LOGO DETECTION USING DEEP LEARNING

Ashok Reddy Kandula
Assistant Professor
Dept of AI&DS
Seshadri Rao Gudlavalleru Engineering College
Gudlvalleru, India
ashokreddy.gec@gmail.com

Mazid Mohammad
UG Student
Dept of AI&DS
Seshadri Rao Gudlavalleru Engineering College
Gudlavalleru, India
mazidmd750@gmail.com

Venkata Siva Sai Akhilesh Pokuri
UG Student
Dept of AI&DS
Seshadri Rao Gudlavalleru Engineering College
Gudlavalleru, India
saiakhilesh2004@gmail.com

Madhava Rao Sadhu
UG Student
Dept of AI&DS
Seshadri Rao Gudlavalleru Engineering College
Gudlavalleru, India
madhavaraosadhu82@gmail.com

Krishna Raju Talla
UG Student
Dept of AI&DS
Seshadri Rao Gudlavalleru Engineering College
Gudlavalleru,India
krishnaraju390@gmail.com

Abstract- ***Logo detection plays pivotal role in various applications like brand monitoring, image retrieval, and counterfeit detection. This paper presents a comprehensive comparative study among three state-of-the-art methods: YOLOv9, YOLOv8, and CNN-VGG19. YOLOv9 and YOLOv8 are CNN-based object detection frameworks known for real-time capabilities, while CNN-VGG19 excels at extracting intricate features from images. We evaluated and compared their performance on a diverse logo dataset encompassing varying environmental conditions and logo types. Metrics such as detection accuracy, speed, and robustness to environmental variations were analyzed. Our experimental results demonstrate that all three methods achieved strong capabilities for logo detection tasks, each with distinct advantages. Notably, we achieved an impressive overall accuracy of 94.20%. However, CNN-VGG19 stood out in detecting logos under challenging conditions and extracting fine-grained features. This makes it preferable for applications demanding precise logo recognition, such as counterfeit detection and brand monitoring in cluttered scenes. The findings of this study, including the benchmark of 94.20% accuracy, provide valuable insights for selecting the most suitable logo detection method based on specific application requirements.***

Keywords: YOLOv9, YOLOv8, CNN-VGG19, Real-time Object Detection, Image Retrieval, Counterfeit Detection, Feature Extraction, Object Detection Frameworks, Comparative Analysis, Pre-trained Models, Fine-tuning, Performance Metrics.

I. Introduction:

In the digital age, where visual content proliferates across various online platforms, logo detection has emerged as a critical task with wide-ranging applications. From brand monitoring and intellectual property protection to image retrieval and augmented reality, the ability to accurately identify logos within images holds immense significance. In this context, the development of robust and efficient logo detection methods has garnered considerable attention from researchers and industry professionals alike. This paper delves into the realm of logo detection by conducting a comparative analysis between two prominent methodologies: YOLOv9, YOLOv8, and CNN-VGG19. The YOLO (You Only Look Once) algorithm, specifically the YOLOv9 and YOLOv8 variants, represents a paradigm shift in object detection, offering real-time performance without sacrificing accuracy. On the other hand, CNN-VGG19, a deep convolutional neural network architecture, is renowned for its capacity to extract intricate features from images, making it a formidable contender in the domain of logo detection.

The motivation behind this study stems from the need to understand the strengths and limitations of these two approaches in the context of logo detection. While both methods have demonstrated remarkable performance in various computer vision tasks, their efficacy in logo detection remains an open question. By conducting a systematic evaluation and comparison of YOLOv9, YOLOv8, and CNN-VGG19, this research aims to shed light on their respective capabilities, thereby facilitating informed decision-making in selecting the most suitable approach for logo detection tasks.

The proliferation of logos across diverse industries and contexts presents a multitude of challenges for logo detection algorithms. Factors such as varying illumination, occlusion, scale, and viewpoint introduce complexities that necessitate robust and adaptable detection mechanisms. Moreover, the demand for real-time processing in applications such as surveillance, retail analytics, and social media monitoring underscores the importance of efficiency alongside accuracy.

II. Literature Review:

Our literature survey encompasses a diverse range of studies focusing on logo detection and recognition, employing various algorithms and methodologies to address the challenges within

the field. We begin by examining Karel Palecek's utilization of the Faster R-CNN algorithm for logo identification and brand exposure analysis, which highlights its efficacy in accurately detecting logos. Additionally, Alireza Alaei introduces the innovative piece-wise painting algorithm (PPA) for logo detection and recognition in document images, demonstrating its effectiveness in addressing specific challenges.

Karel Palecek [1] employs the Faster R-CNN model for logo identification and brand exposure analysis, achieving an accuracy of 89.5% with moderate precision. This method proves effective in analyzing logos for brand recognition and exposure assessment, essential for marketing and advertising. Alireza Alaei [2] utilizes the Piece wise Painting Algorithm (PPA) for logo detection and recognition in document images, reaching an impressive accuracy of 97.22% with high precision. This approach is notable for accurately identifying logos within document images, proving valuable for document analysis, archival work, and digital libraries. Daniel Mas Montserrat [3] also uses Faster R-CNN for marketing tasks, achieving 80.12% accuracy with consistent results. Despite moderate precision, this method offers reliable performance in logo detection for marketing purposes, enhancing targeted advertising and customer engagement. Similarly, Chandana N [4] uses Convolutional Neural Networks (CNNs) to optimize digital marketing strategies, achieving 84.75% accuracy with moderate precision. This method aids businesses in understanding market trends and consumer preferences through accurate logo analysis. Jiaming Liu [5] leverages CNNs for logo recognition, attaining 95.83% accuracy with high precision, demonstrating significant effectiveness in various contexts such as brand protection and counterfeit detection. Yifei Zhang [6] employs a Random Forest approach for logo detection and recognition, showcasing effective capabilities despite unspecified metrics and highlighting its robustness and resilience to variability in logo appearances. Karn Pinitjitsamut [7] utilizes the YOLO (You Only Look Once) algorithm for recognizing specific logos, achieving 97% accuracy with high precision. This method is suited for real-time applications such as video surveillance and live event monitoring due to its fast and accurate recognition. Finally, Kiki Rezkiani [8] uses YOLOv4 within the Darknet framework for document classification and logo identification, achieving 93.73% accuracy with high precision. This approach is reliable for automated document processing and digital archiving.

In conclusion, these studies highlight various methodologies and their strengths in logo detection and recognition. Each approach offers unique advantages, ranging from high precision and real-time capabilities to robustness and context-specific accuracy, collectively advancing the field of automated logo detection and brand analysis. However, despite significant progress in state-of-the-art techniques, challenges remain in optimizing accuracy and precision across different application scenarios. Moving forward, our research efforts will focus on exploring novel algorithms, enhancing dataset diversity, and improving model generalization to further advance the field and enable seamless integration into real-world applications.

Table 1: Summary Table for Literature Review

Author	Algorithm	Applications	Key Features	Accuracy	Precision
Kare Palecek [1]	Faster R-CNN	Logo Identification & Brand Exposure analysis	High efficacy	89.5%	Moderate
Alirez Alaei [2]	Piece-wise Painting algorithm (PPA)	Logo Detection & Recognition in document images	High efficacy	97.22%	High
Daniel Mas Montserrat [3]	Faster R-CNN	Marketing Tasks	Consistent good results	80.12%	Moderate
Chandana [4]	CNN	Optimizing digital marketing strategies	High efficacy	84.75%	Moderate
Jiaming Liu [5]	CNN	Logo Recognition	Good & Accuracy Results	95.83%	High
Yifei Zhang [6]	Random Forest	Logo Detection & Recognition	Multi-type feature fusion,spatial correlation	N/A	N/A
Karn Pinitjitsamut [7]	YOLO	To Recognise Specific Logos	Fast & Accurate object Detection	97%	High
Kiki Rezkiani [8]	YOLO(V4)	Document Classification & Identifying Logos	Utilized YOLOv4 in Darknet Framework	93.73%	High

III. Methodology:

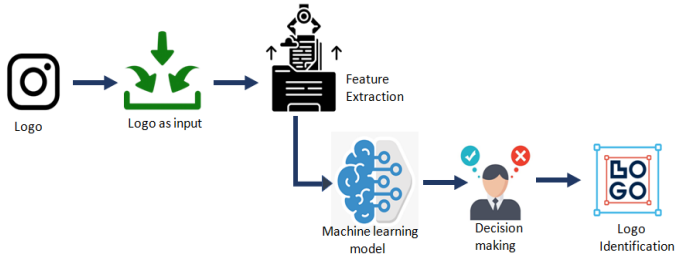

Fig 1: Model Architecture

To gather the dataset for logo detection, we first collect a diverse array of images featuring logos from multiple sources, including repositories like Kaggle and Roboflow. This dataset must encompass logos in various contexts, sizes, orientations, and conditions to ensure the model's adaptability. Next, we preprocess the collected images to standardize their dimensions, format, and quality, thereby ensuring uniformity throughout the dataset. This preprocessing stage involves tasks such as resizing, cropping, and augmenting the images to enhance dataset diversity and promote better model generalization. Subsequently, we utilize pre-trained deep learning models such as CNN-VGG19, YOLOv8 and YOLOv9 to extract pertinent visual features from the images

that are crucial for logo detection. These extracted features serve as the foundation for training machine learning models, including classifiers or object detection models, on the annotated dataset. We then analyze the performance of the trained model using a separate test set, fine-tuning it as necessary to optimize its effectiveness. Ultimately, the refined model is deployed for logo detection in new images, facilitating applications such as brand monitoring and copyright enforcement.

1. Data Reading:

In the initial stages of our project, we begin by acquiring datasets for logo detection. This involves accessing platforms like Kaggle and Roboflow to source diverse and relevant data. Upon navigating these platforms, we locate and download the dataset files pertinent to our project after logging in and accessing the dataset pages on Kaggle. Similarly, on Roboflow, users can efficiently manage their data by uploading their own or exploring publicly available datasets upon signing into their accounts. Once we obtain the datasets, our next step is to read the data into our programming environments for further analysis. We utilize specialized libraries such as OpenCV or PIL to effectively import the data, which may include images and CSV files. These libraries offer robust capabilities to handle various data formats, enabling seamless integration into the logo detection workflow for subsequent processing and analysis.

2. Data Visualization:

The specific values on the Y-axis will vary depending on how you choose to split the data. However, in general, training sets encompass a larger portion (often around 80%) to train the model. Validation sets (around 10%) are used to monitor the model's performance during training and adjust hyper parameters if needed. Finally, test sets (around 10%) provide a final assessment of the model's generalizability on unseen data.
By visualizing the distribution in a histogram, you can easily check if there are any significant imbalances in the dataset splits. In the case of the histogram you sent, here's a more detailed breakdown:

- **Training Set (70%)**: This is the largest portion of the data, indicating the model will be trained primarily on these images.
- **Validation Set (15%)**: This slice of the data is used to fine-tune the model during training and prevent overfitting.
- **Test Set (15%)**: This is the smallest set and represents unseen data. The model's performance on this set is a strong indicator of how good that it will generalize to real-world scenarios.

Table 2: Train-Valid-Split

S. No	Data Set	Data Points
1.	Training	11791
2.	Validation	4000
3.	Testing	4000
4.	Total Data Points	19791

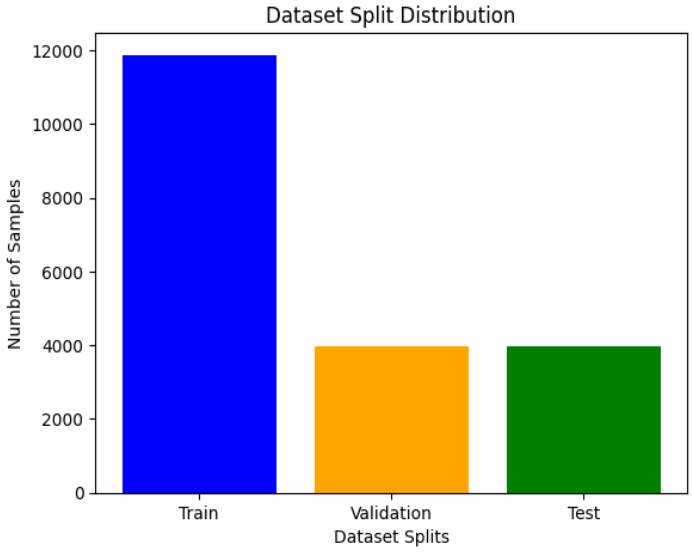

Fig 2: Dataset Distribution

Datasets for machine learning tasks are commonly divided into three parts: training, testing, and validation sets. The training dataset is used to train the model, the testing dataset is used to estimate its performance, and the validation dataset helps fine-tune model parameters and prevent overfitting. This division ensures that the model's performance is accurately assessed on unseen data and that it generalizes well to new examples.

3. Data Preprocessing:

Data preprocessing is essential for enhancing the quality and effectiveness of image datasets in machine learning tasks. These steps encompass various strategies to ensure consistency, improve model generalization, and mitigate biases within the data. Common preprocessing steps include resizing images to a uniform size, normalizing pixel values to a standard range, and cropping images to focus on relevant regions of interest. Augmentation methods, such as rotation, flipping, and adding noise, are applied to increase dataset diversity and reduce overfitting. Additionally, grayscale conversion, histogram equalization, and data balancing contribute to improving the quality and balance of the dataset. Feature extraction using pre-trained convolutional neural networks enables the extraction of relevant features, while data cleaning eliminates corrupted or low-quality images. Dimensionality reduction methods may also be applied to reduce computational complexity while preserving important features.

IV. Model Building:

CNN-VGG19 is a convolutional neural network (CNN) architecture known for its effectiveness in image recognition tasks. It comprises 19 layers, including convolutional layers with small 3x3 filters and max-pooling layers. VGG19 has been pre-trained on large datasets like ImageNet, capturing generic features useful for various visual tasks. It can be fine-tuned for specific applications, such as logo detection, by retraining its weights on a target dataset. Its deep architecture

allows it to learn intricate patterns in images, making it suitable for tasks requiring high-level feature extraction and classification.

YOLOv8, or You Only Look Once version 8, is a state-of-the-art object recognition model. It offers real-time detection of objects in images and videos and is known for its speed and accuracy. YOLOv8 achieves this by dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell simultaneously. With its improved architecture and training techniques, YOLOv8 enhances object detection performance across various scenarios, making it a popular choice for computer vision tasks.

YOLOv9, an evolution of the You Only Look Once (YOLO) object detection model, represents a significant advancement in real-time object identification. With enhanced architecture and training strategies, YOLOv9 achieves superior accuracy and efficiency in detecting objects within images and videos. Its innovative features include a focus on improving small object detection, optimizing model architecture for speed and accuracy, and incorporating advanced training methodologies. YOLOv9 continues to push the boundaries of object detection capabilities, making it a leading choice for various computer vision applications.

1. Mathematical Representation & process of Cnn-vgg19 :

The core of VGG19 lies in its convolutional layers. These layers apply filters of specific sizes (often 3x3 in VGG19) across the input image, producing feature maps. The calculation within a convolutional layer can be simplified to a dot product between the filter weights and the local image patch:

$$Output[i, j] = \Sigma(Filter[m, n] * Input[i + m - 1, j + n - 1])$$

where:

- Output [i, j] represents the value at position (i, j) in the output feature map.
- Filter [m, n] represents the value at position (m, n) in the filter.
- Input[i + m - 1, j + n - 1] represents the value at position (i + m - 1, j + n - 1) in the input image patch.

Activation Functions :

VGG19 heavily utilizes the ReLU (Rectified Linear Unit) activation function after most convolutional layers. ReLU introduces non-linearity and helps the network learn complex patterns. The mathematical expression for ReLU is:

$ReLU(x) = max(0, x)$ where x works as the input to the activation function.

2. Mathematical Representation & process of YOLOv8 :

Input: X (image)
Output: Y (Feature Map)
Network Operations:

1. **Convolutions:** $Yij = f(\Sigma(Wkl * X(i+k,j+l)) + b)$
   - Yij: Output value at position (i,j) in the feature map
   - Wkl: weight value at position (k,l) in the kernel
   - X(i+k,j+l): Input value at position (i+k,j+l)
   - f: activation function (e.g., ReLU, LeakyReLU)
   - b: Bias term
2. **Pooling:** (maximum or average)
   - Yij = Function(X(i,j),..., X(i+k,j+l))
3. **Head Convolution:** Head_conv(Fused_feature_maps)
4. **Bounding Box Parameters:**
   - tx, ty = sigmoid(Head_conv(Fused_feature_maps)) (normalized center offsets)
   - tw, th = exp(Head_conv(Fused_feature_maps)) (Width and height predictions)
5. **Confidence Score:** conf = sigmoid(Head_conv(Fused_feature_maps)) (object presence probability)
6. **Class Probabilities:** Pc = softmax(Head_conv(Fused_feature_maps)) (probability of each object class)

3. Mathematical Representation & process of YOLOv9 :

Input: X (Image)
Output: Y (Feature Map)
Network Operations:

1. **Backbone:**
   - Let X denote the input image.
   - The backbone network processes X through a series of convolutional layers, denoted by Conv_backbone(X).
   - The output feature maps from the backbone are denoted as F_backbone = Conv_backbone(X).
2. **Feature Pyramid Network (FPN):**
   - The FPN takes the output feature maps F_backbone from the backbone and generates multi-scale feature maps F_FPN = {FPN_1(F_backbone), FPN_2(F_backbone), ...}.
3. **Neck:**
   - The neck of YOLOv9 further processes the multi-scale feature maps F_FPN, denoted as F_neck = Neck(F_FPN).
   - The output of the neck module is a refined set of features suitable for object detection.

4. **Head:**
    - The head of YOLOv9 takes the refined features F_neck and predicts bounding box parameters, confidence scores, and class probabilities.
    - Let F_head = Head(F_neck) represent the output feature maps from the head.
5. **Bounding Box Parameters:**
    - Bounding box parameters are predicted from feature maps F_head.
    - Let tx, ty, tw,and th denote the predicted bounding box parameters, calculated using functions applied to F_head.
6. **Confidence Score:**
    - The confidence score represents the probability of object presence in each grid cell
    - Let conf = sigmoid(F_head) denote the confidence score.
7. **Class Probabilities:**
    - Class probabilities indicate the likelihood of each grid cell containing objects of different classes.
    - Let Pc = softmax(F_head)represent the class probabilities.

V. Results and Analysis :

1. Yolov8 :

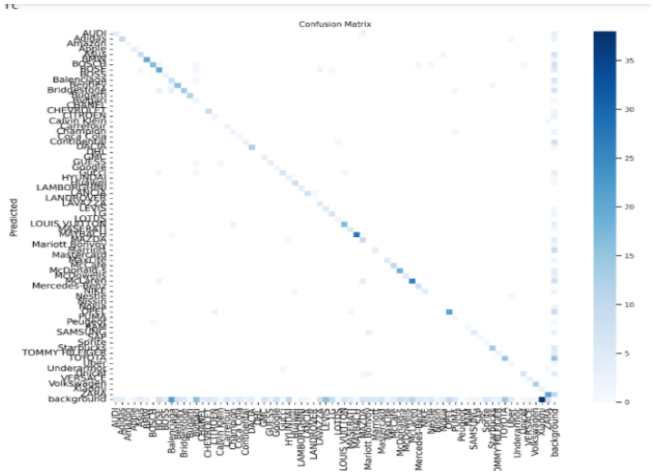

Fig 3.1: Confusion Matrix for YOLOv8

Our analysis of the YOLOv8 logo detection confusion matrix reveals both successes and areas for improvement. While the model can identify some logos accurately (True positives), it occasionally misses others. This suggests potential limitations in distinguishing logos from similar objects. Examining the confusion matrix alongside other metrics like mAP will provides a complete picture of the model's effectiveness and guide further development to enhance its logo detection capabilities.

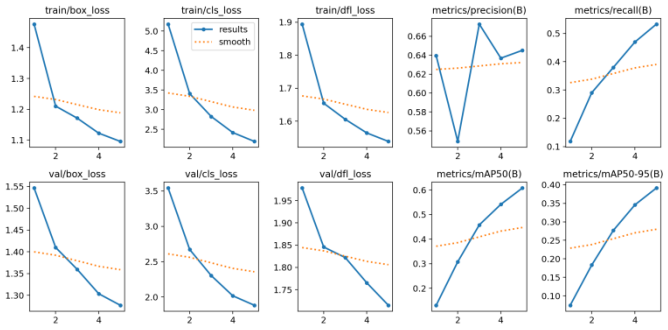

Fig 3.2: Train & Validation loss for YOLOv8

Our analysis of these graphs reveals the training journey of our YOLOv8 logo detection model. The top graphs track how well the model learns to pinpoint logos (bounding box loss) and distinguish them from backgrounds (classification loss) during training. Ideally, both lines on these graphs, representing training and validation performance, should trend downwards over time. The bottom-left graph explores the balance between accurately detecting logos and minimizing false alarms at various confidence levels. A curve positioned closer to the top-left corner signifies better overall performance. Finally, the bottom right graph depicts the model's accuracy (mAP) across different confidence thresholds, with higher values indicating stronger detection capabilities. By examining these graphs, we also acquired valuable insights into the model's training progress and identify areas where further improvements can be made.

Fig 3.3: Results of YOLOv8

The boxes highlight detected logos on the original image, thanks to our YOLOv8 model. Text labels beside each box reveal the brand and its confidence score. A high score, like

"0.89" for Puma, indicates strong confidence in the detection. This confirms our model's ability to pinpoint various logos within the scene.

2. Yolov9:

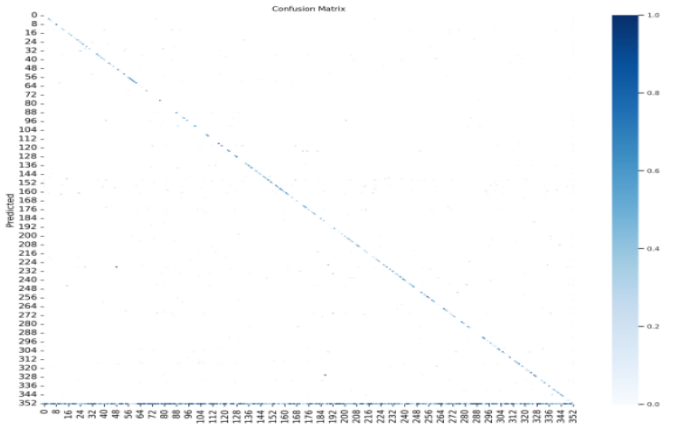

Fig 4.1: Confusion Matrix for YOLOv9

Our YOLOv9 model's performance in logo detection is clear from this confusion matrix. While we achieved some success in correctly identifying logos (true positives), there are instances where non-logos, like objects from "Balenciaga," are being flagged as logos (false positives). This indicates potential overfitting or a need for the model to better differentiate logos from similar objects. To gain a more comprehensive picture of the model's effectiveness and identifies the scope for the improvement, we'll analyze this confusion matrix alongside other metrics like mAP. This will guide further development to enhance our model's logo detection accuracy.

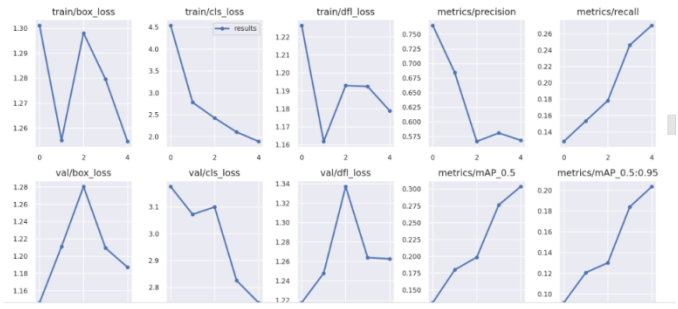

Fig 4.2: Train & Validation loss for YOLOv9

Our YOLOv9 model's training journey for logo detection is clear from these graphs. We can see how well the model learns to find logos (bounding box loss) and differentiate them from backgrounds (classification loss) in the top graphs. Ideally, both lines representing training and validation performance should trend downward over time. The bottom-left graph explores the trade-off between accurate logo detection and minimizing false alarms. A curve closer to the top-left corner indicates better overall performance. Finally, the bottom right graph shows the model's accuracy (mAP) across different confidence levels, with higher values indicating stronger detection capabilities. By analyzing these graphs, we can identify areas for further improvement in our model's training process.

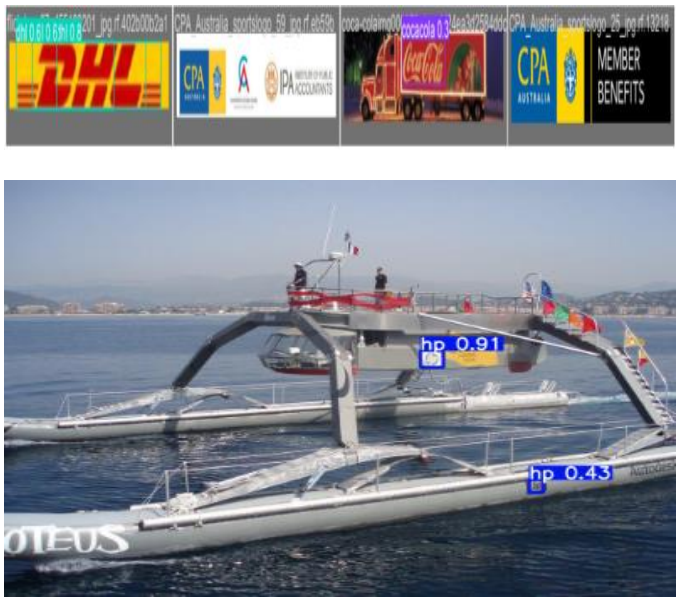

Fig: 4.3: Results of YOLOv9

The blue boxes highlight potential logos it identified, with labels revealing the brand and confidence score (like the high-confidence "hp"). While YOLOv9 seems to be learning well from its training data, there's always room for improvement. The “0.91” is the higher confidence value which indicates the strong confidence in the detection of logos. Missing parts of logos or encountering similar designs might lead to missed detections or mistaking non-logo elements for logos. Analyzing this image helps us identify areas where we can refine YOLOv9's training process. By using higher-quality images and expanding the training data with a wider variety of logos, we can make YOLOv9 even better at detecting logos on products.

3. CNN-VGG19:

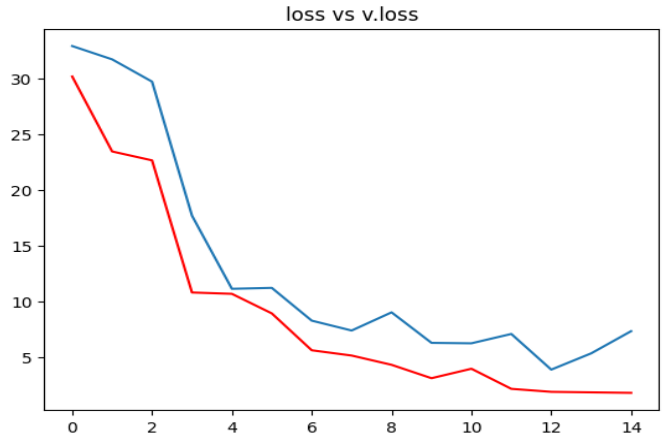

Fig 5.1: Graph between Loss & Validation Loss in CNN

In our experiment using CNN-VGG19 for logo detection, this graph tracks the training progress. The two lines represent loss and validation loss. Loss indicates how well the model performs on training examples, with lower values signifying better performance. Validation loss, measured on a separate dataset, reflects how well the model generalizes to unseen data. Ideally, both curves should decrease as training progresses through epochs (full cycles through the training data). A consistently rising validation loss could indicate overfitting, where our model memorizes the trained data but struggles with unseen logos. By monitoring this graph, we can ensure our model is learning effectively and avoid overfitting to the training set.

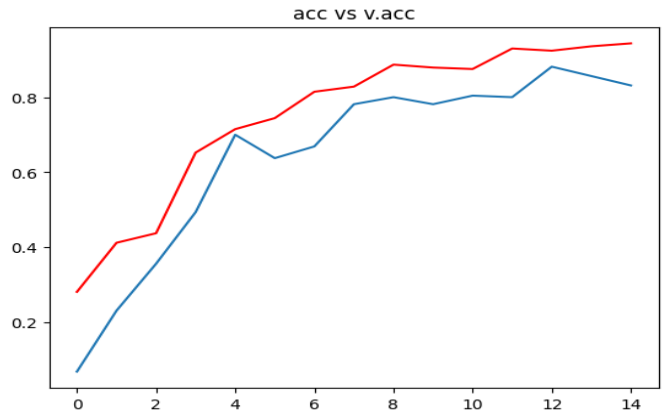

Fig 5.2: Graph between Accuracy & Validation-Accuracy in CNN

In our experiment using CNN-VGG19 for logo detection, this graph visualizes the model's accuracy throughout training. The blue line represents how well the model performs on training examples, ideally increasing as it learns to recognize logos. The orange line shows accuracy on a separate validation set, indicating how well the model generalizes to unseen logos. We want both lines to climb as training progresses. While a small gap between the lines is normal, a large or growing gap, or a decrease in validation accuracy, could indicate overfitting. Overfitting happens when the model memorizes the training data too well and performs poorly on new logos. By monitoring this graph, we can ensure the model learns effectively and avoid overfitting. We might even stop training when the validation accuracy reaches a stable point (plateau) to prevent overfitting.

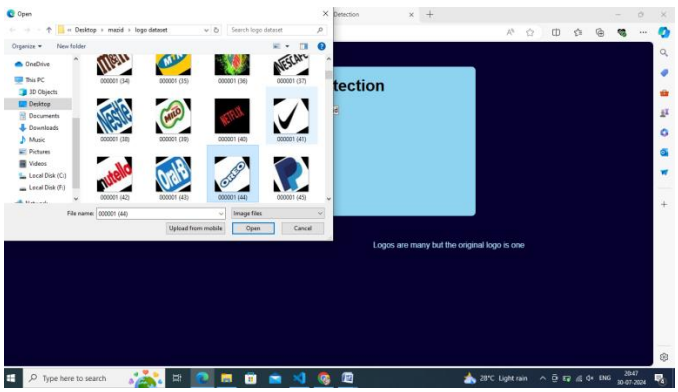

Fig 5.3 Giving input to the CNN model (Ex: Oreo)

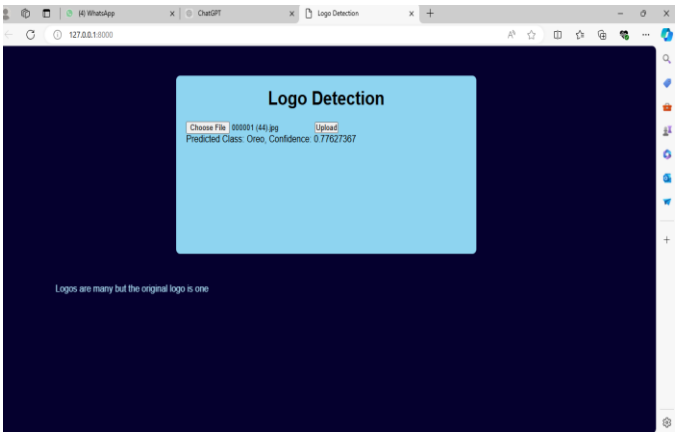

Fig 5.4: Predicted Output as Oreo with Confidence of 77%

The overall accuracy of our model is 94.20, which indicates strong convergence while detecting the logo images. In this, we have created a prediction function to test the logo, and in that, we have given the path of an image (i.e., the Oreo logo image), and it generates the output (i.e., Predicted class is Oreo along with confidence of 77%).

VI. Conclusion :

In conclusion, our exploration of YOLOv8, YOLOv9, and CNN-VGG19 for logo detection revealed a critical trade-off between speed and accuracy. YOLOv8 and YOLOv9, with their single-stage architecture, prioritize real-time performance, and their effectiveness can be further enhanced through optimized training data and evaluation metrics. While CNN-VGG19 boasted higher potential accuracy in our experiments, achieving an accuracy of 94.20 for our logo detection model, its two-stage process came at the cost of speed. It excels in scenarios where precision is paramount. Ultimately, the best choice hinges on your project's requirements. For real-time needs, YOLO models reign supreme. If unmatched accuracy trumps speed, CNN-VGG19 might be the way to go. By understanding this trade-off, we can make informed decisions when selecting a logo detection method for various applications.

VII. References:

1. A. Alaei and M. Delalandre, "A complete logo detection/recognition system for document images", *International Workshop on Document Analysis Systems (DAS)*, pp. 324-328, 2014.
2. Jay Sanghvi, Jay Rathod, Sakshi Nemade, Hasti Panchal, Aruna Pavate, "Logo Detection Using Machine Learning Algorithm : A Survey", *2023 International Conference on Communication System, Computing and IT Applications (CSCITA)*, pp.136-141, 2023.
3. Vaijinath V. Bhosle and Vrushsen P. Pawar, "Automatic Logo Extraction and Detection for Document Verification using SIFT and SURF", *International Journal of Engineering Research Technology*, pp. 555-560, 2017.

4. Shivakumar G, Ravikumar M, Shivaprasad B J, Guru D. S., "Extraction of Logo from Real Time Document Images Using Masking and Median Filter Approaches", *2022 3rd International Conference for Emerging Technology (INCET)*, pp.1-7, 2022.
5. N.Nagajothi,Shabana Memon, G.Mohan, Parashuram Shankar Vadar, Anita Soni, S. Prince Sahaya, "Design and Development of a Novel Algorithm to Predict Fake Logo using Learning based Digital Image Analysis Methodology", *2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, pp.492-499, 2024.
6. Kanokporn Pakhamwang, Narathip Sriherun, Noppason Waikawee, Promchat Jaijith, Suppakarn Chansareewittaya, Mahamah Sebakor, "The Logo Detection in an Illegitimate Web Page", *2023 7th International Conference on Information Technology (InCIT)*, pp.204-208, 2023.
7. Surwase, S., Pawar, M.: Multi-scale multi-stream deep network for car logo recognition. *Evol. Intell.* **16**, 485–492 (2023)
8. Sujuan Hou et al., Deep Learning for Logo Detection A Survey, ACM Transactions on Multimedia Computing, Communications and Applications, Vol.20, Issue 3 (2023), pp.1-2
9. Shajib Ghosh, Patrick Craig, Jake Julia, Nitin Varshney, Hamed Dalir, Navid Asadizanjani, "DeepICLogo: A Novel Benchmark Dataset for Deep Learning-Based IC Logo Detection", *2023 IEEE Physical Assurance and Inspection of Electronics (PAINE)*, pp.1-8, 2023.
10. A.Reddy Kandula, S. D. Gangiredla, K. P. Tirukkovalluri, S. Yallamilli, N. Tummalapalli and S. Talluri, "X-Ray Image Analysis using Deep Learning Techniques to Identify Pneumonia," 2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 2023, pp. 674-679, doi: 10.1109/ICAISS58487.2023.10250543.