the authentication check should be completed before performing any action that could trigger an external call. Using modifiers like nonReentrant can also help prevent re-invocation of the function until the current transaction is fully completed. This ensures that even if an attempt is made to re-invoke, the contract will not execute the function again and bypass security checks.

Limiting the authentication logic to a single transaction, without relying on interactions with other contracts, can also reduce the risk of re-entrancy attacks.

If possible, the authentication logic should be kept simple, avoiding any interaction with external contracts that could be exploited.

Re-entrancy attacks pose a threat to the security of smart contracts, but with proper design and the application of best practices, their risk can be significantly reduced. In the context of authentication, protection against these attacks is crucial to ensuring the safety and reliability of decentralized systems.

5. Conclusion

This paper examined the security of authentication in the context of blockchain technologies, comparing crypto wallets like MetaMask with smart contracts. The study demonstrated that both approaches offer significant advantages in terms of decentralization, cryptographic protection, and reducing reliance on centralized intermediaries. However, each has specific challenges and vulnerabilities that must be carefully considered when selecting the appropriate technology for a given application.

One of the main conclusions from the analysis is that the choice between MetaMask and smart contracts depends on the context and specific requirements of the system. MetaMask provides a high degree of flexibility and user control, which is particularly useful in cases where user involvement is crucial. On the other hand, smart contracts offer strong protection through immutability and automation, making them highly suitable for systems requiring reliable and decentralized authentication.

Future research in this area should focus on a deeper analysis of vulnerabilities associated with the use of smart contracts, as well as the development of new techniques to mitigate these risks. Specifically, attention should be given to the integration of multi-factor authentication and the application of artificial intelligence to enhance the security and adaptability of these systems. Additionally, exploring the possibilities for standardizing smart contracts and cryptographic protocols used for authentication will be important to facilitate their implementation in real-world applications.

In conclusion, the findings highlight the importance of blockchain technologies in the advancement of modern cybersecurity. They offer new opportunities for creating decentralized, secure, and reliable systems for identity and access management. However, the successful implementation of these technologies requires a thorough understanding of their advantages and limitations, as well as the application of best practices in development, testing, and security auditing. While this study represents only an initial step in the process of integrating blockchain into the field of cybersecurity, the results demonstrate the potential of this approach to significantly contribute to the development of secure and reliable identity and access management systems.

References

[1] Patel, K. (2013). Incremental journey for World Wide Web: Introduced with Web 1.0 to recent Web 5.0–A survey paper. *International Journal of Advanced Research in Computer Science and Software Engineering, 3*(10).

[2] FBI. (2024, December 26). FBI's IC3 report: Losses from cybercrime surpass $12.5 billion—a new record. Retrieved from https://www.proofpoint.com/us/blog/email-and-cloud-threats/fbis-ic3-report-losses-cybercrime-surpass-125-billion-new-record

[3] FBI. (2023). *FBI IC3 Internet crime report 2023*. Retrieved from https://www.ic3.gov/AnnualReport/Reports/2023_ic3report.pdf

[4] CNBC. (2024, September 7). *Estimated cost of cybercrime worldwide 2018-2029*. Retrieved from https://blog.biocomm.ai/2024/09/07/cnbc-estimated-cost-of-cybercrime-worldwide-2018-2029-in-trillion-u-s-dollars/

[5] eSentire. (2024, December 26). *The state of cybersecurity threats*. Retrieved from https://www.esentire.com/

[6] Conti, M., Dehghantanha, A., Franke, K., & Watson, S. (2018). Internet of Things security and forensics: Challenges and opportunities. *Future Generation*

Computer Systems, 78, 544-546. https://doi.org/10.1016/j.future.2017.07.060

[7] Crosby, M., Pattanayak, P., Verma, S., & Kalyanaraman, V. (2016). Blockchain technology: Beyond Bitcoin. *Applied Innovation, 2*(6-10), 71.

[8] Precedence Research. (2024). *Blockchain technology market size, share and trends 2024 to 2034*. Retrieved from https://www.precedenceresearch.com/blockchain-technology-market

[9] Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on blockchain technology? A systematic review. *PLOS One, 11*(10), e0163477. https://doi.org/10.1371/journal.pone.0163477

[10] Szabo, N. (1997, September 1). Formalizing and securing relationships on public networks. *First Monday, 2*(9). https://doi.org/10.5210/fm.v2i9.548

[11] Mahmood, S., Chadhar, M., & Firmin, S. (2022). Cybersecurity challenges in blockchain technology: A scoping review. *Human Behavior and Emerging Technologies, 2022*(1), 7384000. https://doi.org/10.1155/2022/7384000

[12] Villarreal, E. R. D., García-Alonso, J., Moguel, E., & Alegría, J. A. H. (2023). Blockchain for healthcare management systems: A survey on interoperability and security. *IEEE Access, 11*, 5629-5652. https://doi.org/10.1109/ACCESS.2023.3236505

[13] Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the Internet of Things. *IEEE Access, 4*, 2292-2303. https://doi.org/10.1109/ACCESS.2016.2566339

[14] Almadani, M. S., Alotaibi, S., Alsobhi, H., Hussain, O. K., & Hussain, F. K. (2023). Blockchain-based multi-factor authentication: A systematic literature review. *Internet of Things, 100844*. https://doi.org/10.1016/j.iot.2023.100844

[15] Gimenez-Aguilar, M., De Fuentes, J. M., Gonzalez-Manzano, L., & Arroyo, D. (2021). Achieving cybersecurity in blockchain-based systems: A survey. *Future Generation Computer Systems, 124*, 91-118. https://doi.org/10.1016/j.future.2021.05.007

[16] Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *White Paper, 3*(37), 2-1.

[17] Ekblaw, A., Azaria, A., Halamka, J. D., & Lippman, A. (2016, August). A case study for blockchain in healthcare: "MedRec" prototype for electronic health records and medical research data. In *Proceedings of IEEE Open & Big Data Conference* (Vol. 13, p. 13).

[18] Guo, Y., & Liang, C. (2016). Blockchain application and outlook in the banking industry. *Financial Innovation, 2*, 1-12. https://doi.org/10.1186/s40854-016-0034-9

[19] Huh, S., Cho, S., & Kim, S. (2017, February). Managing IoT devices using blockchain platform. In *2017 19th International Conference on Advanced Communication Technology (ICACT)* (pp. 464-467). IEEE. https://doi.org/10.23919/ICACT.2017.7890132

[20] MetaMask. (2024, December 26). *MetaMask developer documentation*. Retrieved from https://docs.metamask.io/

[21] Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE International Congress on Big Data (BigData Congress)* (pp. 557-564). IEEE. https://doi.org/10.1109/BigDataCongress.2017.85

[22] Nakamoto, S. (2008). A peer-to-peer electronic cash system. *Bitcoin*. Retrieved from https://bitcoin.org/bitcoin.pdf

[23] Swan, M. (2015). *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc.

[24] Comert, O. (2020). *Blockchain revolution: How the technology behind Bitcoin and other cryptocurrencies is changing the world*.

[25] Antonopoulos, A. M. (2014). *Mastering Bitcoin: Unlocking digital cryptocurrencies*. O'Reilly Media, Inc.


https://csirjournal.com