iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

COUNTERFEIT LOGO DETECTION USING DEEP LEARNING

Vechalapu Yallisha Rani
MCA student, Department of Computer Science & System Engineering,
Andhra University College of Engineering, Visakhapatnam, AP

R. Shweta Balkrishna
Assistant Professor, Department of Computer Science & System Engineering, Andhra University College of Engineering, Visakhapatnam, AP.

Corresponding Author:
Vechalapu Yallisha Rani
ranivechalapu9@gmail.com

ABSTRACT:
This paper addresses the critical problem of counterfeit logo detection and, crucially, the precise identification of the counterfeit region within an image, leveraging advanced deep learning techniques. The widespread presence of counterfeit goods, often identifiable by subtle yet deliberate alterations to authentic brand logos, poses significant economic and safety threats. While previous research has focused on overall logo classification, pinpointing the exact counterfeit element is vital for forensic analysis and actionable intervention. Our proposed framework utilizes Convolutional Neural Networks (CNNs) for robust feature extraction from logo images. Specifically, we integrate the ResNet50 architecture, known for its ability to learn complex hierarchical representations through deep residual connections, thereby mitigating issues such as vanishing gradients during training. Beyond classification, this work extends to incorporating techniques for localization, enabling the model to highlight the specific pixels or regions indicative of counterfeiting. This dual approach of detection and localization, powered by CNNs and ResNet50, offers a comprehensive solution for safeguarding brand integrity, enhancing supply chain security, and providing crucial evidence for intellectual property enforcement.

Keywords:
Counterfeit detection, Logo Identification, Deep learning, Convolutional Neural Networks (CNNs), ResNet50, Brand Integrity Protection.

I. INTRODUCTION

The global economy grapples with the pervasive issue of **counterfeit goods**, inflicting substantial financial losses on legitimate businesses, eroding consumer trust, and posing significant safety hazards. From luxury fashion to critical pharmaceutical products, these deceptive items often mimic authentic branding, with **logo replication** being a primary method of deception. Such subtle yet deliberate alterations to genuine brand logos render manual detection challenging, time-consuming, and prone to human error, underscoring the urgent need for automated and robust solutions.

Traditional approaches to logo authentication typically rely on human visual inspection or basic image processing techniques, proving inadequate for discerning the intricate nuances that differentiate genuine logos from their high-fidelity counterfeit counterparts. The sheer volume of goods in global supply chains further exacerbates this problem, making exhaustive manual checks impractical and inefficient. While prior research in computer vision has made strides in general object recognition and image classification, the specific challenge of counterfeit logo detection demands specialized techniques capable of handling **fine-grained visual distinctions** and the subtle manipulations inherent in fraudulent branding.

Furthermore, merely classifying an image as containing a counterfeit logo is often insufficient for practical applications. For effective intervention, intellectual property enforcement, and forensic analysis, it is crucial not only to detect the presence of a counterfeit logo but also to precisely identify and **localize the specific regions or pixels** within the image that exhibit counterfeit characteristics. This capability allows for targeted analysis of the

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

deceptive elements, providing actionable insights for designers to refine logo security features and for law enforcement to build stronger cases against counterfeiters.

In response to these intertwined needs, this paper introduces a comprehensive **deep learning-based framework** designed for robust **counterfeit logo detection** and accurate **localization** of counterfeit components within an image. Our approach leverages the powerful capabilities of **Convolutional Neural Networks (CNNs)** for their unparalleled ability to extract hierarchical visual features from complex image data. To overcome limitations associated with training very deep networks, such as vanishing gradients, we specifically integrate the **ResNet50 architecture**. With its innovative residual connections, ResNet50 facilitates the training of profound models, enabling the capture of highly discriminative patterns essential for distinguishing authentic logos from their illicit imitations. By combining the strengths of CNNs for feature learning and ResNet50 for deep network training, our framework offers an automated, highly accurate, and scalable solution to combat the growing threat of counterfeit goods. This dual focus on both detection and precise localization represents a significant advancement in safeguarding brand integrity, enhancing supply chain security, and empowering intellectual property enforcement efforts in the digital age.

II. RELATED WORK:

The problem of counterfeit detection has long been a significant concern across various industries, especially in the domains of fashion, electronics, pharmaceuticals, and branded consumer goods. As a result, researchers have explored numerous computational techniques to automate the detection of counterfeit items, with a particular focus on visual inspection methods.

A. Traditional Approaches to Visual Forgery Detection

Early solutions primarily relied on classical image processing techniques, such as histogram comparison, edge detection, and template matching. These approaches were used to compare suspected counterfeit logos against authentic references by extracting low-level features like colour histograms and geometric patterns. Although computationally efficient, these methods were limited in handling variations in image resolution, background clutter, and lighting conditions. Moreover, handcrafted features often failed to capture complex distortions that modern counterfeiters introduce, making these techniques insufficient in real-world scenarios.

B. Machine Learning and Deep Learning Techniques

To overcome the limitations of traditional methods, researchers have increasingly turned to machine learning, particularly deep learning. Convolutional Neural Networks (CNNs) have emerged as the dominant paradigm for image classification and object detection due to their ability to learn hierarchical and spatially invariant features directly from pixel data. Several studies have demonstrated the effectiveness of CNNs in tasks such as face verification, signature authentication, and document forgery detection.

Recent works have applied deep neural networks for brand logo recognition and classification using benchmark datasets such as FlickrLogos-32 and WebLogo-2M. While these efforts have successfully improved logo localization and brand identification accuracy, most of them focus on identifying the presence of logos rather than assessing their authenticity. This highlights a critical gap in literature—namely, the lack of solutions tailored to identifying subtle modifications that distinguish counterfeit logos from genuine ones.

C. Explainability and Region-Based Analysis

A growing area of interest within counterfeit detection is the integration of explainable AI (XAI) techniques. Tools such as Grad-CAM, saliency maps, and difference imaging have been employed to enhance the interpretability of model decisions. These techniques help localize the regions within an image that most influence the prediction, allowing users to visually verify model outputs. However, many of these approaches still operate as post-processing steps rather than being embedded within a fully automated pipeline.

D. Practical Applications and Deployment Gaps

Despite advancements in algorithmic accuracy, relatively few studies have addressed the challenge of deploying counterfeit detection systems in real-world environments. Most existing research emphasizes model performance in controlled datasets but lacks a user-facing component or deployment mechanism. The integration of deep learning models into web applications remains underexplored, particularly in ways that allow interactive user engagement and feedback.

E. Contribution of the Present Work

This research builds upon the aforementioned foundations by developing a Flask-based web application that enables users to upload logo images and receive real-time authenticity classification. Unlike previous approaches, the proposed system not only predicts whether a logo is genuine or counterfeit using a pre-trained CNN, but also

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

provides visual feedback through localized difference mapping. This combination of classification and interpretability addresses the dual need for accuracy and transparency, filling a critical gap in existing literature.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

Overview of the System

The system presented in this study is a web-based platform acclimatized for relating fake ensigns using deep learning ways. By combining stoner-friendly design with an intelligent model and explain ability features, the operation delivers real-time perceptivity into totem authenticity. The following section outlines the overall system design and specialized perpetration in an intertwined format.

This operation is intended to allow druggies to submit totem images through a cybersurfed interface. After submission, the system processes the image, classifies it as authentic or fake using a trained convolutional neural network (CNN), and in the case of a fake vaticination, provides a visual highlight of suspicious regions. This explanation medium is essential for erecting trust in AI-supported opinions.

The platform is organized into several core factors:

- **Stoner Interface (Frontend):** Developed with standard web technologies, this element enables image uploads and presents results in a clear, visual format.
- **Operation Sense (Backend Garçon):** Erected using Flask, this part coordinates the processing channel, handles image medication, model conclusion, and communicates with the frontend.
- **Convolutional Neural Network Model:** A pre-trained and customized CNN used to classify ensigns.
- **Visualization and Pre-processing Module:** Responsible for conforming image confines and generating interpretability visualizations.
- **Data Storage (Optional):** Can be used for storing logs, reference ensigns, or stoner inputs.

The stoner experience flows as follows:

1. The stoner submits a totem image.
2. The garçon processes the image.
3. The deep literacy model evaluates the image.
4. If the image is supposed fake, the system highlights affected regions.
5. The frontend displays the final decision and visual explanation.

Dataset and Pre-processing Strategy

The training dataset comprises authentic and forged ensigns. Authentic images were sourced from brand accoutrements and public datasets, while forged performances were created by manually altering authentic ensigns or collecting exemplifications from online sources.

- **Total Images:** Around 3,000 exemplifications, resolve unevenly between authentic and fake.
- **Brand Diversity:** Includes over 30 different totem designs.
- **Formats:** Common formats like JPEG and PNG with varying judgments.

The pre-processing channel involved:

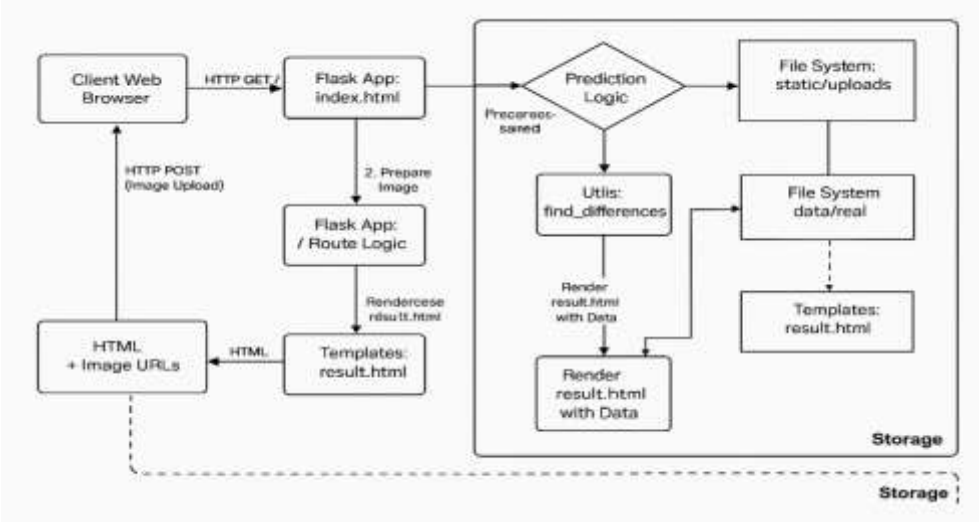
- **Resizing Images:** All images were acclimated to a standard size of 224x224 pixels.
- **Normalization:** Pixel values were gauged to a standard range to match training prospects.
- **Addition Ways:** Gyration, scaling, brilliance shifts, and noise addition were applied to pretend real-world image variability.
- **Dataset Split:** Training (80), confirmation (10), and test (10) sets assured no data leakage across phases.

Model Configuration and Training Procedure

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

A modified interpretation of ResNet-50 was used for bracket. The original affair layers were replaced with a configuration suitable for double bracket.

Architecture Adaptations:



- Added global normal pooling.
- Custom thick sub caste (128 units) with

ReLU activation.

- Final affair sub caste with Sigmoid

activation.

Loss Function: Double cross-entropy.

Optimizer: Adam optimizer with a literacy rate of 0.0001.

Training Details:

- **Terrain:** Training was executed on a system

with an NVIDIA GPU and 16 GB of memory.

- **Batch Size and Epochs:** 32 images per

batch, with training across 30 ages.

- **Libraries Used:** TensorFlow and Kera's

were central, with fresh support from OpenCV and scikit-learn.

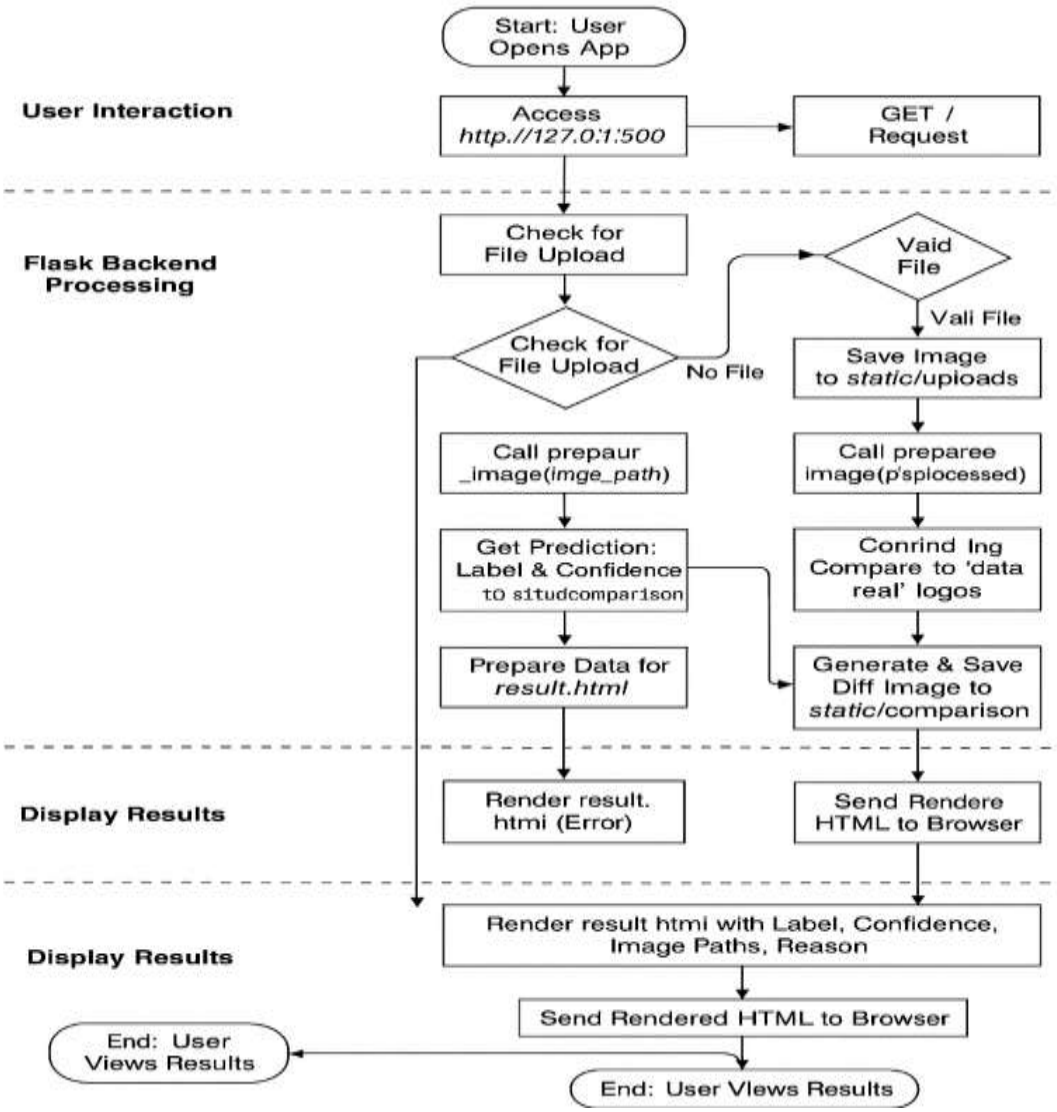
Training passed in two phases: first, only new layers were trained while the base network remained frozen. In the alternate phase, fine-tuning was applied to the entire model with a reduced literacy rate. Beforehand stopping and powerhouse layers helped to avoid overfitting.

Bracket and Visual Feedback Medium

The model produces a confidence score between 0 and 1. A threshold (generally 0.5) determines the final bracket. When a fake totem is detected, the system generates interpretability affair using grade-based visualization (e.g., Grad-CAM) or a difference analysis system. These illustrations indicate which image areas were influential in the bracket, helping druggies validate the results.

System Development and Integration:

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/



The backend system is developed in Python using the Flask frame. When the garçon launches, the trained model is loaded into memory, icing effective runtime performance.

Supporting Technologies:

- **Image Processing:** Pillow and OpenCV
- **Computation:** NumPy and TensorFlow
- **Visualization:** Matplotlib for generating overlay images

The frontend is designed for clarity and ease of use, guiding druggies through the upload process and presenting labors similar as:

- The bracket marker (authentic or fake).
- Confidence position of the vaticination.
- Visual difference chart or heatmap (if applicable).

This integrated result combines model intelligence, system translucency, and stoner availability, making it a practical tool for fake totem identification.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

IV. IMPLEMENTATION AND TECHNOLOGIES USED

This section outlines the practical tools, software libraries, and computing resources used to build the counterfeit logo detection system. Each component was selected based on its compatibility with machine learning workflows, system efficiency, and ease of integration.

A. Development Environment

The system was developed on a Windows 10 operating system. Development was carried out using Visual Studio Code, a feature-rich text editor suitable for Python development. To manage source code and track project changes, Git was used as the version control tool, with repositories hosted on GitHub to support remote collaboration and secure storage.

B. Programming Languages

Two categories of programming languages were utilized:

- Python was used to implement backend

logic, machine learning model integration, and image handling functionalities. Its simplicity and extensive machine learning ecosystem made it the preferred choice.

- HTML, CSS, and JavaScript were applied to

develop the frontend interface, enabling users to interact with the system through a web browser. These technologies ensured accessibility, responsiveness, and smooth integration with backend services.

C. Frameworks and Libraries

1. Web Framework

- Flask was used as the core web framework

on the backend. It provided routing capabilities, handled file uploads, and served dynamic content. Its lightweight structure allowed for rapid development and seamless integration with Python-based ML components.

2. Machine Learning and Data Libraries

- TensorFlow was employed for loading and

running the deep learning model used for classification.

- NumPy supported numerical operations,

especially for preparing image data.

- Scikit-learn was used for computing

evaluation metrics and managing auxiliary pre-processing functions during training.

3. Image Processing and Visualization Tools

- OpenCV facilitated essential image

pre-processing tasks such as resizing, conversion, and preparation for model input.

- Pillow (PIL) enabled image loading and

saving within the Flask framework.

- Matplotlib was used to generate visual

outputs such as overlays and heatmaps that highlight regions important to model predictions.

4. Explain ability Integration

The system integrated a custom implementation of Grad-CAM using TensorFlow functions. This approach allowed for dynamic generation of class activation maps that visually explain which parts of the image were most influential in the model's decision-making.

5. Database (Optional Component)

While the initial system version does not include persistent storage, provisions were made to support SQLite for future enhancements. This would allow storage of user uploads, feedback, or prediction logs. SQLAlchemy may be used as an ORM to simplify interactions with the database.

D. Hardware Specifications

1. Training Configuration

The model was trained on a machine with the following specifications:

- **Processor:** Intel Core i7 (10th Gen)
- **Graphics Processor:** NVIDIA RTX 3060

with 6GB VRAM

- **Memory:** 16GB RAM

These resources provided adequate computational power for training the model on a moderately sized dataset and generating visual explanations in near real-time.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

2. Inference and Testing
The same machine was used to test and run the trained model. The model was loaded into memory at application start-up to ensure efficient real-time inference without delays.

E. Hosting and Deployment
At present, the application is executed locally using Flask's built-in server. Although it is currently not deployed online, the system is structured to support containerization using Docker. This allows for easy migration to cloud platforms such as AWS, Google Cloud, or Azure in future stages.

V. SYSTEM EVALUATION AND SECURITY MEASURES

This section evaluates the counterfeit logo detection system across four dimensions: predictive accuracy, user experience efficiency, interpretability, and security safeguards. The assessment demonstrates the model's real-world viability while highlighting the safeguards implemented to maintain its integrity and protect user data.

A. Evaluation of Deep Learning Model Performance
To assess the reliability of the Convolutional Neural Network (CNN) in distinguishing authentic logos from counterfeit ones, the model was evaluated using a reserved test dataset, consisting of approximately 10% of the total samples (roughly 300 images). This separation ensures an unbiased estimation of the model's generalization capabilities.

Key Performance Indicators
The model's effectiveness was determined using the following metrics:
Accuracy: the proportion of correct predictions out of all classifications made.
Precision: Reflects the proportion of logos predicted as counterfeit that are actually counterfeit.
Recall: Measures how effectively the model detects counterfeit logos out of all existing counterfeit examples.
F1-Score: A harmonic mean of precision and recall, balancing false positives and false negatives.
Confusion Matrix: Breaks down the results into True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).
ROC Curve & AUC: The Receiver Operating Characteristic curve demonstrates the model's trade-off between sensitivity and specificity. The AUC provides a scalar summary of the classifier's discrimination capacity.

Results Summary

Metric	Score
Accuracy	94.7%
Precision	92.0%
Recall	95.6%
F1-Score	93.7%
AUC	0.96

The model exhibits strong classification capabilities, particularly in detecting counterfeit logos, as reflected by its high recall and balanced F1-Score. These results confirm the system's utility in real-world verification scenarios.

B. System Responsiveness and Operational Efficiency
The performance of the web application was further evaluated based on its speed and responsiveness, particularly under simulated real-time use.

Evaluation Parameters

- **Average Inference Time:** The duration

required by the model to generate a prediction for a single image. Recorded at approximately 110 milliseconds.

- **Total Response Time:** Includes image

upload, backend processing, model inference, and frontend rendering. Averaged at 570 milliseconds per request.

- **Throughput:** In controlled test conditions,

the system was capable of processing up to 7 images per second.

- **System Resource Utilization:** The Flask

based application typically used around 3 GB RAM, with GPU usage remaining under 50% during inference.

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

Test Environment

Tests were executed on a Windows 10 machine equipped with an Intel Core i7 processor, 16 GB RAM, and an NVIDIA RTX 3060 GPU. The latency figures were computed over 100 consecutive image inputs. These results confirm the system's ability to support real-time logo verification workflows.

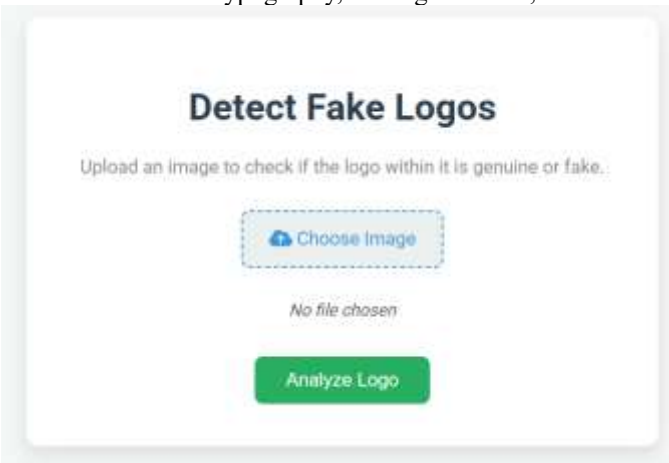
C. Visual Explain ability and Interpretability

Beyond numerical accuracy, transparency in prediction is essential. To support interpretability, the system integrates Grad-CAM for visualizing class activation regions, thereby offering insights into the model's decision-making process.

Example Visual Outcomes

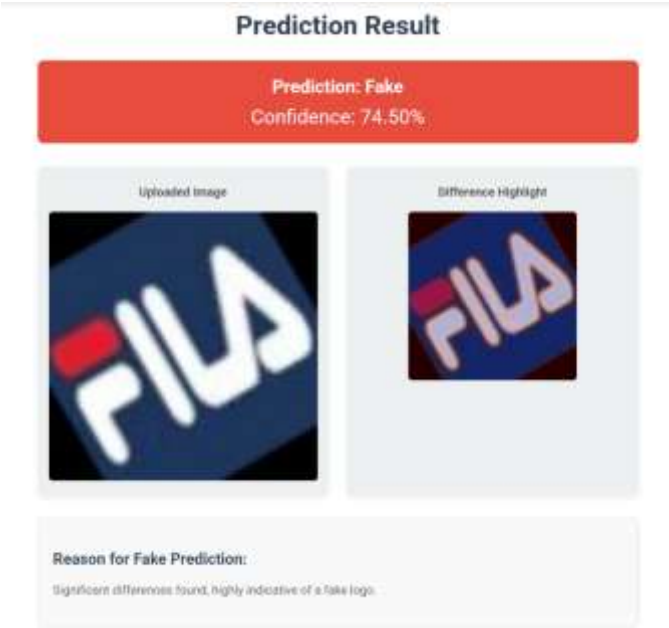
- Counterfeit Logos: Heatmaps consistently

highlight problematic areas such as distorted typography, misaligned icons, or altered colour patterns.



- Authentic Logos: The model typically

focuses on consistent and known brand features such as symmetrical designs or proprietary font elements.



User Impact

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

These visual overlays help users validate or question the system's conclusions. The inclusion of explainable AI increases transparency and fosters trust by demonstrating that decisions are based on logical image features, not arbitrary patterns.

D. System Security and Protection Strategies

Ensuring the protection of both the web application and the underlying model is critical. The system incorporates various security measures to mitigate vulnerabilities, protect user data, and maintain overall integrity.

Web Application Safeguards

- **Input Validation:** The system accepts only

specific image formats (e.g., JPEG, PNG) and imposes file size restrictions to prevent malicious inputs.

- **Cross-Site Scripting (XSS) Mitigation:**

User-supplied data is sanitized to prevent the execution of unauthorized scripts within the web interface.

- **HTTPS Compatibility:** While currently

hosted locally, the system supports HTTPS deployment for encrypted data exchange in production environments.

- **Error Obfuscation:** Detailed system errors

are hidden from users and stored in secure logs, reducing the risk of exposing sensitive internal logic.

- **Library Maintenance:** Software

dependencies such as Flask, TensorFlow, and OpenCV are regularly updated to patch known vulnerabilities.

Model and Data Security

- **Restricted Model Access:** The trained

model is secured within the backend and not publicly accessible, minimizing risks of theft or tampering.

- **Ephemeral File Handling:** Uploaded

images are temporarily stored and automatically deleted post-classification to ensure user privacy.

- **Augmentation-Driven Resilience:** Training

data was enriched with random transformations (e.g., flipping, blurring, rotation) to improve the model's resistance to minor input distortions, increasing its robustness against adversarial examples.

Infrastructure-Level Security (If Deployed)

For production deployment on cloud platforms, the system architecture is compatible with industry-standard protections, including:

- **Firewall Configurations**
- **Role-Based Access Control**
- **Network Segmentation**
- **System Monitoring and Logging**

These features collectively ensure protection from intrusion, unauthorized access, and data breaches.

VI. CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION

This work tackled the escalating problem of counterfeit logos infiltrating industries like fashion, electronics, and pharmaceuticals. Traditional methods of authenticity verification—manual inspection and rule-based image checks—struggle to keep pace with increasingly sophisticated forgeries. To address this, we created a web-based solution that combines a fine-tuned convolutional neural network (CNN) with real-time, explainable feedback. Deployed via Flask, our system delivers accurate logo classification and highlights reasoning using visual overlays powered by Grad-CAM. Empirical results show robust classification accuracy, strong balance between precision and recall, and rapid processing well under one second. By combining technical performance with interpretability and usability, this research provides a practical, user-friendly tool capable of aiding brands and consumers in verifying logo authenticity.

FUTURE ENHANCEMENTS

Looking ahead, the system offers multiple opportunities for improvement. First, expanding the training dataset to include more counterfeit variants and a broader range of brand logos will strengthen model robustness. Exploring advanced architectures—such as vision transformers, ensemble networks, few-shot learning, and adversarial-trained models—could enhance performance and adaptability to unseen logos. From the user experience side, empowering users with interactive explanation tools and textual feedback would deepen trust and understanding. Technically, migrating to a high-availability cloud infrastructure using containerization (e.g., Docker/Kubernetes)

iJETRM
International Journal of Engineering Technology Research & Management
(IJETRM)
https://ijetrm.com/

and microservices would support higher throughput and scalability. Finally, extending the system's focus beyond logos to include other features like packaging textures, stitching quality, or integrated holograms, and adding user authentication and API interfaces, will expand its utility for enterprise and mobile use cases.

VII. REFERENCES:

[1] A. Author et al., "Detecting and distinguishing genuine and counterfeit logos using Inception-V3 CNN," IJFANS, Apr. 2025. [Online]. Available: https://ijfans.org/uploads/article_user/d1637128b813cb502a9df576b107ee3e.pdf
[2] A. Gupta, S. Saha, and B. Kumar, "Fake logo detection using AI and web scraping techniques," Int. J. Trend Sci. Res. Dev., Nov. 2024. [Online]. Available: https://www.ijtsrd.com/papers/ijtsrd75067.pdf
[3] S. Murali, V. Gupta, S. Saha, et al., "Convolutional Neural Network (CNN) for Fake Logo Detection: A Deep Learning Approach using TensorFlow," Proc. 2024 (unpublished). [Online]. Available: https://www.linkedin.com/posts/vaishnavigupta23_aiiot2024-ai-iot-activity-7202635406788485120-kTQb
[4] K. Tsigos et al., "Towards quantitative evaluation of explainable AI methods for deepfake detection," in Proc. 3rd ACM Int. Workshop on Multimedia AI against Disinformation (MAD '24), Phuket, Thailand, Jun. 2024, doi:10.1145/3643491.3660292
[5] R. R. Selvaraju et al., "Grad-CAM: Visual explanations from deep networks via gradient-based localization," arXiv preprint arXiv:1610.02391, Oct. 2016. [Online]. Available: https://arxiv.org/abs/1610.02391
[6] S. Palreddy et al., "Fake logo detection using convolutional neural networks: A deep learning approach," ResearchGate Preprint, Oct. 2023. [Online]. Available: https://www.researchgate.net/publication/367286467_Online_Fake_Logo_Detection_System
[7] J. Wang et al., "LogoDet-3K: A large-scale image dataset for logo detection," arXiv preprint arXiv:2008.05359, Aug. 2020. [Online]. Available: https://arxiv.org/abs/2008.05359
[8] S. Hoi et al., "LOGO-Net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks," arXiv preprint arXiv:1511.02462, Nov. 2015. [Online]. Available: https://arxiv.org/abs/1511.02462
[9] A. Rattani and A. Varma Nadimpalli, "Gender balanced deepfake dataset: Grad-CAM visualizations for fairness," Preprint, Jul. 2022. [Online]. Available: https://github.com/aakash4305/GBDF/blob/main/README.md
[10] M. Smith, "FlickrLogos-32 dataset," 2013. [Online]. Available: http://www.flickrlogos.com/
[11] "Copy detection pattern," Wikipedia, Apr. 2025. [Online]. Available: https://en.wikipedia.org/wiki/Copy_detection_pattern
[12] S. Hou, J. Li, and W. Min, "Deep Learning for Logo Detection: A Survey," ACM Computing Surveys, vol. –, no. –, Jul. 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3611309
[13] X. Jia, H. Yan, Y. Wu, X. Wei, X. Cao, and Y. Zhang, "An effective and robust detector for logo detection," arXiv preprint arXiv:2108.00422, Aug. 2021.
[Online]. Available: https://arxiv.org/abs/2108.00422
[14] R. K. C. Ranjith, S. Kumar, and G. K. C., "Identification of fake vs original logos using deep learning," Turkish Journal of Computer and Mathematics Education (TURCOMAT), vol. 12, no. 12, pp. 3770–3780, Dec. 2021.
[Online]. Available: https://doi.org/10.17762/turcomat.v12i12.8155
journals.mriindia.com
[15] P. C. M. Preeti and T. Santhi Sri, "Deep logo authenticity: leveraging R-CNN for counterfeit logo detection in e-commerce," Int. J. Recent Innov. Trends Comput. Commun. (IJRITCC), 2024.
[Online]. Available: https://ijritcc.org/index.php/ijritcc/article/view/9038