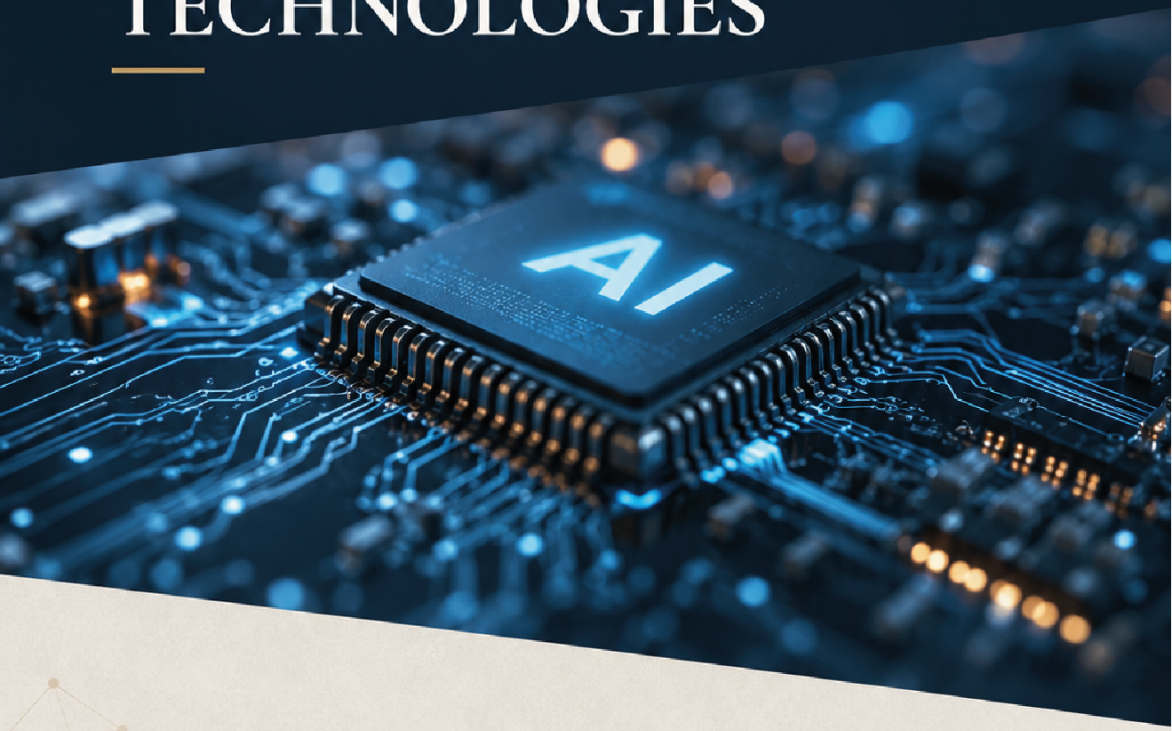
ARTIFICIAL INTELLIGENCE, DIGITAL SYSTEMS AND APPLIED COMPUTING TECHNOLOGIES



EDITOR:

EYYÜP GÜLBANDILAR

BİDGE

BİDGE Yayınları

Artificial Intelligence, Digital Systems and Applied Computing Technologies

Editor: EYYÜP GÜLBANDILAR

ISBN: -

1st Edition
Page Layout By: Gözde YÜCEL
Publication Date: 2026-06-25
BİDGE Yayınları

Certificate No: 71374

www.bidgeyayinlari.com.tr - bidgeyayinlari@gmail.com
Krc Bilişim Ticaret ve Organizasyon Ltd. Şti.
Güzeltepe Mahallesi Abidin Daver Sokak Sefer Apartmanı No: 7/9 Çankaya / Ankara

CONTENTS

CHAPTER 0

APPROACHES BASED ON THE INTERNET OF THINGS AND ARTIFICIAL INTELLIGENCE IN SMART CITIES

AYKUT KARAKAYA[1]

Introduction

In recent years, cities have become much more than physical spaces where people live, they have evolved into dynamic systems that generate data, process it, and increasingly support decision-making through that data. Issues such as population growth, traffic congestion, energy consumption, environmental challenges, security needs, and the demand for more efficient public services have pushed urban administrations to develop smarter solutions. At this point, the concept of the smart city has emerged as an important approach that aims to offer technology-supported and sustainable solutions to the problems faced by modern cities. Many components, ranging from traffic lights and security cameras to air quality sensors, smart meters, public transportation systems, and waste management infrastructures, have become part of this structure. The ability of these components to communicate with one another and

[1] Assoc. Prof. Dr., Zonguldak Bulent Ecevit University, Department of Computer Engineering, Orcid: 0000-0001-6970-3239

share data in real time demonstrates the importance of Internet of Things (IoT) technologies for smart cities (Zanella et al., 2014).

The Internet of Things acts as a bridge between the physical environment and digital systems in smart cities. Through sensors and connected devices deployed across urban areas, a wide range of data can be continuously collected, including traffic flow, energy consumption, environmental conditions, security incidents, and citizen mobility. However, the key issue is not merely the collection of data, but how this data is processed, interpreted, and transformed into decision-making mechanisms. For this reason, artificial intelligence (AI) has become a complementary and strengthening element of IoT-based smart city applications. Methods such as machine learning, deep learning, fuzzy logic, optimization algorithms, and explainable AI enable the effective analysis of large volumes of data generated by cities. For example, AI-based approaches can be used for many problems, such as predicting traffic congestion, forecasting energy demand, monitoring air pollution levels, detecting abnormal situations, and allocating resources more efficiently (Syed et al., 2021).

The combined use of IoT and artificial intelligence in smart cities makes decision-making processes faster and more flexible. In traditional urban management practices, decisions are often based on historical data, manual observations, or fixed rules. In contrast, IoT- and AI-supported systems can use real-time data to produce decisions that are more up to date and more suitable for the current situation. Automatically adjusting traffic signal durations during peak hours or planning waste collection routes according to bin occupancy levels can be given as examples of applications that benefit from such real-time data. In these applications, data are collected from different devices, in different formats, and at different time intervals. These data may often be incomplete, noisy, or inconsistent. In addition, the amount of data generated by city-scale

systems is considerably high. In this context, data fusion approaches enable smart city applications to combine information from different data sources and produce more meaningful and reliable results (Pik et al., 2019).

The transition from traditional methods to smart city systems offers important opportunities, but it also brings several critical challenges. The presence of a large number of network-connected devices across a city makes these systems more vulnerable to security risks such as unauthorized access, data manipulation, denial-of-service attacks, and privacy violations. This issue becomes even more complex when data related to citizens' location, mobility, energy consumption, or health are processed. In such cases, the problem is no longer only technical, it also gains ethical, legal, and social dimensions. In addition to cybersecurity risks, factors such as infrastructure deficiencies, financial constraints, and limited trust in these technologies are among the main barriers to the adoption of AI and IoT in smart cities (Wang et al., 2021). The explainability of artificial intelligence models is also an important requirement. It is not enough for these systems to produce accurate results, it should also be possible to understand which data and decision processes lead to these results. Therefore, explainable artificial intelligence approaches are considered an important research area for making IoT-based AI systems in smart cities more understandable, auditable, and trustworthy (Javed et al., 2023).

There is a growing body of literature on IoT and AI-supported smart city applications. However, a considerable part of these studies focuses on different application areas, data types, and methodological approaches. While some studies address traffic and transportation problems, others concentrate on energy efficiency, environmental monitoring, security, healthcare, or urban planning. This chapter examines the use of IoT and AI integration in the context of smart cities. The studies reviewed, filtered from 2025 to

the present, are comparatively evaluated in terms of their methods, application areas, handled problems, contributions and reported findings.

IoT and AI-Based Approaches for Smart Cities

There are many studies on IoT and AI-based smart city systems. These studies focus on different application areas, such as transportation, energy management, environment and air quality, waste and water management, smart homes, security, and emergency response. They also address a wide range of technical issues, including cybersecurity, data processing, storage systems, monitoring centers, lightweight communication protocols, management, and optimization.

In (Ali et al., 2025), cybersecurity problems in advanced communication networks for AI and IoT-based smart cities were examined. In the study, AI-based security approaches such as machine learning (ML), deep learning (DL), anomaly detection, federated learning (FL), reinforcement learning (RL), generative adversarial networks (GAN), and graph neural networks (GNN) were discussed. Existing solutions in this field were classified, and future research directions were presented from technical, architectural, and policy-oriented perspectives. In (Zhang et al., 2025), an IoT and AI-supported green supply chain framework was proposed to reduce carbon emissions in the manufacturing sector. Data obtained from IoT sensors were processed in cloud and fog computing environments. Carbon emission prediction and decision optimization were carried out using a support vector regression (SVR) model improved with particle swarm optimization (PSO). It was reported that the model achieved high predictive performance and contributed to the reduction of carbon emissions.

In (Li & Wang, 2025), a DL-based multi-label regression model was proposed to assign patents in the fields of smart cities and

industrial IoT to technology areas more accurately. In the study, models such as text convolutional neural network (TextCNN), deep pyramid convolutional neural network (DPCNN), recurrent convolutional neural network (RCNN), bidirectional long short-term memory (Bi-LSTM), bidirectional encoder representations from transformers (BERT), and GPT-4 were compared. It was emphasized that the BERT model produced the most successful results, while GPT-4 was a strong alternative, especially in few-shot learning settings. In (Patthi et al., 2025), a framework integrating IoT, blockchain, and AI was proposed for environmental monitoring and resource management in agricultural areas. Temperature, humidity, soil moisture, CO_2, and light data were collected through IoT sensors. Data integrity was ensured through blockchain, while irrigation, misting, and cooling operations were automated using AI and smart contracts. It was reported that this model reduced water and energy consumption, improved plant health, and provided secure management with low latency. In (Macko et al., 2025), an IoT and AI-supported smart building energy management system was examined through an office building case. Using temperature, humidity, CO_2, occupancy, and energy consumption data obtained from IoT sensors, energy consumption, cost savings, and CO_2 reduction were predicted with long short-term memory (LSTM) and convolutional neural network (CNN) models. It was stated that the system reduced energy consumption and costs, with the best accuracy being approximately 82%.

In (Alkhudhayr, 2025), an IoT-supported image-based parking slot detection and occupancy classification system was proposed for smart city traffic management. In the study, parking spaces were detected using the single shot multibox detector (SSD)-ShuffleNet model, occupied and vacant spaces were classified using optimum support vector machine (OSVM), and the model parameters were optimized with the improved weight-based white

shark optimizer (IW-WSO) algorithm. It was emphasized that the system offers a real-time and lightweight smart parking management approach as an alternative to costly sensor-based solutions. In (Qaffas, 2025), an AI-supported distributed IoT communication architecture was proposed to optimize traffic flow in real time. Data obtained from cameras and traffic sensors were processed at edge nodes, and vision transformer (ViT) was used for visual traffic detection, while temporal convolutional network (TCN) was used to predict temporal traffic patterns. It was reported that this distributed edge-based structure provided lower latency than centralized and cloud-based systems and achieved improvements in vehicle delay, stop time, and traffic density. In (Reis, 2025), a multimodal data fusion framework was proposed for secure and sustainable green mobility in smart cities. Traffic flow, weather conditions, vehicle data, and mobility data such as GPS were combined, traffic congestion was predicted using long short-term memory (LSTM), and dynamic route optimization was performed using deep Q-network (DQN). It was stated that the method provided a scalable and secure solution by reducing travel time, energy consumption, and CO_2 emissions.

In (Salim et al., 2025), an IoT-based waste management system was proposed for sustainable smart cities. Ultrasonic sensors, load cells, GPS, and GSM modules were integrated into waste bins to monitor fill levels and location information in real time. It was shown that the system reduced fuel consumption, labor costs, and total operational costs, while improving waste collection efficiency. In (Shah et al., 2025), a low-cost digital twin system supported by IoT and ML was proposed for solid waste management in smart cities. Waste level, gas, temperature, humidity, and GPS data were monitored, and linear regression, artificial neural network (ANN), support vector machine (SVM) regression, and Bayesian Ridge models were compared to predict waste generation. It was stated that

the model achieved the best results with ANN and that the low-cost digital twin approach could be used for real-time monitoring and prediction in waste management.

In (Sallam et al., 2025), a ML-based framework was proposed to improve city services by making use of IoT data. Five different smart city scenarios were considered: transportation, smart agriculture, smart grid, security, and Industry 4.0. Random forest (RF), extra trees (ET), histogram gradient boosting (HGB), and their bagging-based ensemble structure were compared. It was stated that different models achieved the best performance in different scenarios, however, in the transportation scenario in particular, HGB reached an accuracy of 99.99%. In (Ragab et al., 2025), a FL-based framework was proposed for privacy threats detection in sustainable smart cities. Harris hawks optimization (HHO) was used for feature selection on big IoT data, stacked sparse auto-encoder (SSAE) was used to classify cyber threats, and the walrus optimization algorithm (WOA) was used to improve the model parameters. It was reported that the model achieved high accuracy on a dataset consisting of 12 classes and 30,000 samples.

Tablo 1 Details of reviewed papers

Paper	Problem	Contributions	Used AI Methods	Application Area	Light-weight	Results
(Ali et al., 2025)	Attack surface, data privacy, DDoS, unauthorized access, 5G security risks	Multi-layer security classification, AI-based solution analysis	ML, DL, AD, FL, RL, TL, GAN, GNN	Transportation, smart grid, healthcare, public safety, environmental monitoring	No	No experimental evaluation. Classification, comparison, challenges, recommendations
(Zhang et al., 2025)	Sustainable manufacturing, green supply chain, emission prediction	AI-IoT-GSC framework, carbon reduction, decision optimization	SVR, PSO, ML optimization	Smart manufacturing, industry	No	R^2=0.945, MSE=0.006, RMSE=0.102, MAE=0.101, MAPE=0.013
(Li & Wang	International patent	MLR model, D0MSELoss,	TextCNN,	Industrial IoT	No	BERT R=0.7958,

, 2025)	classification uncertainty, label imbalance	two-stage regression, BERT, GPT4 comparison	DPCNN, RCNN, Bi-LSTM, BERT, GPT4			R^2=0.5976; IIoT: BERT R=0.8292, R^2=0.5916
(Patthi et al., 2025)	Environmental monitoring, data security, resource waste, lack of real-time response	IoT, Blockchain, and AI framework, smart contracts, automated management	No algorithm. Prediction, resource optimization	Smart agriculture and green spaces	Partially – LoRa WAN and PBFT support	30% reduction in water and energy consumption, 15% improvement in plant health, response time less than 1 s, ensuring data integrity
(Macko et al., 2025)	Lack of holistic data-driven energy management, scalability issues	AI-IoT energy management, energy and cost optimization	LSTM, CNN, Dense NN, RL, anomaly detection	Smart building, smart grid	Partially – edge and lightweight model	Acc=%81.79, RMSE=0.01, energy savings of %20.9–33.8, cost reduction of %18–35, CO_2 reduction of %25.1–40.7
(Alkhudhayr, 2025)	Scalability, lighting and weather variations, real-time detection	Image-based smart parking system, parameter optimization	SSD-ShuffleNet, OSVM, SVM-RBF, IW-WSO	Smart parking, traffic management, smart transportation	Yes – ShuffleNet, edge and FLOPs	Acc=%96.5, Prec=%95.8, Rec=%96.2, F1=%96.0, AUC=0.97, mAP=%93.2, 12–50 ms
(Qaffas, 2025)	Scalability issue, adaptation to dynamic traffic conditions	Distributed IoT traffic architecture, edge-based decision-making, hybrid ViT-TCN model	ViT, TCN, Ranger optimizer, multitask learning	Smart transportation, traffic management	Partially – low latency through edge and MQTT	40.4% reduction in vehicle delay, 42.2% in stop time, 29.5% in density. 32.7% increase in throughput, decision latency of 64.9 ms
(Reis, 2025)	Lack of real-time data fusion, scalability	IoT-AI mobility framework, multimodal data fusion, secure route optimization	LSTM, DQN, DL, AI-based anomaly detection	Smart transportation, green mobility	Partially – PBFT and edge	20% reduction in travel time, 15% in energy consumption, 10% in CO_2 emissions, and 43% in the congestion index
(Salim et al., 2025)	Manual waste collection, high route,	IoT-based waste management, real-time monitoring,	No algorithm. AI- and ML-based	Smart waste management,	Partially – low-power	28% reduction in fuel consumption, 40% in labor costs, and

	fuel, and time costs	route optimization	route optimization	sustainable city	hardware	14.6% in total costs. 41.5% increase in collection efficiency
(Shah et al., 2025)	Lack of real time tracking, need for waste prediction, high cost	Low-cost digital twin, ML-based waste prediction	ANN, Linear Regression, SVM Regressor, Bayesian Ridge	Smart waste management, sustainable city	Yes – ESP32, LoRa and low power	Best score for ANN: R^2=0.944, PCC=0.998, WIoA=0.987, RMSE=8130.27 MAE=7656.989
(Sallam et al., 2025)	Need to improve city services	ML framework, five smart city domains, comparative model analysis	RF, ET, HGB, Bagging, SMOTE	Transportation, smart agriculture, smart grid, security, Industry 4.0	No	Transportation HGB=%99.99, Agriculture RF=%99.79, Smart grid ET=%96.87, Security RF=%95.93, Industry 4.0 RF=%99.54
(Ragab et al., 2025)	Data privacy, cyber threats, big IoT data	FL-based privacy, threat detection, and optimization	FL, HHO, SSAE, WOA	Smart city security, privacy	No	Acc=%99.47, Prec=%97.20, Rec=%96.84, F1=%96.92, AUC=%98.28, PT=4.51 s

Discussion

When the reviewed papers are evaluated as a whole, it is clear that the integration of IoT and AI has a wide range of uses in smart cities. These studies generally cover different areas such as transportation, traffic management, smart parking, waste management, energy efficiency, smart buildings, smart manufacturing, environmental monitoring, smart agriculture, and cybersecurity. However, the most visible trend appears to be in transportation and mobility. Scopes such as smart parking, traffic optimization, and green mobility are among the most important application areas of IoT and AI integration, mainly because they require real-time decision-making.

When the reviewed studies are evaluated in terms of the handled problems, common limitations stand out. These include the

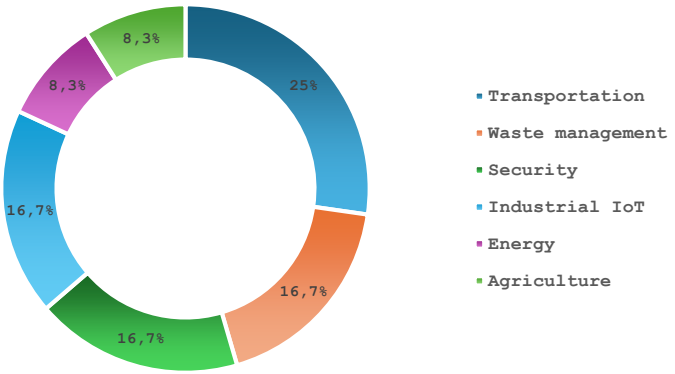
lack of real-time data processing, scalability problems in existing systems, high operational costs, data security and privacy concerns, resource waste, and the inability to adapt to dynamic urban conditions. In transportation-oriented studies, low latency, reducing traffic congestion, and route optimization are among the main priorities. In waste management, manual collection processes, unnecessary route usage, fuel consumption, and the lack of real-time monitoring are seen as the main problems. In security-oriented studies, the expansion of the attack surface, DDoS attacks, unauthorized access, and data privacy are among the key research issues.

There is diversity in terms of the AI methods used in the reviewed studies. In classical ML, methods such as RF, ET, HGB, SVR, SVM, Bayesian Ridge, Bagging, and SMOTE are commonly used. In deep learning, models such as ANN, CNN, LSTM, Dense NN, SSAE, ViT, TCN, and BERT stand out. In addition, optimization-based approaches such as PSO, HHO, WOA, IW-WSO, and Ranger optimizer are used to improve model performance or optimize model parameters. However, in some studies, the AI method is not clearly named. Instead, it is described in general terms, such as AI or ML-based prediction, route optimization, or resource optimization. This indicates that methodological transparency is not at the same level across all studies in the literature.

When the results of the reviewed studies are considered, AI and IoT integration appears to improve both predictive performance and operational efficiency in smart city applications. The systems developed in different application areas provide several benefits, including faster decision-making, lower latency, more efficient use of resources, reduced energy consumption, lower costs, and limited environmental impact. In addition, real-time monitoring, automated decision support mechanisms, and optimization processes contribute to the more flexible and sustainable management of urban services.

The distribution of smart city application areas across the reviewed studies is shown in Graph 1. Overall, the findings indicate that AI and IoT integration not only improves technical performance, but also provides important advantages in terms of service quality, sustainability, and management effectiveness in smart cities.

Graph 1 Distribution of Application Areas in Smart Cities



One of the key requirements in many IoT applications is the development of lightweight systems. Among the reviewed studies, only %16,7 clearly adopted a lightweight approach. While %41,7 of the reviewed studies included partially lightweight features, the remaining %41,7 did not present a lightweight design. In the studies evaluated as partially lightweight, this feature did not come directly from the model architecture, but rather from supporting components such as edge computing, LoRaWAN, and low-power hardware. Therefore, future studies are expected to place greater emphasis on TinyML, model compression, pruning, quantization, lightweight models compatible with edge AI, and low-energy architectures.

Conclusion

This chapter provides a detailed review, analysis, and evaluation of recent studies on IoT and AI integration in smart cities. The reviewed studies show that this integration makes important contributions to various areas, including transportation, energy, waste management, environmental monitoring, smart manufacturing, and cybersecurity. The main trend is a transition from simple IoT based data collection toward AI supported real time prediction, edge based decision making, optimization, and sustainability oriented smart city management. However, real world city scale implementation, lightweight architectures, data privacy, explainability, and scalability still stand out as areas that need further development. Therefore, future studies should focus not only on developing high performance models, but also on producing low cost, reliable, explainable, and real time applicable smart city solutions.

References

Ali, J., Kumar, S., Jiang, W., Alenezi, A. M., Islam, M., Ibrahim, Y., & Mehmood, A. (2025). A deep dive into cybersecurity solutions for AI-driven IoT-enabled smart cities in advanced communication networks. Computer Communications, 229(July 2024), 108000. https://doi.org/10.1016/j.comcom.2024.108000

Alkhudhayr, H. (2025). Engineering Applications of Artificial Intelligence Internet of things based parking slot detection and occupancy classification for smart city traffic management. Engineering Applications of Artificial Intelligence, 152(February), 110802. https://doi.org/10.1016/j.engappai.2025.110802

Javed, A. R., Ahmed, W., Pandya, S., Kumar, P., Maddikunta, R., Alazab, M., & Gadekallu, T. R. (2023). A Survey of Explainable Artificial Intelligence for Smart Cities. Electronics, 1–40. https://doi.org/10.3390/electronics12041020

Li, M., & Wang, L. (2025). Leveraging patent classification based on deep learning : The case study on smart cities and industrial Internet of Things. Journal of Informetrics, 19(1), 101616. https://doi.org/10.1016/j.joi.2024.101616

Macko, M., Bednarek, T., Rojek, I., Mikołajewski, D., & Mrozi, A. (2025). Internet of Things Applications for Energy Management in Buildings Using Artificial Intelligence — A Case Study. Energies, 1–28. https://doi.org/10.3390/en18071706

Patthi, S., Karthiga, M., Priyanka, K., Kumar, S., Venkata, P., Bhagyalakshmi, L., Sreelatha, P., & Alagarsamy, M. (2025). Sustainable Computing : Informatics and Systems Integration of Internet of Things blockchain and artificial intelligence for scalable and secure precision environmental management. Sustainable Computing: Informatics and Systems, 48(February), 101251. https://doi.org/10.1016/j.suscom.2025.101251

Pik, B., Lau, L., Hasala, S., Zhou, Y., Ul, N., Yuen, C., Zhang, M., & Tan, U. (2019). A survey of data fusion in smart city applications. Information Fusion, 52(January), 357–374. https://doi.org/10.1016/j.inffus.2019.05.004

Qaffas, A. A. (2025). AI - driven distributed IoT communication architecture for smart city traffic optimization. The Journal of Supercomputing, 81, 916. https://doi.org/10.1007/s11227-025-07426-0

Ragab, M., Ashary, E. B., Alghamdi, B. M., Aboalela, R., Alsaadi, N., Maghrabi, L. A., & Allehaibi, K. H. (2025). Advanced artificial intelligence with federated learning framework for privacy-preserving cyberthreat detection in IoT-assisted sustainable smart cities. Scientific Reports, 15, 4470. https://doi.org/10.1038/s41598-025-88843-2

Reis, M. J. C. S. (2025). Internet of Things and Artificial Intelligence for Secure and Sustainable Green Mobility : A Multimodal Data Fusion Approach to Enhance Efficiency and Security. Multimodal Technologies and Interaction, 9(5), 39. https://doi.org/10.3390/mti9050039

Salim, K., Yarubi, A., Khairy, S. O. F., Hossain, S. M. E., & Hayder, G. (2025). Internet of Things-Driven Waste Management : Paving the Way for Sustainable Smart Cities. Processes,13(4), 1140. https://doi.org/10.3390/pr13041140

Sallam, K. M., Alrashdi, I., Ali, A. M., & Radwan, I. (2025). A robust framework utilizing artificial intelligence to enhance services in smart city environments. Cluster Computing, 28, 741. https://doi.org/10.1007/s10586-025-05431-9

Shah, K. B., Deepesh, S. V., Guragain, P., & Panigrahi, R. (2025). Advancing smart city sustainability with Internet of Things and artificial intelligence aided low-cost digital twin systems for waste management. Microsystem Technologies, 31, 2783–2796. https://doi.org/10.1007/s00542-024-05827-4

Syed, A. S., Sierra-sosa, D., Kumar, A., & Elmaghraby, A. (2021). IoT in Smart Cities: A Survey of Technologies , Practices and Challenges. Smart Cities, 4(2), 429-475. https://doi.org/10.3390/smartcities4020024

Wang, K., Zhao, Y., & Gangadhari, R. K. (2021). Analyzing the Adoption Challenges of the Internet of Things (IoT) and Artificial Intelligence (AI) for Smart Cities in China. Sustainability, 13(19), 10983. https://doi.org/10.3390/su131910983

Zanella, A., Member, S., Bui, N., Castellani, A., Vangelista, L., Member, S., & Zorzi, M. (2014). Internet of Things for Smart Cities. IEEE Internet of Things Journal, 1(1), 22–32. https://doi.org/10.1109/JIOT.2014.2306328

Zhang, L., Innab, N., Mohamed, S., Pan, Y., & Zhang, Y. (2025). Artificial intelligence-driven internet of things-based green supply chain for carbon reduction in sustainable manufacturing. Journal of Environmental Management, 389(November 2024), 126170. https://doi.org/10.1016/j.jenvman.2025.126170

CHAPTER 1

ACCESSIBILITY GUIDELINES IN MODERN DIGITAL SYSTEMS: A COMPREHENSIVE REVIEW OF WCAG 2.1

AHMET TOPRAK[1]

Introduction

The proliferation of digital technologies has fundamentally transformed the ways in which information is produced, accessed, and consumed [1]. From web-based platforms and mobile applications to embedded systems and multi-device ecosystems, digital environments now mediate a wide range of daily activities, including communication, education, healthcare, commerce, and public services. Within this rapidly evolving landscape, ensuring that digital systems are accessible to all users has become a central concern for researchers, developers, policymakers, and organizations [2].

Digital accessibility refers to the design and development of digital content and interfaces that can be effectively perceived, understood, navigated, and interacted with by individuals with diverse abilities. This includes users with visual, auditory, motor, and

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

cognitive impairments, as well as those experiencing situational limitations such as temporary disabilities, aging-related challenges, or environmental constraints [3]. Accessibility considerations are therefore closely aligned with the broader concept of inclusive design, which aims to create systems that accommodate the widest possible range of users without the need for specialized adaptations.

The increasing reliance on digital systems has also elevated accessibility from a desirable feature to a fundamental requirement. In many regions, accessibility is not only a technical or ethical consideration but also a legal obligation, enforced through regulations and standards that mandate equal access to digital services. These developments reflect a growing recognition that inaccessible systems can create significant barriers, limiting participation in essential aspects of modern life and reinforcing existing inequalities.

To address these challenges, structured guidelines and standards have been developed to support the design of accessible digital systems. Among these, the Web Content Accessibility Guidelines (WCAG) 2.1, established by the World Wide Web Consortium, have emerged as one of the most widely adopted and influential frameworks. Although originally introduced to guide web accessibility, WCAG 2.1 has evolved into a comprehensive and technology-agnostic standard that can be applied across a variety of digital platforms, including mobile applications, responsive interfaces, and interactive systems [4].

WCAG 2.1 builds upon earlier versions by introducing additional success criteria that specifically address challenges associated with modern digital environments. These include issues related to mobile interaction patterns, such as touch gestures and orientation changes, as well as accessibility considerations for users with low vision and cognitive limitations. The guidelines are structured around four foundational principles—Perceivable,

Operable, Understandable, and Robust (POUR)—which collectively define the essential requirements for accessible content. Each principle is further elaborated through a set of guidelines and testable success criteria, enabling both conceptual clarity and practical implementation [5].

The application of WCAG 2.1 in contemporary digital systems presents several challenges. One of the primary difficulties lies in the diversity of platforms and technologies that must be considered [6]. Unlike traditional web environments, modern systems often involve complex interactions across multiple devices, operating systems, and input modalities. Ensuring consistent accessibility across such heterogeneous environments requires not only adherence to guidelines but also a deep understanding of user needs and context-specific design considerations.

Another significant challenge is the gap between theoretical compliance and practical usability. While many systems may meet certain accessibility criteria at a technical level, they may still present usability barriers that affect real-world user experience. This limitation highlights the importance of combining automated evaluation tools with manual testing and user-centered design approaches. Accessibility cannot be fully achieved through rule-based validation alone; it requires iterative evaluation, continuous refinement, and active involvement of users with diverse abilities [7].

Recent advancements in artificial intelligence and machine learning have introduced new opportunities for enhancing accessibility in digital systems. Technologies such as automatic speech recognition, natural language processing, and computer vision are increasingly used to support features like real-time captioning, text simplification, image description, and adaptive user interfaces [8]. These innovations have the potential to significantly improve accessibility, particularly in dynamic and content-rich

environments. At the same time, they introduce new challenges related to system reliability, bias, interpretability, and ethical considerations, which must be carefully addressed to ensure equitable outcomes.

The growing complexity of digital ecosystems, combined with the evolving nature of accessibility requirements, underscores the need for a comprehensive and adaptable framework. WCAG 2.1 [9] provides a structured foundation for addressing accessibility across a wide range of contexts, yet its effective implementation depends on the integration of technical, organizational, and user-centered perspectives. This chapter examines the role of WCAG 2.1 [10] within modern digital systems, focusing on its principles, structure, and application across different platforms, as well as the challenges and opportunities associated with its adoption.

Background of Digital Accessibility

Digital accessibility has emerged as a fundamental research and development area within modern computing, driven by the increasing dependence on digital systems for essential aspects of daily life[11]. The field is inherently multidisciplinary, combining principles from software engineering, human-computer interaction, cognitive science, and inclusive design. Its primary objective is to ensure that digital environments are usable by individuals with diverse physical, sensory, and cognitive abilities, without requiring specialized adaptations or alternative versions of content.

The early stages of accessibility research were closely associated with desktop computing environments, where interaction was primarily based on keyboard and mouse input. During this period, accessibility solutions were largely implemented through assistive technologies such as screen readers, screen magnifiers, alternative pointing devices, and speech recognition systems. These tools were designed to bridge the gap between system interfaces and

user capabilities, enabling individuals with disabilities to interact with digital content more effectively.

With the rapid expansion of the internet, accessibility considerations shifted toward web-based systems. The increasing complexity of web applications, combined with the diversity of users accessing content through different devices and browsers, highlighted the need for standardized accessibility frameworks. This led to the development of structured guidelines aimed at ensuring consistent accessibility practices across platforms and organizations.

Over time, digital ecosystems have evolved significantly beyond traditional web environments. Modern systems now include mobile applications, responsive web interfaces, wearable technologies, voice-based assistants, augmented reality systems, and Internet of Things (IoT) devices. Each of these platforms introduces unique interaction models and technical constraints. For instance, mobile applications rely heavily on touch gestures, screen orientation changes, and variable screen sizes, while voice-based systems depend on speech recognition accuracy and natural language understanding. These differences have expanded the scope of accessibility challenges and increased the need for adaptable and technology-independent guidelines [12].

Within this context, accessibility standards have played a crucial role in guiding both research and industrial practice. Among the most influential frameworks is the WCAG 2.1 developed by the World Wide Web Consortium [13]. Although originally designed to address web content accessibility, WCAG 2.1 has evolved into a broader framework that is applicable to a wide range of digital systems. Its technology-agnostic structure allows it to be adapted to mobile applications, desktop software, and other interactive systems.

WCAG 2.1 builds upon earlier versions by introducing additional success criteria that address accessibility challenges

specific to modern digital environments. These include requirements related to mobile accessibility, such as support for different screen orientations, improved touch target sizes, and enhanced usability for users with low vision and cognitive impairments. The guidelines are organized into a hierarchical structure consisting of principles, guidelines, and testable success criteria, which collectively provide both conceptual clarity and practical implementation pathways [14].

Despite the existence of well-defined standards, the implementation of accessibility remains a complex and often inconsistent process in practice. One of the primary challenges is the gap between guideline specification and real-world system development. Accessibility requirements must often be integrated into existing software architectures, which may not have been designed with accessibility in mind. This creates technical and organizational barriers that can limit effective adoption.

Another significant challenge is the diversity of user needs and usage contexts. Accessibility is not limited to permanent disabilities but also includes situational limitations such as temporary injuries, environmental conditions, or device constraints. This variability makes it difficult to design solutions that are universally effective across all scenarios. As a result, accessibility evaluation must consider a combination of automated testing tools and human-centered evaluation methods to capture both technical compliance and actual usability.

The increasing complexity of modern digital systems further complicates accessibility implementation. Applications are now expected to operate seamlessly across multiple devices, screen sizes, and interaction modalities. This cross-platform requirement introduces additional constraints that must be considered during the design and development process. Ensuring consistency in accessibility behavior across these heterogeneous environments remains a non-trivial challenge.

In parallel with these challenges, recent advancements in artificial intelligence and machine learning have introduced new opportunities for improving accessibility. Techniques such as automatic image captioning, speech-to-text conversion, text simplification, and adaptive user interfaces are increasingly being integrated into digital systems. These technologies have the potential to reduce accessibility barriers by automating certain aspects of content adaptation. However, they also introduce new concerns related to transparency, bias, and reliability, which must be carefully addressed in practical implementations.

The growing importance of accessibility in digital systems has also led to increased attention from regulatory bodies and standardization organizations. Various international regulations require compliance with accessibility standards to ensure equal access to digital services. These regulatory frameworks reinforce the importance of accessibility as both a technical requirement and a legal obligation.

The combination of evolving technologies, diverse user needs, and regulatory pressures has made accessibility a central consideration in modern system design. Frameworks such as WCAG 2.1 continue to serve as foundational references for addressing these challenges, while ongoing research explores new methods for improving accessibility in increasingly complex digital environments[15].

WCAG 2.1 Overview

The Web Content Accessibility Guidelines (WCAG) 2.1 constitute a structured framework developed by the World Wide Web Consortium to provide measurable and technology-independent criteria for digital accessibility. WCAG has undergone a series of evolutionary stages, beginning with WCAG 1.0, followed by WCAG

2.0, and subsequently extended to WCAG 2.1 in response to emerging accessibility requirements in modern digital ecosystems.

WCAG 1.0 primarily focused on HTML-based web content and provided early guidance for ensuring accessibility in static web pages [16]. However, the rapid evolution of web technologies and the transition toward dynamic, interactive, and application-like web environments revealed significant limitations in the initial framework. WCAG 2.0 introduced a more robust and flexible structure by abstracting accessibility requirements into principles that are independent of specific technologies. This shift enabled broader applicability across different platforms and development paradigms.

WCAG 2.1 further extends this framework by addressing accessibility gaps that became increasingly relevant with the widespread adoption of mobile devices and modern interaction paradigms. The updates introduced in WCAG 2.1 specifically target users with low vision, cognitive limitations, and users interacting through mobile interfaces. These enhancements reflect the growing diversity of digital environments and the need for accessibility standards that extend beyond traditional desktop-based web usage.

At the core of WCAG 2.1 lies the POUR model, which defines four foundational principles: Perceivable, Operable, Understandable, and Robust. These principles serve as the conceptual foundation for all guidelines and success criteria defined within the framework [17]. Each principle addresses a distinct aspect of accessibility and contributes to a holistic approach to inclusive design.

The Perceivable principle focuses on ensuring that information and interface components are presented in ways that can be perceived by users, regardless of sensory limitations. This includes requirements related to text alternatives for non-text

content, adaptable content structures, and sufficient contrast between foreground and background elements. The goal is to ensure that users can access information through multiple sensory modalities.

The Operable principle emphasizes the importance of interface navigation and interaction. It requires that all functionality be available through various input methods, including keyboard navigation and assistive technologies. It also addresses issues such as time constraints, navigational consistency, and avoidance of content that may trigger seizures or physical discomfort. This principle is particularly relevant in mobile and touch-based environments where interaction mechanisms differ significantly from traditional desktop systems.

The Understandable principle focuses on ensuring that content and interface behavior are predictable and easy to comprehend. This includes requirements related to readable text, consistent navigation patterns, and clear input assistance mechanisms. The principle aims to reduce cognitive load and prevent user confusion, particularly in complex or information-dense systems.

The Robust principle ensures that content is compatible with a wide range of user agents, including assistive technologies. It emphasizes the importance of using standardized markup, semantic structure, and interoperable coding practices. This principle is essential for ensuring long-term accessibility across evolving technologies and platforms.

WCAG 2.1 introduces additional success criteria that extend the applicability of these principles to modern digital environments. These include requirements for improved mobile accessibility, such as support for different screen orientations and enhanced touch target sizing. Additional criteria address cognitive accessibility by

improving guidance for input assistance, error prevention, and content clarity.

The guidelines within WCAG 2.1 are organized into three levels of conformance: Level A, Level AA, and Level AAA. Level A represents the minimum accessibility requirements, addressing the most critical barriers to access. Level AA includes a broader set of criteria that significantly improve usability and is commonly adopted as the standard compliance level in both public and private sectors. Level AAA represents the highest level of accessibility, although it is not typically required for full compliance due to its stringent requirements.

The structured nature of WCAG allows each success criterion to be testable, enabling both automated and manual evaluation methods. However, the interpretation of certain criteria may vary depending on context, particularly in complex interactive systems such as mobile applications and dynamic web platforms. This variability necessitates careful consideration during implementation to ensure that accessibility requirements are meaningfully addressed rather than interpreted in a purely technical manner.

The applicability of WCAG 2.1 extends beyond traditional web environments, influencing the design of mobile applications, desktop software, and cross-platform systems. Its technology-agnostic structure allows it to be integrated into diverse development workflows, making it a foundational reference in both academic research and industrial practice [18].

Implementation Challenges and Practices

The implementation of accessibility guidelines in real-world digital systems presents a range of technical, organizational, and methodological challenges. Although the WCAG 2.1 provide a well-defined and structured framework, translating these abstract

requirements into functional system design often requires significant adaptation at the development level.

One of the primary challenges arises from the integration of accessibility requirements into existing software development workflows. In many cases, systems are initially designed without accessibility considerations, resulting in architectures that are not inherently compatible with assistive technologies or inclusive design principles. Retrofitting accessibility into such systems can require extensive modifications to user interface components, interaction logic, and underlying code structure.

Another important difficulty is the discrepancy between automated evaluation and real-world usability. While automated tools can efficiently detect certain types of accessibility issues, such as missing alternative text or insufficient color contrast, they are limited in their ability to evaluate more complex interaction patterns and user experience factors. Tools such as static analyzers and browser-based accessibility checkers provide valuable insights, but they cannot fully capture cognitive load, navigational clarity, or contextual usability.

The diversity of platforms further complicates implementation. Modern digital systems operate across a wide range of environments, including desktop browsers, mobile applications, and hybrid interfaces. Each platform introduces distinct interaction paradigms, such as keyboard-based navigation, touch gestures, or voice commands. Ensuring consistent accessibility behavior across these heterogeneous environments requires careful design abstraction and platform-specific adaptation.

Mobile accessibility introduces additional challenges due to limited screen space, dynamic layout behavior, and gesture-based interaction models. Unlike desktop environments, mobile interfaces rely heavily on context-sensitive navigation and adaptive content

presentation. These constraints necessitate careful consideration of element sizing, focus management, and responsive design techniques to ensure compliance with accessibility requirements.

In practice, accessibility implementation is often supported by a combination of automated tools and manual evaluation techniques. Automated tools provide rapid identification of structural issues, while manual testing enables deeper assessment of usability and interaction quality. Manual evaluation typically involves keyboard-only navigation testing, screen reader compatibility checks, and user testing with individuals who have disabilities. This combination is necessary to address both technical compliance and experiential accessibility.

Despite the availability of guidelines and tools, organizational factors significantly influence the effectiveness of accessibility implementation. Limited awareness among developers, insufficient training, and lack of dedicated accessibility processes often result in incomplete or inconsistent adoption of guidelines. In some cases, accessibility is treated as a final-stage compliance requirement rather than an integral part of the design and development lifecycle.

Recent developments [19] in development frameworks and design systems have introduced components with built-in accessibility support. However, these components still require correct configuration and contextual adaptation to ensure full compliance with accessibility standards. Improper usage or misunderstanding of these components can lead to unintended accessibility barriers.

The increasing complexity of user interfaces, particularly in applications with rich interactive elements, further highlights the importance of structured accessibility practices. Dynamic content updates, asynchronous interactions, and real-time data rendering

introduce additional challenges for maintaining accessibility states and ensuring compatibility with assistive technologies.

Evaluation Tools and Platform-specific Accessibility Assessment

Accessibility evaluation is a critical component of implementing and validating compliance with the WCAG 2.1. Evaluation approaches are generally divided into automated testing tools, manual inspection methods, and user-based testing procedures. Each approach contributes differently to identifying accessibility barriers in digital systems [20].

Automated evaluation tools are widely used due to their efficiency and scalability. These tools analyze source code, rendered pages, or application structures to detect common accessibility issues such as missing alternative text, improper heading hierarchy, insufficient color contrast, and incorrect ARIA usage. Commonly used tools include browser-based extensions and integrated development environment plugins that provide real-time feedback during development. While these tools are effective for identifying structural and rule-based violations, they are limited in their ability to assess contextual usability and interaction quality.

Manual evaluation methods are required to complement automated tools. These methods involve systematic inspection of digital interfaces using assistive technologies such as screen readers, keyboard-only navigation, and alternative input devices. Manual testing is particularly important for evaluating dynamic content, complex navigation structures, and interactive components where automated tools may not fully capture accessibility issues. This type of evaluation requires domain expertise and a detailed understanding of accessibility principles.

User-based testing represents another essential layer of accessibility evaluation. In this approach, individuals with

disabilities are directly involved in testing digital systems under realistic usage conditions. This method provides valuable insights into real-world usability challenges that cannot be identified through automated or expert-based evaluation alone. User feedback often reveals issues related to cognitive load, interaction clarity, and task completion efficiency.

Platform-specific differences significantly influence accessibility evaluation strategies. In web environments, accessibility assessment typically focuses on HTML semantics, document structure, keyboard navigation, and compatibility with assistive technologies such as screen readers. Browser-based tools play a central role in identifying compliance issues in this context.

In contrast, mobile accessibility evaluation introduces additional dimensions due to the nature of touch-based interaction and device heterogeneity. Mobile applications must be evaluated for gesture accessibility, screen orientation adaptability, touch target sizing, and compatibility with mobile screen readers such as VoiceOver and TalkBack [21]. Unlike web environments, mobile platforms require testing across multiple operating systems, screen sizes, and device configurations, which increases evaluation complexity.

Hybrid and cross-platform applications further complicate accessibility assessment. These systems combine web technologies with native components, leading to inconsistencies in accessibility behavior across different layers of the application. Ensuring consistent accessibility in such environments requires coordinated evaluation across both web and native interfaces.

The integration of accessibility testing into continuous integration and continuous deployment pipelines has become increasingly common in modern development practices. This approach enables automated detection of accessibility issues during

the software development lifecycle, reducing the likelihood of regressions and improving overall compliance consistency. However, such automated integration still requires manual validation to ensure that usability aspects are adequately addressed.

Future Directions and Emerging Trends

The evolution of digital systems continues to reshape the requirements and expectations surrounding accessibility. As interfaces become increasingly adaptive, data-driven, and context-aware, accessibility standards are also expected to evolve beyond static rule-based frameworks. In this context, research and development efforts are progressively focusing on enhancing the adaptability, automation, and scalability of accessibility solutions.

One of the most significant emerging directions is the integration of artificial intelligence into accessibility workflows. Machine learning techniques are increasingly being used to automate tasks such as image caption generation, speech-to-text conversion, and content simplification. These capabilities are particularly relevant for users with visual and cognitive impairments, as they reduce dependency on manually authored accessibility metadata. However, the reliability and interpretability of such systems remain active research concerns, especially in high-stakes or safety-critical applications.

The use of natural language processing techniques has also contributed to improving accessibility in textual content. Models capable of summarization, paraphrasing, and readability optimization are being integrated into digital platforms to support users with different reading abilities. These systems aim to dynamically adjust content complexity based on user preferences or inferred cognitive load, thereby extending accessibility beyond static compliance rules.

Another important development area is adaptive user interface design. Unlike traditional static interfaces, adaptive systems modify their layout, behavior, or content presentation based on user context, device characteristics, or interaction history. This approach has strong implications for accessibility, particularly in mobile and cross-platform environments where screen size, input modality, and environmental conditions vary significantly. Adaptive interfaces have the potential to reduce cognitive and physical effort required for interaction, although they also introduce challenges related to predictability and user control.

The evolution of accessibility standards is also expected to continue through updates and expansions of existing frameworks. The Web Content Accessibility Guidelines (WCAG) 2.1 represent a transitional stage toward more comprehensive models, with subsequent versions such as WCAG 2.2 and ongoing discussions around WCAG 3.0 reflecting a shift toward more outcome-based and flexible evaluation approaches. These developments indicate a movement away from strictly technical compliance criteria toward broader user experience–oriented assessments.

Cross-platform consistency remains another key focus area in accessibility research. Modern digital ecosystems require seamless interaction across web browsers, mobile applications, wearable devices, and emerging immersive environments such as augmented and virtual reality systems. Ensuring accessibility across such diverse platforms requires abstraction of interface logic and consistent application of accessibility principles, regardless of underlying technology.

At the same time, the increasing complexity of automated systems raises important ethical and practical considerations. AI-driven accessibility solutions may introduce unintended biases,

particularly if training data does not adequately represent diverse user populations. Additionally, over-reliance on automation may reduce the emphasis on manual evaluation and user-centered design practices, which remain essential for capturing nuanced accessibility issues.

The growing emphasis on personalization also plays a role in shaping future accessibility approaches. Systems that dynamically adjust content presentation, navigation structure, and interaction modalities based on individual user profiles represent a shift toward more individualized accessibility experiences. While this approach offers significant benefits, it also introduces challenges related to privacy, data governance, and system transparency.

References

[1] Yu, Jiadong & Bekerian, D. & Osback, Chelsee. (2024). Navigating the Digital Landscape: Challenges and Barriers to Effective Information Use on the Internet. Encyclopedia. 4. 1665-1680. 10.3390/encyclopedia4040109.

[2] Kocarslan, H., & Stoycheva, B. (2025). The Effect of Digital Literacy on Online Purchase Intention: The Mediating Role of Social Media Use. Journal of Theoretical and Applied Electronic Commerce Research, 20(4), 355. https://doi.org/10.3390/jtaer20040355.

[3] Verhoef, Peter & Broekhuizen, Thijs & Bart, Yakov & Bhattacharya, Abhi & Dong, John & Fabian, Nicolai & Haenlein, Michael. (2021). Digital transformation: A multidisciplinary reflection and research agenda. Journal of Business Research. 122. 10.1016/j.jbusres.2019.09.022.

[4] Shah, Hardik. (2023). Advancing Web Accessibility: A Guide to Transitioning Design Systems From WCAG 2.0 to WCAG 2.1. 233-245. 10.5121/csit.2023.132218.

[5] Vera-Amaro, Guillermo & Rojano-Cáceres, José. (2025). Towards accessible website design through artificial intelligence: A systematic literature review. Information and Software Technology. 186. 107821. 10.1016/j.infsof.2025.107821.

[6] Silva, L., Pimentel, B., Duarte, B., Escarpini, R., Sousa, L., Cruz, N., & Silva, R. (2025). Accessibility by Design: A Systematic Review of Inclusive E-Book Standards, Tools, and Practices. Sustainability, 17(24), 11173. https://doi.org/10.3390/su172411173

[7] Washington Chiriboga-Casanova, Nuria Medina-Medina, and Patricia Paderewski-Rodríguez. 2025. Accessible Web Design

for Older Adults: Challenges and Solutions. ACM Trans. Access. Comput. 18, 3, Article 14 (September 2025), 50 pages. https://doi.org/10.1145/3763243

[8] T. Chatziemmanouil and C. Katsanos, "Accessibility Academy: Interactive Learning of the WCAG 2.1 Web Accessibility Guidelines," 2024 IEEE Global Engineering Education Conference (EDUCON), Kos Island, Greece, 2024, pp. 1-7, doi: 10.1109/EDUCON60312.2024.10578915.

[9] N. M., P. Chawla and A. Rana, "A Practitioner's Approach to Assess the WCAG 2.0 Website Accessibility Challenges," 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates, 2019, pp. 958-966, doi: 10.1109/AICAI.2019.8701320.

[10] Y. Akgül, "The most violated WCAG 1.0 guidelines by the developers of university websites in Turkey," 2017 12th Iberian Conference on Information Systems and Technologies (CISTI), Lisbon, Portugal, 2017, pp. 1-7, doi: 10.23919/CISTI.2017.7976007.

[11] A. P. Almenara, J. H. Elich and T. G. Saltiveri, "Collaborative wiki with accessible and non-accessible examples of WCAG guidelines," 2023 18th Iberian Conference on Information Systems and Technologies (CISTI), Aveiro, Portugal, 2023, pp. 1-6, doi: 10.23919/CISTI58278.2023.10211515.

[12] P. Thammachokmongkol, W. Lapawong and W. Rueangsirarak, "Evaluating Online Content Accessibility Using WCAG 2.1: A Case Study of University Platforms and an e-Learning System for Students with Visual Impairments," 2025 9th International Conference on Information Technology (InCIT), Phuket, Thailand, 2025, pp. 578-583, doi: 10.1109/InCIT66780.2025.11276028.

[13] R. S. Germano and I. Frango Silveira, "WCAG-Easy Tool : A tool based in the WCAG to learn web accessibility," 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), Madrid, Spain, 2022, pp. 1-6, doi: 10.23919/CISTI54924.2022.9820012.

[14] David F. Beer, "Web Accessibility for People with Disabilities: An Introduction for Web Developers," in Writing and Speaking in the Technology Professions: A Practical Guide , IEEE, 2003, pp.484-492, doi: 10.1002/9781119134633.ch76.

[15] Shah, Hardik. (2023). Harnessing Web Accessibility Tools for WCAG 2.1 Migration of a Design System. 54-60. 10.1109/CoNTESA61248.2023.10384953.

[16] S. K. Oswal and H. K. Oswal, "Examining the Accessibility of Generative AI Website Builder Tools for Blind and Low Vision Users: 21 Best Practices for Designers and Developers," 2024 IEEE International Professional Communication Conference (ProComm), Pittsburgh, PA, USA, 2024, pp. 121-128, doi: 10.1109/ProComm61427.2024.00030.

[17] Ordoñez-Briceño, K., Hilera, J. R., De-Marcos, L., & Saraguro-Bravo, R. (2025). Generating Accessible Webpages from Models. Computers, 14(6), 213. https://doi.org/10.3390/computers14060213

[18] Correa, M., Vitoriano, M. A., & Llanos, C. H. (2025). Web Accessibility in an Academic Management System in Brazil: Problems and Challenges for Attending People with Visual Impairments. Informatics, 12(3), 63. https://doi.org/10.3390/informatics12030063

[19] Sajek, D., Korotenko, O., & Kyrychok, T. (2025). Research on the Accessibility of Different Colour Schemes for Web

Resources for People with Colour Blindness. Journal of Imaging, 11(8), 268. https://doi.org/10.3390/jimaging11080268

[20] Andruccioli, M., Bassi, B., Delnevo, G., & Salomoni, P. (2025). Leveraging Large Language Models for Sustainable and Inclusive Web Accessibility. Big Data and Cognitive Computing, 9(10), 247. https://doi.org/10.3390/bdcc9100247

[21] Bogdanova, G., Todorov, T., Dochkova-Todorova, J., Noev, N., & Sabev, N. (2024). Design and Development Model of a Web Accessibility Ecosystem. Information, 15(10), 613. https://doi.org/10.3390/info15100613

CHAPTER 2

CONTENT MANAGEMENT SYSTEMS: ARCHITECTURE, TOOLS, SECURITY AND INFRASTRUCTURE PERSPECTIVES

AHMET TOPRAK[1]

Introduction

The management of digital content has become one of the central operational challenges of modern organizations. As the volume of information produced and consumed through digital channels continues to grow, the need for systematic approaches to creating, organizing, storing, and delivering content has intensified across virtually every sector [1].

A content management system (CMS) is a software platform that provides structured mechanisms for handling digital content throughout its lifecycle, from initial creation and editorial review to publication, versioning, and eventual archiving. Unlike simple file-based approaches, a CMS abstracts the underlying storage and delivery infrastructure, offering users and administrators a coherent

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

interface through which content can be managed without requiring direct manipulation of technical components.

Content management systems first emerged in the late 1990s as organizations began to recognize that maintaining a web presence required more than a one-time technical effort. The early systems were largely proprietary and monolithic, designed to serve specific content types within closed environments. As the web matured and the diversity of content formats expanded, demand grew for more flexible platforms that could accommodate different organizational structures, content workflows, and delivery targets.

Today, the CMS landscape is highly varied. It includes traditional web-focused platforms that tightly couple content management with front-end presentation, as well as headless architectures that separate the content repository from the delivery layer, enabling content to be consumed by multiple applications and devices simultaneously. Enterprise-grade systems offer advanced governance, versioning, and workflow capabilities, while lightweight open-source alternatives prioritize ease of deployment and community-driven extensibility [2].

The technical architecture of a CMS determines not only what kinds of content can be managed, but also how that content is stored, indexed, retrieved, and delivered. Core architectural decisions—such as the choice between relational and document-oriented databases, the design of the content model, and the approach to caching and scalability—have direct implications for system performance, maintainability, and operational cost.

Security represents another critical dimension of CMS design and operation. Content management systems frequently serve as the public-facing entry point for organizational data and services, making them attractive targets for a wide range of attacks. Common vulnerabilities include injection flaws, authentication weaknesses,

insecure file handling, and misconfigured access controls. The consequences of security failures can range from content defacement and data leakage to complete system compromise, with significant reputational and regulatory implications.

Infrastructure considerations are equally important. A CMS does not operate in isolation; it depends on a surrounding environment that includes web servers, database engines, caching layers, content delivery networks, and identity management services. The configuration and maintenance of this infrastructure directly affect the availability, performance, and security of the system.

This chapter examines content management systems from architectural, tooling, security, and infrastructure perspectives. It provides a structured analysis of CMS design patterns, reviews widely adopted platforms and their distinguishing characteristics, discusses the principal security challenges and mitigation strategies, and addresses the infrastructure requirements that underpin reliable CMS deployment.

Background and Evolution of Content Management Systems

The origins of content management systems are closely tied to the broader evolution of the World Wide Web. In the early years of the web, creating and maintaining online content required direct authorship of HTML files and manual deployment to web servers. This approach was technically feasible for small-scale sites but became impractical as organizations began to publish larger volumes of content and required multiple contributors with varying levels of technical expertise [3].

The first generation of CMS platforms emerged in the mid-to-late 1990s, initially as internal tools developed by large organizations to manage their own publishing workflows. These

early systems typically operated on server-side scripting technologies and relied on relational databases to store content separately from presentation logic. The separation of content from layout, though often partial and inconsistent by later standards, represented a significant conceptual advance over static file-based approaches.

The early 2000s saw the emergence of open-source CMS platforms that made structured content management accessible to a much wider audience. Systems such as Drupal and Joomla, followed shortly by WordPress, brought content management capabilities to individual developers and small organizations that could not afford commercial alternatives. The open-source model accelerated innovation through community contributions and established plugin and extension ecosystems that substantially extended core functionality.

As web applications grew more complex and organizations began to operate across multiple digital channels simultaneously, the limitations of tightly coupled CMS architectures became apparent. Systems that combined content storage, business logic, and presentation rendering within a single codebase were difficult to scale, adapt to new delivery channels, and integrate with external services. This tension gave rise to architectural experiments that sought to decouple the content repository from the front-end layer.

The concept of the headless CMS, which exposes content through application programming interfaces rather than generating rendered output directly, gained significant traction during the 2010s. This model aligned well with the proliferation of mobile applications, single-page web applications, and IoT devices, all of which required structured content but had their own rendering environments. Headless architectures offered greater flexibility at the cost of increased implementation complexity, since the front-end layer had to be developed and maintained independently.

More recently, the rise of cloud-native deployment models and software-as-a-service offerings has further transformed the CMS landscape. Managed CMS services remove much of the infrastructure burden from organizations, offering guaranteed availability, automatic updates, and integrated content delivery network capabilities. At the same time, concerns about vendor lock-in, data sovereignty, and customization limitations have led many organizations to maintain on-premises or hybrid deployments.

Enterprise CMS platforms developed distinct characteristics in parallel with these architectural shifts. Large organizations required capabilities beyond basic content publishing, including structured workflow management, role-based access control, version history, multi-site management, and integration with enterprise resource planning and customer relationship management systems. Platforms serving this market responded to these requirements but introduced significant complexity and total cost of ownership [4].

CMS Architecture: Patterns and Design Considerations

The architecture of a content management system encompasses the structural decisions that define how content is modeled, stored, managed, and delivered. Architectural choices made during system design have long-lasting implications for performance, scalability, flexibility, and maintainability. Understanding the principal architectural patterns is therefore essential for both system selection and custom development.

Content Modeling

Content modeling refers to the process of defining the types, structures, and relationships of content that a CMS will manage. A well-designed content model reflects the semantic distinctions meaningful to the organization and supports the retrieval and

presentation needs of its applications. Poor content modeling leads to rigid, difficult-to-maintain systems where content cannot be repurposed across channels or easily extended as requirements evolve.

Content types define the structure of individual content items through fields with specified data types and validation rules. A news article content type might include fields for title, body text, publication date, author reference, category taxonomy, and featured image. The granularity at which content is decomposed into structured fields directly affects reusability: more granular models support more flexible repurposing, while overly granular models increase editorial complexity.

Relationships between content types introduce additional complexity. A CMS must be capable of expressing references between items, such as an article referencing its author, or a product referencing a set of related categories. The approach to implementing these relationships affects how related content is retrieved and how referential integrity is maintained during updates and deletions.

Storage and Data Layer

The choice of storage technology is a fundamental architectural decision with implications for content flexibility, query performance, and scalability. Relational database management systems have historically been the dominant storage backend for CMS platforms, offering transactional integrity, well-understood query semantics, and mature tooling. Systems such as WordPress, Drupal, and Joomla rely primarily on relational databases, with table structures designed to represent content types, metadata, taxonomy relationships, and user data [5].

Document-oriented databases offer an alternative approach that aligns naturally with the variable structure of content items.

Since content fields can differ significantly between items even within the same content type, a document model allows schemas to be more flexible without requiring null columns or auxiliary tables. Headless CMS platforms and newer-generation systems have increasingly adopted document-oriented or hybrid storage models.

Content delivery performance is typically enhanced through caching mechanisms implemented at multiple levels. Object caches store frequently accessed database query results or rendered fragments in memory, reducing database load under high traffic conditions. Full-page caches store complete rendered responses for anonymous users, enabling the web server to bypass application logic entirely for cached requests. Cache invalidation—ensuring that stale cached content is replaced when source content changes—is one of the more technically challenging aspects of CMS infrastructure management [6].

Tools and Platforms

The CMS market encompasses a wide range of platforms serving different use cases, organizational scales, and technical requirements. Understanding the characteristics of major platforms helps organizations make informed selection decisions and provides context for evaluating architectural trade-offs in practice.

WordPress

WordPress is the most widely deployed CMS platform globally, powering a substantial share of all publicly accessible websites. Originally developed as a blogging platform in 2003, it has evolved into a general-purpose CMS through a combination of core development and an extensive ecosystem of themes and plugins. WordPress follows a coupled architecture in its default configuration, though headless deployments using the REST API or the WPGraphQL plugin have become increasingly common [5].

The core platform is built on PHP and relies on MySQL or MariaDB for data storage. Its plugin ecosystem, which encompasses tens of thousands of publicly available extensions, allows functionality to be added without modifying core code. This extensibility has been central to WordPress's adoption but also introduces significant security and performance risk when plugins are poorly maintained or incompatible with one another [7].

Drupal

Drupal is an open-source CMS platform with a stronger orientation toward complex content architectures and enterprise requirements. Its content modeling capabilities are more sophisticated than those of WordPress, supporting structured field definitions, reusable content components, and granular access control. Drupal's module system provides a lower-level extension API than WordPress plugins, enabling deep customization at the cost of a steeper learning curve [16].

Drupal 8 and subsequent versions represented a major architectural rewrite, adopting modern PHP practices and introducing decoupled architecture support as a first-class consideration. The platform is well-suited to organizations with complex editorial workflows, multilingual requirements, or regulatory compliance needs.

Headless and API-First Platforms

Dedicated headless CMS platforms have emerged as alternatives to extending traditional CMS platforms for decoupled deployments. These systems are built from the ground up around API-first delivery and typically offer schema-driven content modeling, role-based access management, and rich querying capabilities without the legacy constraints of older platforms [8].

Some platforms operate as software-as-a-service with strong tooling for content modeling and delivery API customization, while open-source alternatives can be self-hosted, offering greater control over data residency and customization. The choice between managed and self-hosted headless platforms involves trade-offs between operational convenience and control that are similar to those encountered in other cloud service decisions.

Enterprise CMS Platforms

Enterprise-grade platforms target large organizations with complex digital experience requirements. These systems typically offer deep integration with marketing automation, customer data platforms, and analytics services, as well as built-in multisite management and personalization capabilities. The total cost of ownership for enterprise CMS platforms is substantially higher than for open-source alternatives, encompassing licensing fees, implementation services, and ongoing support and customization [9].

Security Considerations in Content Management Systems

Content management systems represent attractive attack targets due to their prevalence, their role as public-facing applications, and the sensitivity of the data and access they control. Security vulnerabilities in CMS platforms can result in unauthorized content modification, data exfiltration, credential theft, malware distribution, and complete system compromise. Understanding the principal attack vectors and corresponding mitigations is essential for any organization operating a CMS in a networked environment.

Injection Vulnerabilities

SQL injection remains one of the most consequential vulnerability classes affecting CMS platforms and their extensions.

When user-supplied input is incorporated into database queries without adequate parameterization or escaping, an attacker can manipulate query structure to retrieve unauthorized data, modify records, or execute administrative database operations. Modern CMS platforms use parameterized queries and prepared statements in their core code, but third-party plugins and themes frequently introduce injection vulnerabilities through inadequate input handling.

Cross-site scripting (XSS) vulnerabilities arise when user-supplied content is rendered in web pages without proper encoding. An attacker who can inject arbitrary script content into a page served to other users can hijack sessions, redirect users to malicious sites, or extract sensitive information from the browser context. CMS platforms that allow rich text authoring or user-generated content must implement robust input sanitization and output encoding to prevent XSS exploitation.

Authentication and Access Control

Authentication weaknesses are a persistent source of CMS security incidents. Default or weak administrator credentials, the absence of multi-factor authentication, and insecure password reset mechanisms all create opportunities for unauthorized access. Credential stuffing attacks, in which credentials obtained from unrelated breaches are tested against CMS login endpoints, are particularly effective against systems where users share passwords across services [10].

Role-based access control is a foundational security mechanism in CMS platforms, determining which users or user groups can perform which operations on which content. Misconfigured access controls can grant excessive permissions, allowing authenticated users to access administrative functions, view unpublished content, or modify content outside their editorial scope.

Implementing the principle of least privilege—granting only the permissions necessary for a given role—reduces the impact of compromised accounts.

Plugin and Extension Security

The plugin and extension ecosystem that provides much of the functional value of open-source CMS platforms is also a principal source of security risk. Third-party extensions are developed with varying levels of security expertise and may receive infrequent security updates or be abandoned by their maintainers. A vulnerable plugin installed on a CMS instance can provide an attacker with a foothold that bypasses the security of the core platform entirely.

Organizations operating CMS platforms should maintain an inventory of installed extensions and monitor security advisories for each. Automated tools can assist in identifying outdated or vulnerable components, but organizational processes must ensure that updates are applied promptly and that unmaintained extensions are replaced or removed. The number of installed extensions should be minimized to reduce the overall attack surface.

File Upload and Media Handling

CMS platforms that allow authenticated or anonymous users to upload files introduce additional attack surface. Unrestricted file uploads can enable an attacker to upload executable scripts that are subsequently executed by the web server, a vulnerability known as remote code execution via unrestricted file upload. Mitigations include restricting allowed file types and extensions, storing uploaded files outside the web root or in a dedicated object storage service, and scanning uploads for malicious content.

Security Hardening Practices

Beyond addressing specific vulnerability classes, a comprehensive CMS security posture involves a set of hardening practices applied across the application and its infrastructure. These practices include disabling unnecessary features and default accounts, enforcing HTTPS for all communications, implementing HTTP security headers, restricting administrative interfaces to trusted IP ranges, and enabling detailed logging for security-relevant events [11].

Web application firewalls are frequently deployed in front of CMS applications to filter and block malicious requests based on pattern matching and behavioral analysis. While they provide a useful additional layer of defense, they should not be relied upon as the primary security control; they can be bypassed and require ongoing rule maintenance to remain effective. A defense-in-depth approach, combining application-level hardening with infrastructure-level controls, provides more robust protection.

Infrastructure Perspectives

A content management system's operational effectiveness depends not only on the quality of its application code but also on the infrastructure within which it is deployed. Infrastructure decisions affect availability, performance under load, disaster recovery capabilities, and the operational burden placed on technical staff. The appropriate infrastructure configuration varies considerably depending on the scale of the deployment, the traffic patterns of the application, and the organization's tolerance for operational complexity [12].

Web Server and Application Server Configuration

CMS applications are typically served through a web server that handles incoming HTTP connections and routes requests to the application backend. For PHP-based platforms such as WordPress and Drupal, the application executes within a PHP runtime

environment managed by a process manager. Server configuration decisions, including worker process counts, request timeouts, and connection limits, must be tuned to the expected request volume and response time characteristics of the application.

TLS configuration requires careful attention to ensure that encrypted connections are established securely and efficiently. Outdated TLS protocol versions and cipher suites must be disabled, and certificate management processes must ensure that certificates are renewed before expiry. Certificate management has been substantially simplified by automated certificate issuance services, though organizations must still ensure that automation is correctly configured and monitored.

Database Infrastructure

The database layer is typically the most performance-sensitive component of a CMS deployment. Under high read traffic, database query load can become a bottleneck, making query optimization, index design, and caching strategy critical concerns. Object caching systems reduce database load by storing query results and computed values in memory, serving repeated requests without database access.

Database replication configurations that maintain one or more read replicas alongside a primary write instance are commonly used to distribute read load across multiple nodes. Write operations are directed to the primary instance and replicated asynchronously to replicas, which serve read queries. This architecture improves read performance and provides failover capability if the primary instance becomes unavailable, though it introduces complexity in managing replication lag and failover procedures.

Content Delivery Networks

Content delivery networks distribute static assets and, in some configurations, cached dynamic content across geographically dispersed edge nodes, reducing latency for end users and decreasing load on origin infrastructure. For CMS deployments with a global or geographically distributed audience, CDN integration is a standard infrastructure practice. Beyond performance benefits, CDNs often provide DDoS mitigation capabilities by absorbing and filtering volumetric attack traffic before it reaches origin servers [13].

Cache purging—ensuring that edge nodes serve updated content promptly after changes are published in the CMS—is an important operational consideration for CDN-integrated deployments. Some CMS platforms provide native CDN integration that automates purge requests on content publication; others require custom integration or rely on TTL-based expiry, which may result in stale content being served for the duration of the TTL interval.

Containerization and Cloud Deployment

Container-based deployment has become increasingly common for CMS infrastructure. Containerization packages the application and its dependencies into portable, reproducible units that can be deployed consistently across development, staging, and production environments. Container orchestration platforms provide capabilities for automated scaling, self-healing, and rolling updates that improve operational resilience and reduce manual intervention requirements.

Cloud provider managed services reduce the operational overhead associated with CMS infrastructure by abstracting database management, caching infrastructure, and storage services behind managed APIs. Organizations can offload tasks such as database patching, backup management, and storage replication to cloud provider services, allowing technical teams to focus on application-level concerns. The trade-off involves dependency on

cloud provider service availability and pricing models that can vary significantly with usage volume.

Backup and Disaster Recovery

CMS deployments must be supported by backup and disaster recovery procedures that ensure content and configuration can be restored in the event of data loss, system failure, or security incident. A complete backup strategy must encompass the application database, uploaded media files, and any custom application code or configuration that is not stored in version control. Backup frequency should be determined by the acceptable recovery point objective—the maximum amount of content changes that the organization can afford to lose.

Backup integrity verification is a frequently overlooked aspect of backup procedures. Backups that have never been tested through a restoration exercise may prove unrestorable when needed due to corruption, incomplete coverage, or changes in system configuration. Regular restoration testing, including periodic full recovery exercises, should be incorporated into operational procedures to validate that recovery capabilities function as expected [14].

Future Directions and Emerging Developments

The content management system domain continues to evolve in response to changes in content consumption patterns, delivery technologies, and organizational requirements. Several developments are likely to shape the direction of CMS architecture, tooling, and infrastructure in the coming years.

The composable architecture paradigm, which assembles digital experience capabilities from best-of-breed specialized services rather than relying on a single monolithic platform, is

gaining traction in enterprise contexts. Under this model, a CMS serves as the content repository and editorial interface, while search, personalization, e-commerce, and analytics capabilities are provided by separate services integrated through APIs. This approach offers flexibility and the ability to replace individual components independently, but it increases integration complexity and the number of external service dependencies.

Artificial intelligence is being integrated into CMS platforms in several distinct ways. AI-assisted content creation tools help authors generate drafts, suggest edits, and produce metadata such as tags and summaries. Automated content classification and tagging reduces editorial effort for large content libraries. Personalization engines use machine learning to dynamically assemble and rank content for individual users based on behavioral signals. Each of these integrations introducess considerations around content quality, editorial oversight, and the accuracy and bias characteristics of the underlying models [15].

The relationship between content management and digital accessibility is receiving increasing attention. CMS platforms are beginning to incorporate accessibility checking directly into editorial workflows, alerting authors to potential compliance issues before content is published. Automated accessibility evaluation tools integrated into the CMS can detect a range of issues, though they must be complemented by human review for comprehensive coverage. This integration represents a shift from accessibility as a post-publication audit concern to an in-process quality dimension.

Edge computing represents another emerging influence on CMS architecture. Content delivery at the network edge enables dynamic content assembly and personalization to be performed with lower latency than centralized cloud deployments permit. CMS platforms are increasingly designed to generate or cache content at

the edge, blurring the traditional distinction between origin infrastructure and CDN.

Security challenges will continue to evolve alongside the CMS ecosystem. The growing use of third-party integrations, API connections, and supply chain dependencies introduces new attack vectors that extend beyond the CMS platform itself. As CMS platforms handle increasingly sensitive data and more complex organizational workflows, security must remain a central rather than peripheral consideration in system design, selection, and operation.

References

[1] S. Borse, A. Jamdade, Y. Joshi and S. Vani, "Smart Content Management System with Subscription Services," 2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG), Indore, India, 2024, pp. 1-5, doi: 10.1109/ICTBIG64922.2024.10911200.

[2] M. Nath and A. Arora, "Content management system : Comparative case study," 2010 IEEE International Conference on Software Engineering and Service Sciences, Beijing, China, 2010, pp. 624-627, doi: 10.1109/ICSESS.2010.5552271.

[3] Eve Astrid Andersson; Philip Greenspun; Andrew Grumet, "Content Management," in Software Engineering for Internet Applications , MIT Press, 2006, pp.97-139.

[4] Y. Tjong, "Successful measurement of Content Management System implementation," 2016 International Conference on Information Management and Technology (ICIMTech), Bandung, Indonesia, 2016, pp. 311-314, doi: 10.1109/ICIMTech.2016.7930351.

[5] A. Kumar, A. Kumar, H. Hashmi and S. A. Khan, "WordPress: A Multi-Functional Content Management System," 2021 10th International Conference on System Modeling & Advancement in Research Trends (SMART), MORADABAD, India, 2021, pp. 158-161, doi: 10.1109/SMART52563.2021.9675311.

[6] Shailesh Kumar Shivakumar, "Content Management System Architecture," in Enterprise Content and Search Management for Building Digital Platforms , IEEE, 2017, pp.104-153, doi: 10.1002/9781119206842.ch4.

[7] Y. Omar and N. S. @Ashaari, "Futuristic model for school's content management systems : A beginning," 2010

International Symposium on Information Technology, Kuala Lumpur, Malaysia, 2010, pp. 1387-1392, doi: 10.1109/ITSIM.2010.5561638.

[8] Dennis Priefer. 2014. Model-driven development of content management systems based on Joomla. In Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (ASE '14). Association for Computing Machinery, New York, NY, USA, 911–914. https://doi.org/10.1145/2642937.2653474

[9] S. Borse, A. Jamdade, Y. Joshi and S. Vani, "Smart Content Management System with Subscription Services," 2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG), Indore, India, 2024, pp. 1-5, doi: 10.1109/ICTBIG64922.2024.10911200.

[10] Drivas, I., Kouis, D., Kyriaki-Manessi, D., & Giannakopoulos, G. (2021). Content Management Systems Performance and Compliance Assessment Based on a Data-Driven Search Engine Optimization Methodology. Information, 12(7), 259. https://doi.org/10.3390/info12070259

[11] Martinez-Caro, J.-M., Aledo-Hernandez, A.-J., Guillen-Perez, A., Sanchez-Iborra, R., & Cano, M.-D. (2018). A Comparative Study of Web Content Management Systems. Information, 9(2), 27. https://doi.org/10.3390/info9020027

[12] Nasir, N. A., & Jeong, S.-H. (2021). Content Management Based on Content Popularity Ranking in Information-Centric Networks. Applied Sciences, 11(13), 6088. https://doi.org/10.3390/app11136088

[13] Sternad Zabukovšek, S., Jordan, S., & Bobek, S. (2023). Managing Document Management Systems' Life Cycle in Relation to an Organization's Maturity for Digital Transformation. Sustainability, 15(21), 15212. https://doi.org/10.3390/su152115212

[14] Martinez-Caro, Jose-Manuel & Aledo-Hernandez, Antonio-Jose & Guillen-Perez, Antonio & Sanchez-Iborra, Ramon & Cano, Maria-Dolores. (2018). A Comparative Study of Web Content Management Systems. Information. 9. 27. 10.3390/info9020027.

[15] Pauliková, A., Lestyánszka Škůrková, K., Kopilčáková, L., Zhelyazkova-Stoyanova, A., & Kirechev, D. (2021). Innovative Approaches to Model Visualization for Integrated Management Systems. Sustainability, 13(16), 8812. https://doi.org/10.3390/su13168812

[16] Collett, Paul. (2010). A Case for the Drupal Content Management System. The JALT CALL Journal. 6. 57-66. 10.29140/jaltcall.v6n1.92.

CHAPTER 3

Clothing Category Detection from RGB Images

Kübra İŞCAN [1]
İsmail BABAOĞLU [2]

1. Introduction

With the rapid advancement of technology, artificial intelligence and deep learning applications have led to profound changes across many sectors. The fashion industry has also been influenced by this transformation, developing innovative solutions such as clothing recognition, classification, and personalized recommendation systems through computer vision techniques. No longer limited to aesthetic concerns, the fashion industry has become one of the leading fields of digitalization, incorporating automated product management and user-centered shopping experiences.

Clothing classification problems require accurate and efficient processing of visual data. In particular, correctly identifying both the category and the color of a garment is critically important for many applications. For example, in e-commerce platforms, color and category information are fundamental components in product

[1] Konya Teknik Üniversitesi, Bilgisayar Mühendisliği, Orcid: 0009-0000-8857-1059
[2] Prof. Dr., Konya Teknik Üniversitesi, Bilgisayar Mühendisliği, Orcid: 0000-0002-2503-1482

filtering, inventory management, and personalized recommendation systems. However, challenges such as complex backgrounds, varying lighting conditions, and garments appearing in different positions in real-world images are factors that directly affect model performance.

In this study, a deep learning–based multi-output classification model was developed. The model is capable of simultaneously predicting both the category and the color information from clothing images. For accurate color detection, the clothing region was successfully identified and cropped using the available bounding box coordinates in the dataset, followed by dominant color extraction. During the model development process, the ResNet50 and EfficientNet architectures were compared, and the EfficientNet model, which demonstrated superior performance, was preferred. All experiments were conducted using the DeepFashion dataset, which contains comprehensive and realistic images.

This study has the potential to provide effective and practical solutions to image processing problems in the fashion industry, while also aiming to contribute to research in this field.

A review of the literature reveals that there are various studies on clothing detection from images.

In this study, Üzüm (2021) proposed a deep learning–based segmentation and classification system to detect clothing items in a photograph taken with a smartphone. The system consists of two main stages. In the segmentation stage, clothing regions in the image are identified and labeled for classification. At this stage, the highest accuracy was achieved using the Mask R-CNN model on the DeepFashion2 dataset. In the classification stage, the labeled images are trained using Convolutional Neural Networks (CNNs), and the category to which the garment belongs is determined. In this stage,

the most successful results were obtained with the Xception architecture.

In this study, Kınlı and Kıraç (2020) addressed the clothing classification problem by using Capsule Networks to overcome certain limitations of traditional Convolutional Neural Networks (CNNs), such as the loss of spatial information and sensitivity to affine transformations. They proposed FashionCapsNet, a four-layer Capsule Network architecture that employs a dynamic routing algorithm, and trained it on the DeepFashion dataset, which contains 290,000 clothing images across 46 categories.As a result of the experiments, FashionCapsNet achieved 83.81% top-3 accuracy and 89.83% top-5 accuracy. These results outperform methods that do not incorporate pose information and demonstrate performance comparable to baseline methods that utilize pose information. Additionally, it was suggested that future advancements in capsule network technologies could further improve performance.

In this study, Xu, J. et al. (2022) compared traditional machine learning and deep learning models using the Fashion-MNIST dataset. Fashion-MNIST is a dataset provided by Zalando Research, consisting of 70,000 grayscale images of size 28×28 pixels across 10 different clothing categories. The dataset is divided into 60,000 training images and 10,000 test images. This study categorizes clothing images used in e-commerce into two groups: “plain clothing images” and “worn clothing images,” and compares classification models suitable for small and medium-sized enterprises. For plain clothing images, the HOG+SVM method achieved the best result with 91.32% accuracy, while for worn clothing images, the deep learning–based Small VGG network stood out with an accuracy of 69.78%. For users with limited computational resources, HOG+SVM is recommended for plain images, whereas lightweight CNN models such as Small VGG are suggested for worn clothing images.

In this study, Schindler et al. (2018) addressed the classification of fashion and clothing products using deep Convolutional Neural Networks (CNNs). Five different CNN architectures were analyzed using cleaned and pre-trained models, and they were evaluated on tasks such as product and gender classification. The deep learning architectures used include CNNs, which are widely preferred for feature extraction and classification of visual information, achieving over 90% accuracy on datasets such as Fashion-MNIST. In addition, Vision Transformers (ViT) were employed for comprehensive visual context learning, reaching an accuracy of 95.25% on Fashion-MNIST. For transfer learning and training on large datasets, the Inception V3 architecture was used, achieving approximately 70% accuracy on around 80,000 fashion images. The study utilized three different datasets: Fashion-MNIST, the iMaterialist Fashion Attribute Dataset, and the Fashion Product Dataset.

In this study, Birim (2022) examined various CNN architectures and regularization techniques on 10 different fashion product classes in the Fashion-MNIST dataset. The study compared a baseline CNN model, a CNN with L2 regularization, a CNN with Dropout regularization, and a CNN model that combines both L2 and Dropout regularization. The results showed that the CNN model using Dropout regularization achieved the highest performance with an accuracy of 94.3%. Although the models using L2 regularization and the combination of both techniques also produced successful results, the best performance was obtained by the model incorporating Dropout.

In this study, Lao and Jagadeesh (2015) used deep Convolutional Neural Networks (CNNs) to address image processing problems in the fashion domain. The aim was to automatically classify clothing types and visual attributes (such as color, pattern, and collar type), as well as to detect the locations of

garments within images. A multi-label classification approach was adopted, and the model was trained using sigmoid activation and binary cross-entropy loss independently for each attribute. For clothing detection, region-based CNN methods were applied. Experiments conducted on a large and diverse dataset, Apparel Classification with Style (ACS), achieved 74.5% accuracy in clothing type classification and demonstrated that visual attributes can also be predicted with high accuracy. In addition, the model performed successfully in clothing search and detection tasks. This study shows that deep learning is an effective method for analyzing fashion images and highlights that multi-task learning improves performance, making a significant contribution to the field of fashion analytics.

In this paper, Cychnerski et al. (2017) developed a computer vision system for the accurate detection and classification of clothing items in e-commerce images. Various experiments were conducted using popular CNN architectures such as Residual Networks, SqueezeNet, and SSD. Clothing detection was trained and tested on the DeepFashion dataset, while classification was performed using dress images collected from online stores, focusing on five attributes: color, pattern, sleeve type, collar type, and hem type. Automatic labeling was carried out using metadata from online stores, achieving approximately 83% accuracy. The study also investigated the effects of data augmentation, network scaling, and ensemble methods on classification performance, analyzing the trade-off between accuracy and computational efficiency. As a result, high success rates were achieved in both clothing detection and classification tasks, and the most effective network configurations were presented.

2. Materials and Methods

2.1. K-Means

K-Means is a commonly used clustering algorithm in machine learning. It divides data points into a predefined number of k clusters. The algorithm works by assigning each data point to the nearest cluster centroid, and these centroids are updated iteratively. This process continues until the within-cluster variance is minimized. In the field of image processing, K-Means is frequently used for detecting dominant colors and segmentation by grouping color pixels. In each iteration, pixels are assigned to their nearest color centroid, and the centroids are updated by taking the mean of the newly assigned pixels, allowing color clusters to become more distinct (Jain, 2010).

2.2. EfficientNet

EfficientNet is a convolutional neural network architecture developed by Google Brain in 2019 that combines high accuracy with computational efficiency. In traditional CNN models, the model size is typically increased by scaling depth, width, and input resolution separately. However, EfficientNet distinguishes itself by using a compound scaling method that balances and systematically scales these three dimensions together. In this way, the model achieves better performance while avoiding unnecessary computational cost.

The basic building block of EfficientNet is the Mobile Inverted Bottleneck Convolution (MBConv) layer. Inspired by previous lightweight models, this structure provides an architecture that is both deep and computationally efficient. The EfficientNet family includes different model sizes ranging from B0 to B7; B0 is the smallest and most lightweight model, while B7 is the largest and the best-performing model. EfficientNet has achieved higher

accuracy than many previous architectures on large datasets such as ImageNet.

Thanks to these characteristics, EfficientNet is a powerful architecture preferred for applications that require high performance even under limited hardware resources. In addition, it is widely used in transfer learning and various computer vision tasks (Tan and Le, 2019).

2.3. ResNet50

ResNet50 is a 50-layer version of the Residual Network (ResNet) architecture, which holds an important place among deep learning models. As the number of layers in deep neural networks increases, issues such as vanishing gradients and performance degradation that occur during training can limit the model's learning capacity. In this context, the ResNet architecture was developed as an innovative solution to overcome these challenges.

The main contribution of the model is preventing information loss while increasing network depth through residual connections. Thanks to these connections, the input of a given layer is directly added to its output, which facilitates information flow during forward propagation and ensures that gradients are transmitted without degradation during backpropagation. This mechanism enables even very deep networks to be trained successfully.

In the ResNet50 architecture, special building blocks called bottlenecks are used to improve performance and efficiency. Each bottleneck block consists of three convolutional layers: a 1×1 dimensionality reduction layer, a 3×3 convolution layer, and a 1×1 dimensionality expansion layer. This structure reduces the computational load of the model while preserving depth in information flow. ResNet50 contains approximately 25 million parameters and is typically trained using input images of size

224×224 pixels. The model has proven its effectiveness by achieving high accuracy on large-scale datasets such as ImageNet.

Today, ResNet50 is widely used not only in image classification tasks but also in various computer vision applications such as object detection, image segmentation, and medical image analysis. In addition, thanks to its suitability for transfer learning methods, it can provide successful solutions for problems with limited data (He et al., 2016).

3. Application

3.1. Dataset

DeepFashion is a large-scale visual dataset widely used in the field of fashion and clothing recognition. It contains over 800,000 clothing images and offers rich diversity in terms of categories, parts, colors, and styles. The dataset provides detailed annotations for tasks such as clothing classification, category detection, clothing segmentation, and similar product retrieval. It serves as a standard benchmark for solving fashion-related computer vision problems in both research and industrial applications.

The Deepfashion sub-dataset used: Category and Attribute Prediction Benchmark,

- Contains approximately 289,000 images.
- The clothes include category (t-shirt, pants, etc.) and detailed attribute tags (color, pattern, sleeve type, etc.).

3.2. Bounding Box-Based Region Cropping and K-Means (Clustering-Based Color Extraction)

First, color detection was performed without cropping the images in the dataset. During model training, the K-Means algorithm was used to determine the color distribution of pixels in the clothing images. K-Means divides the pixels into several color clusters and

defines the mean of the most frequently occurring cluster as the dominant color. However, due to certain factors such as background color and the presence of multiple colors in the image, the model was not successful in accurately predicting the color.

Secondly, to improve the color prediction accuracy, the images were cropped using bounding box coordinates available in the dataset. Then, the K-Means algorithm was applied to determine the color distribution of pixels in the cropped clothing images. Since the influencing factors were reduced, the accuracy of color prediction increased.

A multi-label dataset containing both color and category information of the cropped clothing images was created.

3.3. Model Training and System

The dataset was split into 70% for training, 15% for validation, and 15% for testing.

3.3.1. EfficientNet Results

First, training was performed using the EfficientNet model. EfficientNet is an advanced deep learning model that provides high accuracy and efficiency while using less computational power through its scaling method. The classification report obtained as a result of the training is presented in Table 1.

Table 1. Final EfficientNet Classification Report Obtained in the Last Stage

Class	Precision	Recall	F1-Score	Total Sample Size
Anorak	0.62	0.40	0.49	25
Blazer	0.65	0.57	0.61	1094
Blouse	0.60	0.68	0.64	3603
Bomber	1.00	0.03	0.05	37

Button-Down	0.00	0.00	0.00	50
Cardigan	0.65	0.58	0.62	2084
Flannel	0.33	0.33	0.33	40
Halter	0.00	0.00	0.00	5
Henley	0.51	0.36	0.42	122
Hoodie	0.64	0.56	0.60	622
Jacket	0.58	0.62	0.60	1524
Jersey	0.71	0.09	0.16	115
Parka	0.44	0.36	0.39	107
Peacoat	0.89	0.36	0.52	22
Poncho	0.38	0.27	0.32	103
Sweater	0.61	0.65	0.63	1915
Tank	0.69	0.62	0.66	2318
Tee	0.71	0.77	0.74	5597
Top	0.32	0.22	0.26	1555
Turtleneck	0.00	0.00	0.00	23
Capris	0.00	0.00	0.00	8
Chinos	0.56	0.44	0.49	90
Culottes	0.48	0.31	0.38	81
Cutoffs	0.39	0.25	0.30	249
Gauchos	0.00	0.00	0.00	6
Jeans	0.80	0.89	0.84	1056
Jeggings	0.42	0.10	0.16	83
Jodhpurs	0.00	0.00	0.00	7
Joggers	0.61	0.54	0.57	651
Leggings	0.74	0.85	0.79	740
Sarong	0.00	0.00	0.00	6
Shorts	0.81	0.83	0.82	2949
Skirt	0.85	0.94	0.89	2199

Sweatpants	0.56	0.54	0.55	468
Sweatshorts	0.50	0.08	0.14	174
Trunks	0.74	0.30	0.43	46
Caftan	0.00	0.00	0.00	9
Coat	0.58	0.71	0.64	319
Coverup	0.00	0.00	0.00	2
Dress	0.91	0.97	0.94	10780
Jumpsuit	0.89	0.65	0.75	1002
Kaftan	0.45	0.25	0.32	20
Kimono	0.62	0.54	0.58	335
Onesie	1.00	0.33	0.50	12
Robe	0.50	0.17	0.25	24
Romper	0.77	0.68	0.72	1115

The model's performance varies significantly across classes. High F1 scores are achieved in categories with a large number of samples and visually distinctive features (especially Dress, Skirt, Shorts, Jeans, Leggings, Tee, and Jumpsuit/Romper), whereas performance drops considerably in categories with fewer samples or a higher likelihood of confusion with similar classes (Blazer, Blouse, Cardigan, Hoodie, Jacket, Sweater, and Tank). An F1 score of zero in some classes indicates that the model failed to learn these categories at all. Additionally, in some cases, high precision but very low recall values show that the model only predicts samples it is highly confident about, while largely missing instances of those classes. Overall, the results demonstrate that data imbalance and inter-class similarity negatively affect the model's performance.

The color classification report obtained as a result of the training is presented in Table 2.

Table 2. EfficientNet Color Classification Report

Class	Precision	Recall	F1-Score	Total Sample Size
Siyah	0.91	0.93	0.92	8577
Beyaz	0.90	0.90	0.90	12872
Kırmızı	0.70	0.76	0.73	277
Yeşil	0.73	0.59	0.65	104
Mavi	0.41	0.97	0.58	75
Sarı	0.81	0.61	0.70	36
Turuncu	0.58	0.85	0.69	234
Pembe	0.82	0.84	0.83	9690
Gri	0.85	0.74	0.79	7735
Kahverengi	0.71	0.77	0.74	3390
Mor	0.69	0.61	0.64	392

These results show that the model performs generally well in color classification, but still contains some imbalances. In particular, colors with a high number of samples such as Black, White, Pink, and Gray achieve high precision and recall values, indicating that the model has successfully learned these classes. For colors with a moderate number of samples, such as Brown, Red, and Orange, the performance is at an acceptable level. On the other hand, imbalanced results are observed in classes with fewer samples, such as Blue and Green. Especially for Blue, the very high recall but low precision indicates that the model tends to overpredict this color while making frequent mistakes. Overall, the results demonstrate that dataset balance and class distribution have a significant impact on model performance.

3.3.2. Resnet50 Results

Secondly, training was performed using the ResNet50 model. ResNet50 is a powerful deep learning model that provides high accuracy and facilitates learning through its deep layer structure and

residual connections. The classification report obtained as a result of the training is presented in Table 3.

Table 3. Final ResNet50 Classification Report Obtained in the Last Stage

Class	Precision	Recall	F1-Score	Total Sample Size
Anorak	1.00	0.04	0.08	25
Blazer	0.65	0.55	0.59	1094
Blouse	0.58	0.66	0.62	3603
Bomber	0.18	0.12	0.14	50
Button-Down	0.66	0.51	0.57	2084
Cardigan	0.32	0.47	0.38	40
Flannel	0.00	0.00	0.00	5
Halter	0.45	0.31	0.37	122
Henley	0.61	0.54	0.57	622
Hoodie	0.57	0.61	0.59	1524
Jacket	0.69	0.08	0.14	115
Jersey	0.57	0.15	0.24	107
Parka	0.57	0.36	0.44	22
Peacoat	0.44	0.15	0.22	103
Poncho	0.59	0.59	0.59	1915
Sweater	0.66	0.57	0.61	2318
Tank	0.64	0.80	0.71	5597
Tee	0.32	0.23	0.27	1555
Top	0.00	0.00	0.00	23
Turtleneck	0.00	0.00	0.00	8
Capris	0.49	0.23	0.32	90
Chinos	0.51	0.26	0.34	81
Culottes	0.42	0.29	0.34	249

Cutoffs	0.00	0.00	0.00	6
Gauchos	0.80	0.87	0.84	1056
Jeans	0.50	0.07	0.13	83
Jeggings	0.00	0.00	0.00	7
Jodhpurs	0.55	0.53	0.54	651
Joggers	0.75	0.81	0.78	740
Leggings	0.00	0.00	0.00	6
Sarong	0.80	0.82	0.81	2949
Shorts	0.84	0.91	0.87	2199
Skirt	0.47	0.58	0.52	468
Sweatpants	0.52	0.09	0.15	174
Sweatshorts	0.56	0.39	0.46	46
Trunks	0.00	0.00	0.00	9
Caftan	0.66	0.53	0.59	319
Coat	0.00	0.00	0.00	2
Coverup	0.92	0.94	0.93	10780
Dress	0.82	0.68	0.74	1002
Jumpsuit	0.17	0.25	0.20	20
Kaftan	0.57	0.54	0.55	335
Kimono	1.00	0.17	0.29	12
Onesie	020	0.04	0.07	24
Robe	0.76	0.67	0.71	1115
Romper	1.00	0.04	0.08	25

These results show that although the model performs well in some classes, it generally exhibits an inconsistent and imbalanced structure. In categories with a high number of samples such as Dress, Shorts, Skirt, Joggers, and Sarong, the F1 scores are high, indicating

that the model has successfully learned these classes. In classes with a moderate amount of data, such as Blazer, Blouse, Hoodie, and Sweater, the performance is at an acceptable level, although some misclassifications are present. On the other hand, in many classes (such as Flannel, Top, Turtleneck, Cutoffs, Jeggings, and Coat), an F1 score of zero indicates that the model failed to learn these categories entirely. Additionally, in some classes (for example Anorak, Romper, and Kimono), very high precision but extremely low recall values show that the model only correctly predicts a very small number of instances while missing most samples of these classes. Overall, this table demonstrates that data imbalance, low sample size, and inter-class similarity significantly and negatively affect model performance.

The color classification report obtained as a result of the training is presented in Table 4.

Table 4. ResNet50 Color Classification Report

Class	Precision	Recall	F1-Score	Total Sample Size
Siyah	0.88	0.94	0.91	8577
Beyaz	0.88	0.88	0.88	12872
Kırmızı	0.84	0.67	0.75	277
Yeşil	0.83	0.43	0.57	104
Mavi	0.60	0.95	0.74	75
Sarı	0.86	0.33	0.48	36
Turuncu	0.79	0.62	0.69	234
Pembe	0.83	0.77	0.80	9690

Gri	0.76	0.82	0.79	7735
Kahverengi	0.77	0.66	0.71	3390
Mor	0.59	0.76	0.67	392

These results indicate that the model performs generally balanced and well in color classification. In particular, classes with a high number of samples such as Black, White, Pink, and Gray achieve high precision and recall values, showing that the model can reliably predict these colors. For medium-sized classes (Red, Brown, Orange, and Purple), the performance is generally acceptable, although some imbalances are observed. However, more noticeable issues appear in low-sample classes such as Blue, Green, and Yellow. For instance, the high recall but lower precision for Blue suggests that the model tends to overpredict this color, while the low recall values for Yellow and Green indicate that the model often misses these classes. Overall, the results demonstrate that the amount of data and class balance have a significant impact on model performance.

4. Conclusion and Recommendations

In this study, a multi-task deep learning architecture was developed to simultaneously perform both clothing category classification and color recognition on garment images. Two different convolutional neural network (CNN) architectures, EfficientNet-B0 and ResNet50, were used in the training and testing processes. The study was conducted using cropped clothing images

from the DeepFashion dataset and their corresponding multi-label information.

Both models were compared in terms of overall accuracy, precision, recall, and F1-score metrics. In clothing category classification, the EfficientNet model performed slightly better than ResNet50, achieving 74% accuracy (ResNet50 accuracy: 72%). EfficientNet also demonstrated a more balanced and stable performance overall, with a higher weighted F1-score (0.73). Both models achieved high performance in classes with a large number of samples (such as “Dress,” “Tee,” “Skirt,” and “Jeans”). However, performance dropped significantly in classes with fewer samples (such as “Caftan,” “Sarong,” and “Halter”). In color classification, EfficientNet achieved highly successful results with an accuracy of 85%. The model obtained particularly high precision and recall values for frequently occurring colors such as black, white, pink, and gray. Even for less common colors (such as blue, yellow, and orange), it maintained relatively strong overall performance. Compared to ResNet50, EfficientNet demonstrated higher accuracy and more consistent results in color classification (ResNet50 color accuracy: 83%).

In the subsequent stages of this study, it is recommended to apply several improvement and enhancement methods to increase model performance based on the obtained findings. Although the model shows high performance in some classes, significant performance losses are observed in classes with a low number of

samples (such as Halter, Sarong, and Caftan). This situation is a result of class imbalance in the dataset. Therefore, increasing the number of samples in underrepresented classes, expanding data collection processes to include these categories, or applying data augmentation techniques specific to these classes would be beneficial. In the current study, the EfficientNet-B0 architecture was used and generally satisfactory results were obtained. However, in future stages of the work, the use of different architectures (such as EfficientNet-B4, ConvNeXt, or Vision Transformer) may further improve classification performance, especially in distinguishing visually similar classes. These advanced architectures can provide an advantage in learning complex structural features, and thus may achieve higher accuracy in multi-class and fine-grained visual classification tasks.

References

Öztürk Birim Ş. (2022). Fashion Image Classification: Convolutional Neural Networks with Regularization Techniques. Turkish Informatics Foundation Journal of Computer Science and Engineering, 15(1), 66-76.

Cychnerski J., Brzeski A., Boguszewski A., Marmolowski M., & Trojanowicz, M. (2017). Clothes detection and classification using convolutional neural networks. In 2017 22nd IEEE international conference on emerging technologies and factory automation (ETFA) (pp. 1-8). IEEE.

He K., Zhang X., Ren S., & Sun J. (2016). Deep Residual Learning for Image Recognition. CVPR, 770–778.

Jain A. K. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters, 31(8), 651–666.

Kınlı F., & Kıraç F. (2020). FashionCapsNet: clothing classification with capsule networks. Bilişim Teknolojileri Dergisi, 13(1), 87-96.

Lao B., & Jagadeesh K. (2015). Convolutional neural networks for fashion classification and object detection. CCCV 2015 Comput. Vis, 546, 120-129.

Schindler A., Lidy T., Karner S., & Hecker M. (2018). Fashion and apparel classification using convolutional neural networks. arXiv preprint arXiv:1811.04374.

Tan M., & Le Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Proceedings of the 36th International Conference on Machine Learning (ICML), 6105–6114.

Üzüm E. (2021). Deep learning based image segmantation and classification for fashion detection on smartphones. In 2021 29th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.

Xu J., Wei Y., Wang A., Zhao H., & Lefloch D. (2022). Analysis of clothing image classification models: a comparison study between traditional machine learning and deep learning models. Fibres & Textiles in Eastern Europe, 30(5).

CHAPTER 4

Terrorist, Soldier and Human Image Classification System

Selahaddin İŞCAN [1]
Mustafa Servet KIRAN [2]

1. Introduction

Image classification is a machine learning and deep learning technique that enables objects or entities within an image to be categorized into specific classes. This technique is used in many different fields such as disease diagnosis in medicine, object detection in autonomous vehicles, and suspicious behavior analysis in security cameras. Especially in the field of security, image classification plays a critical role in the automatic detection of threats and rapid response to incidents.

Terrorism is one of the greatest threats faced by modern societies, endangering both individual safety and public order. Terrorist acts are generally carried out through planned attacks targeting human or military objectives. Therefore, it is of great

[1] Konya Teknik Üniversitesi, Bilgisayar Mühendisliği, Orcid: 0009-0000-9773-8973
[2] Prof. Dr., Konya Teknik Üniversitesi, Bilgisayar Mühendisliği, Orcid: 0000-0002-5896-7180

importance to detect potential threat elements in advance and to take security measures accordingly.

In this study, it is aimed to classify individuals in images as soldier, human, or terrorist. The main objective of the study is to help identify individuals who may have the potential to be terrorists in security cameras or similar surveillance systems. In this way, security measures can be taken before incidents occur, and public safety can be ensured more effectively.

When the literature on terrorist detection and image classification is examined, it is observed that studies directly classifying terrorists based on clothing, accessories, or appearance are limited. This study aims to make an important contribution both to this underexplored area in the literature and to the strengthening of security measures.

In the research carried out by Volkan et al. (2021), a system was designed to alert security personnel, enabling them to rapidly recognize the presence of dynamite on individuals or railway tracks in surveillance camera images and respond with suitable precautions. The dataset employed in the study was formed by collecting and organizing dynamite images obtained from the internet. To assess the effectiveness of the proposed model, tests were performed using images containing dynamite on people or railway tracks, resulting in a detection accuracy of 98.4% and a loss value of 0.024.

Sheldon and Mukul (2021) sought to identify the most suitable Convolutional Neural Network (CNN) architecture for addressing real-world product classification tasks and improving price comparison optimization. In their study, the VGG16, VGG19, and ResNet50 architectures were evaluated and compared in terms of classification accuracy, with all three models tested on the same image classification task. Based on the results obtained from the

comparison, ResNet50 was determined to be the most successful architecture.

Ghalleb and Amara (2020) introduced a new reference database consisting of real-life facial images of terrorists associated with attacks conducted around the world since 2013. In addition, the study proposed a baseline identification method utilizing HOG features together with an SVM classifier.

Gül Cihan et al. (2024) performed image classification through deep transfer learning by incorporating varying numbers of hidden layers with sigmoid, tanh, and ReLU activation functions into CNN-based VGG16 and ResNet50 architectures, and subsequently compared their performances. According to the evaluation results, the highest performance was obtained from the ResNet50-based model using sigmoid activation, which included two hidden layers containing 256 and 128 neurons, along with a softmax classification layer of 10 neurons, achieving a classification accuracy of 97.5%.

Gözde Sena et al. (2024) utilized the Oxford-17 dataset, which consists of 1,360 flower images categorized into 17 different classes. In this convolutional neural network-based study, the performances of the deep learning architectures AlexNet and MobileNetV2 were compared. The results showed classification accuracies of 93.1% for the AlexNet architecture and 93.9% for the MobileNetV2 architecture.

Suting et al. (2024) proposed automatic rice leaf disease detection techniques based on various deep learning classifiers (namely MobileNetV2, ResNet50, VGG16, and LeNet-5), and also developed an Android application to instantly identify possible rice diseases from uploaded rice leaf images captured by smartphones. The experimental findings indicate that the MobileNetV2 model achieves better performance than the other models with respect to classification accuracy, recall, and F1-score.

Abdalsalam et al. (2024) introduced a framework named BiGRU-SA, developed for the classification and prediction of terrorist groups by employing bidirectional recurrent units together with self-attention mechanisms. The proposed BiGRU-SA framework showed remarkable performance in identifying 36 terrorist organizations involved in terrorist attacks, attaining 98.68% accuracy, 96.06% precision, 96.83% sensitivity, 99.50% specificity, and a Matthews correlation coefficient of 97.50%.

Murat Erhan (2024) aimed to detect foxes that prey on animals such as chickens, geese, ducks, and turkeys in the coops of poultry farmers. The outputs of architectures such as DenseNet, MobileNet, ResNet50, VGG16, VGG19, Xception, and YOLOv8 were adapted to perform transfer learning for detecting the presence of foxes in poultry farms. The models were then trained, and their performances were compared in terms of evaluation metrics such as loss, accuracy, precision, and F1-score. The results show that the YOLOv8 architecture generally achieved the best performance.

In the study conducted by Impana and Chethana (2022), CCTV-based video image processing techniques were utilized for real-time monitoring of moving objects. The proposed system was designed to enhance child safety by continuously observing nearby children and identifying potential threats such as fire hazards or other dangerous situations at an early stage. A neural network-based approach was employed to analyze surveillance data and generate an audible warning signal whenever a risky condition was detected, enabling rapid awareness and response.

In the study presented by Gao et al. (2024), an anomaly detection framework was developed using surveillance video footage to identify criminal activities such as arson, theft, and vandalism. Multiple deep learning models were trained for both object detection and crime classification tasks in order to improve the recognition of suspicious events. The proposed approach was

designed to automatically analyze CCTV recordings and determine crime categories even in the absence of a human monitoring operator. Furthermore, the system was integrated with an SMS notification service to alert registered users about detected incidents, enabling faster awareness and response to potential threats.

In the study conducted by Shanthi and Manjula (2025), a deep learning-based intelligent surveillance system was developed for real-time crime detection in crowded environments. The proposed framework combined Faster R-CNN and Mask R-CNN structures within the FMR-CNN model and integrated YOLOv8 to improve detection speed and localization performance. Experimental results demonstrated that the system achieved high accuracy under different environmental conditions and provided an effective solution for security and crime prevention applications.

2. Materials and Methods

2.1. MobileNetV2

MobileNetV2 is a neural network architecture developed for deep learning-based image processing tasks and is particularly optimized to run efficiently on mobile and embedded devices. Due to its lightweight architecture and low computational requirements, it is extensively utilized in applications such as image classification, object detection, and segmentation, providing high accuracy even on devices with constrained hardware resources. MobileNetV2 mainly reduces both the number of parameters and computational load by using depthwise separable convolutions and bottleneck blocks, while keeping accuracy loss to a minimum. It has been successfully used in applications such as face recognition, medical diagnosis, object detection in autonomous vehicles, and augmented reality. The model stands out for its high performance even on low-resolution images and can become even more effective when used with transfer learning techniques on limited datasets (Nitika, 2024).

2.2. ResNet50

ResNet50 is a deep learning architecture designed to overcome the vanishing gradient problem that arises during the training of deep neural networks. It is commonly employed in computer vision applications, including image classification, object detection, and segmentation. The main principle of ResNet50 is to use skip connections to pass information directly between layers. These connections add the output of a previous layer to a later layer, allowing gradient information to propagate without disappearing even in very deep networks.

ResNet50 features a deep network structure composed of 50 layers and includes elements such as convolutional layers, pooling layers, and fully connected layers. Thanks to residual blocks, training becomes more stable even as deeper layers are added to the model. This architecture has been trained on large-scale datasets like ImageNet and is recognized for its high levels of accuracy.

ResNet50 is effectively used for classifying complex objects in high-resolution images, medical image analysis (for example, tumor detection), satellite image analysis, and in security applications such as face recognition and threat detection. It provides high accuracy, especially when working with large datasets, and can also achieve successful results on smaller datasets when used with transfer learning techniques (Süheda, 2021).

2.3. YOLOv8

YOLOv8 (You Only Look Once version 8) is an advanced deep learning model designed for real-time object detection and image analysis tasks. As an updated version of the YOLO architecture, YOLOv8 provides notable enhancements compared to earlier versions in both processing speed and detection accuracy. It is commonly applied in computer vision tasks such as object detection, image segmentation, and object tracking.

YOLOv8 processes an image in a single pass (as a single-stage detector) and simultaneously predicts the locations and classes of objects. This architecture uses an anchor-free structure, requiring less computation and enabling more precise detections through direct regression of bounding boxes. In addition, the depth and width of the model have flexible configurations that can be optimized for different tasks.

YOLOv8 is applied in many different areas, including detecting pedestrians and vehicles in autonomous driving systems, identifying suspicious objects and individuals in security systems, monitoring crops in agriculture, detecting abnormal cells in medical image analysis, and performing quality control in manufacturing production lines. Success Factors: YOLOv8 is highly successful in real-time object detection on both high-resolution and low-resolution images. It is particularly preferred in applications where the balance between speed and accuracy is critical (Meryem, 2023).

3. Application

3.1. Preparation of the Dataset

In this study, in order to develop a classification model based on the appearance of terrorist suspects, the need to first create a dataset emerged. As a result of the literature review, it was determined that there is no readily available and balanced dataset for the terrorist class. To address this limitation, manual data collection methods were considered; however, due to the large amount of data and the time-consuming and labor-intensive nature of the manual process, an automated data collection approach was required.

In this direction, a Python script was developed to automatically collect data from Google Images. The developed script enables downloading the desired number of images by performing searches on Google Images using specified keywords. For the terrorist class, various keywords such as “terrorist”, “armed

terrorist", and "masked terrorist" were used to collect data. At this stage, 480 terrorist images were obtained.

Then, the creation of the second class, the soldier dataset, was carried out. While constructing the soldier class dataset, data was collected both from Google Images using a similar Python script and from existing datasets obtained from the Kaggle platform, and these data were merged. By combining these two sources, a balanced dataset was constructed, consisting of a total of 518 soldier images.

Finally, the human class dataset was created. It was observed that ready-made and balanced datasets for the human class are more common and easily accessible. Therefore, the human dataset was compiled by utilizing existing datasets available on the Kaggle platform. As a result of this process, a total of 504 human images were obtained.

In this way, datasets for all three classes were created and made ready for training the classification model.

3.2. Model Training and System

Once the dataset had been prepared, the model training process began. The data was divided into 70% for training, 15% for validation, and 15% for testing.

3.2.1. MobileNetV2 Experiment and Results

First, training was performed using the MobileNetV2 model. MobileNetV2 is a deep learning model known for its low computational cost and fast execution. The confusion matrix obtained as a result of the training is presented in Table 1.

Table 1. Confusion Matrix for MobileNetV2

Real \ Prediction	Terrorist	Soldier	Human
Terrorist	2	2	70
Soldier	6	2	64
Human	0	0	67

- In the terrorist class, 2 correct predictions were made, while 2 were misclassified as soldier and 70 as human.
- In the soldier class, 2 correct predictions were made, while 6 were misclassified as terrorist and 64 as human.
- In the human class, 67 correct predictions were made, achieving a perfect classification performance.

The MobileNetV2 model showed a significant imbalance across classes and achieved low accuracy, particularly in the terrorist and soldier classes. Therefore, the model was considered unsuccessful.

3.2.2. ResNet50 Experiment and Results

After the poor performance of MobileNetV2, training was repeated on the same dataset using another deep learning model, ResNet50. Accordingly, the confusion matrix is presented in Table 2.

Table 2. Confusion Matrix for ResNet50

Real \ Prediction	Terrorist	Soldier	Human
Terrorist	52	20	4
Soldier	37	9	25
Human	0	0	79

- In the terrorist class, 52 correct predictions were made, while 20 were misclassified as soldier and 4 as human.
- In the soldier class, 9 correct predictions were made, while 37 were misclassified as terrorist and 25 as human.
- In the human class, 79 correct predictions were made, achieving a perfect classification performance.

ResNet50 showed a more successful classification performance compared to MobileNetV2. However, significant misclassifications were still observed in the soldier class. These results were considered to be caused by limitations in the dataset.

3.2.3. YOLOv8, Data Augmentation, and Results

First, human detection was performed on the dataset using the YOLOv8 algorithm. YOLOv8 is a modern deep learning-based model preferred for object detection due to its high accuracy. Human detection was successfully carried out for each category (terrorist, soldier, and human). In images containing multiple people, each detected person was cropped using YOLOv8, and the dataset was updated accordingly. In this way, individuals in the dataset were separated, and the data was divided into 70% for training, 15% for validation, and 15% for testing.

As a result of this process, the dataset sizes were updated as follows:

- Terrorist Images: 981
- Soldier Images: 834
- Human Images: 1554

Second, in order to achieve better model performance and to overcome problems caused by the limited dataset, data augmentation was applied using ImageDataGenerator. ImageDataGenerator artificially expands the dataset by performing operations such as rotation, zooming, horizontal/vertical flipping, and brightness adjustment.

Using this expanded dataset, the ResNet50 model was retrained, and more balanced and higher accuracy results were obtained compared to previous outcomes. The results are presented in Table 3 and Table 4.

Table 3. Classification Report in the Final State

Class	Precision	Recall	F1-Score	Total Sample Size
Terrorist	0.94	0.80	0.86	157
Soldier	0.83	0.93	0.88	129
Human	0.95	0.99	0.97	220

1. Class: Terrorist

- Precision = 0.94: The model correctly classified 94% of the instances predicted as “Terrorist” as actually terrorist. This indicates that the model is generally reliable when making a “Terrorist” prediction.

- Recall = 0.80: The model correctly identified 80% of all actual “Terrorist” instances. This means that 80% of all terrorist samples were successfully detected.
- F1-Score = 0.86: This is a balanced measure of precision and recall. An F1 score of 0.86 indicates a good balance between correctly identifying terrorist cases and minimizing misclassifications.
- Support = 157: A total of 157 samples belonging to the “Terrorist” class were evaluated.

2. Class: Soldier

- Precision = 0.83: The model correctly classified 83% of the instances predicted as “Soldier” as actually soldiers. This indicates that the model generally makes correct predictions when labeling a sample as “Soldier.”
- Recall = 0.93: The model correctly identified 93% of all actual “Soldier” instances. This means that the majority of soldier samples were correctly recognized.
- F1-Score = 0.88: For the soldier class, the F1 score of 0.88 shows a good balance between correctly identifying soldiers and minimizing misclassifications.
- Support = 129: A total of 129 samples belonging to the “Soldier” class were evaluated.

3. Class: Human

- Precision = 0.95: The model correctly classified 95% of the instances predicted as “Human” as actually humans. This shows that the model makes highly accurate predictions when labeling a sample as “Human.”

- Recall = 0.99: The model correctly identified 99% of all actual “Human” instances. This indicates that nearly all human samples were successfully detected.
- F1-Score = 0.97: For the human class, the F1 score of 0.97 demonstrates a very strong balance between precision and recall, indicating highly effective classification performance.
- Support = 220: A total of 220 samples belonging to the “Human” class were evaluated.

Table 4. Confusion Matrix in the Final State

Real \ Prediction	Terrorist	Soldier	Human
Terrorist	125	23	9
Soldier	7	120	2
Human	1	1	218

1. Terrorist Class:

- 125 images were correctly classified as “Terrorist.”
- 23 images were incorrectly assigned to the “Soldier” class.
- 9 images were incorrectly assigned to the “Human” class.

2. Soldier Class:

- 120 images were correctly classified as “Soldier.”
- 7 images were incorrectly assigned to the “Terrorist” class.
- 2 images were incorrectly assigned to the “Human” class.

3. Human Class:

- 218 images were correctly classified as “Human.”

- 1 image was incorrectly assigned to the “Terrorist” class.
- 1 image was incorrectly assigned to the “Soldier” class.

4. Conclusion and Recommendations

In this study, an image classification system including terrorist, soldier, and human classes was developed. In the first stage of the study, a dataset was created. The dataset was compiled using both a Python-based script to download images from Google Images and existing datasets available on the Kaggle platform.

MobileNetV2 and ResNet50 models were used to compare classification performance. While MobileNetV2 requires lower computational power due to its lightweight structure, ResNet50 provided higher accuracy thanks to its deeper architecture. Based on the initial results, YOLOv8 algorithm and data augmentation techniques were applied to improve classification performance. Human detection was performed on images using the YOLOv8 algorithm. YOLOv8 was able to accurately detect human figures in the dataset thanks to its strong precision and real-time detection performance. Following this, data augmentation techniques such as horizontal flipping, rotation, and brightness adjustment were applied, allowing the model to produce more generalizable and reliable predictions. As a result, the highest performance was achieved with the ResNet50 model.

By automatically distinguishing between terrorists, soldiers, and human from RGB images, the study demonstrates the potential of deep learning-based approaches to support real-time threat analysis and decision-making processes. The project was also inspired by recent terrorist attacks targeting critical institutions, highlighting the growing need for proactive security technologies capable of detecting potential threats before incidents occur. Therefore, the proposed approach has the potential to contribute not only to academic research but also to practical security applications

such as smart city surveillance systems, border security, and critical infrastructure protection.

In the future stages of the study, it is planned to further enrich the dataset and test different deep learning models. In particular, the use of models such as EfficientNet, VGG19, and Vision Transformer (ViT) has the potential to further improve classification accuracy. At the same time, it is aimed to use transfer learning techniques more effectively with larger and more balanced datasets. These processes are expected to make significant contributions both to the literature and to practical applications in the field of security.

References

Abdalsalam M., Li C., Dahou A. & Kryvinska N. (2024). Terrorism group prediction using feature combination and BiGRU with self-attention mechanism. PeerJ Computer Science, 10:e2252. https://doi.org/10.7717/peerj-cs.2252

Çimen, M. E. (2024). Comparison of Deep Learning and Yolov8 Models for Fox Detection Around the Henhouse. Journal of Smart Systems Research, 5(2), 76-90. https://doi.org/10.58769/joinssr.1498561

Gao, J., Shi, J., Balla, P., Sheshgiri, A., Zhang, B., Yu, H., & Yang, Y., 2024, Camera-Based Crime Behavior Detection and Classification. Smart Cities, 7(3), 1169-1198. https://doi.org/10.3390/smartcities7030050

Ghalleb, A. E. K., & Amara, N. E. B. (2020). A Benchmark Terrorist Face Recognition Database. 2020 International Conference on Cyberworlds (CW), IEEE, 285-288. https://doi.org/10.1109/cw49994.2020.00052

Habek, G. C., Tasdemir, S., Basciftci, F., & Yılmaz, A. (2024). The Effect of Activation Functions on Performance in Image Classification with Transfer Deep Learning Techniques.

Afyon Kocatepe University Journal of Science ann Engineering, 24(2), 294-307. https://doi.org/10.35414/akufemubid.1334098

Impana, H.C., Chethana, H.T., 2022, Video Classification and Safety System. Advances in Data and Information Sciences. Lecture Notes in Networks and Systems, https://doi.org/10.1007/978-981-16-5689-7_52

Karabay, G. S., Çavaş, M., & Avcı, E. (2024). Performance comparison of AlexNet and MobileNetV2 architectures in flower classification. Journal of the Faculty of Engineering and Architecture of Gazi University, 40(2), 829-836. https://doi.org/10.17341/gazimmfd.1463663

Kaya, V., Baran, A., & Tuncer, S. (2021). Deep Learning Based Security Support System For The Prevention Of Dynamite-Backed Terrorist Activities. European Journal of Science and Technology, (22), 81-85. https://doi.org/10.31590/ejosat.845467

Meryem, S. (2023). Discover the Power of YOLOv8: "The Next-Generation Object Detection Algorithm", Accessed in April 2026 from https://medium.com/@meryemmsakinn/yolov8in-g%C3%BCc%C3%BCn%C3%BC-ke%C5%9Ffedin-yeni-nesil-nesne-tespit-algoritmas%C4%B1-d98efcda3e8d

Nitika, S. (2024). What is MobileNetV2? Features, Architecture, Application and More, Accessed in April 2026 from https://www.analyticsvidhya.com/blog/2023/12/what-is-mobilenetv2/?utm_source=chatgpt.com

Shanthi P ve Manjula V., 2025, Weapon detection with FMR-CNN and YOLOv8 for enhanced crime prevention and security. Sci Rep 15, 26766. https://doi.org/10.1038/s41598-025-07782-0

Sheldon, M. & Mukul, A. (2021). A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification. Proceedings of the IEEE Central America and Panama Convention (CENTCON). doi:10.1109/CENTCON52345.2021.9687944.

Suting, A., Kumar, A., Halder, A., & Chanu, L. L. (2024). Rice Leaf Diseases Classification Using Deep Learning Algorithms for Smartphone Application Development: An Empirical Study. Içinde International Journal of Image and Graphics. World Scientific Pub Co Pte Ltd. https://doi.org/10.1142/s0219467826500300

Süheda, A. (2021). ResNet (Residual Network) Nedir?, Accessed in April 2026 from https://suhedacilek.medium.com/resnet-residual-network-nedir-49105e642566


BİDGE