

# Real or Deepfake Face Detection in Images and Video Data using YOLO11 Algorithm

Wawan Kurniawan<sup>1\*</sup>, Aliyah Kurniasih<sup>2</sup>, Muhamad Abdul Ghani<sup>3</sup>

<sup>1,3</sup>Universitas Bina Sarana Informatika

<sup>2</sup>Universitas Ary Ginanjar, Indonesia

[wawan.wwk@bsi.ac.id](mailto:wawan.wwk@bsi.ac.id)<sup>1\*</sup>, [aliyah.kurniasih@uag.ac.id](mailto:aliyah.kurniasih@uag.ac.id)<sup>2</sup>, [muhamad.mag@bsi.ac.id](mailto:muhamad.mag@bsi.ac.id)<sup>3</sup>

## Abstract

This study focuses on the detection of real and deepfake faces in images and video data using the YOLO11 algorithm. Deepfakes, generated using advanced deep learning techniques, have the potential for misuse, including spreading false information and identity theft. The research employs public datasets, namely EC2-DeepFake and Deepfake Dataset, to train a YOLO11-based model. The performance of the model is evaluated using metrics such as mAP50 and mAP50-95. Results indicate moderate accuracy in distinguishing real from fake faces, with notable challenges in handling diverse data. The findings emphasize the need for improved training techniques and larger datasets to enhance detection performance. This work contributes to developing tools for mitigating the risks posed by deepfake technology.

**Keywords:** Deepfake, Object Detection, YOLO11

## 1. Introduction

Etymologically, deepfake comes from the English word 'deep learning' and the word 'fake'. Where at the same time, 'fake' refers to the technical achievement of deepfakes presented in the form of unoriginal media. Deep learning is the core foundation of the technology that gave rise to deepfake software [1]. Deepfake is one of the applications of artificial intelligence technology that is used to manipulate images or videos of an object or event by using deep learning technology to perform a thorough and fundamental scan of human images [2]. Deepfake applied to human faces are very dangerous, because with the naked human eye it tends to be difficult to distinguish from the real face. The negative impact of deepfake can be in the form of the spread of false information, identity misuse, and pornography [3].

The negative impact on pornography cases is for example what happened in South Korea involving minors. Deepfake exploit images and videos by combining photos of unknown people with explicit content being disseminated. The victims included hundreds of junior high schools, high schools, and colleges [4]. Even in Indonesia, deepfake are also used to commit fraud, child exploitation, and sexual abuse [5][6]. As a result, the information spread by deepfake manipulation is very detrimental to a person, starting from psychic and material.

Object detection is a process used to find out the existence of certain objects in a digital image, be it an image or video. The detection process can be carried out using various methods that basically read the characteristics of all objects in the input image, compare the characteristics of the object in the image with the characteristics of the reference object, then compare and determine whether the detected object is the object to be detected, the detected object will be marked with a bounding box as a sign of identification [7]. However, to recognize real or artificial faces of deepfake has been done only by classifying images. For example, by utilizing the Convolutional Neural Network algorithm using an architecture of 6 convolutional layers, 3 max pooling and batch normalization layers, on 5,492 public image data that has been divided into three parts, namely train, test, and validation data as many as dual real and fake classes, the model produces a fairly good model evaluation of 91% [3]. Then in another study, they also classified real and deepfake facial images using random search to perform hyperparameter tuning on the Convolutional Neural Network algorithm, the best results were obtained by 83% on the Adam optimizer, using a combination of 32 filters measuring 3x3 on the first layer and 128 filters measuring 5x5 on the second layer [8]. Although in both studies obtained qualified model evaluation results, the classification method of pixel values in the background of the image can affect the results.

In this study, the YOLO11 (You Only Look Once 11) algorithm is used to create a machine learning model that can identify objects in the form of faces in image or video content, whether they are 'real' or 'fake' faces. Where 'fake' here is an identification that states that the content is from deepfake manipulation. The application of training from this algorithm uses public datasets, and the machine learning model is evaluated using the mAP50 and mAP50-95 metrics, and the model results are tested with new data in the form of images and video inputs.

## 2. Related Work

YOLO11 has a different name from its predecessor, YOLO, which is without the word 'version', it only stands for 'You Only Look Once 11'. YOLO11 is the latest innovation developed by Ultralytics based on the previous YOLO series, YOLOv8. YOLO11 has functions that include use for Object Detection, Instance Segmentation, Image Classification, Pose Estimation, Object Tracking, and Oriented Object Detection (OBB) cases [9][10].

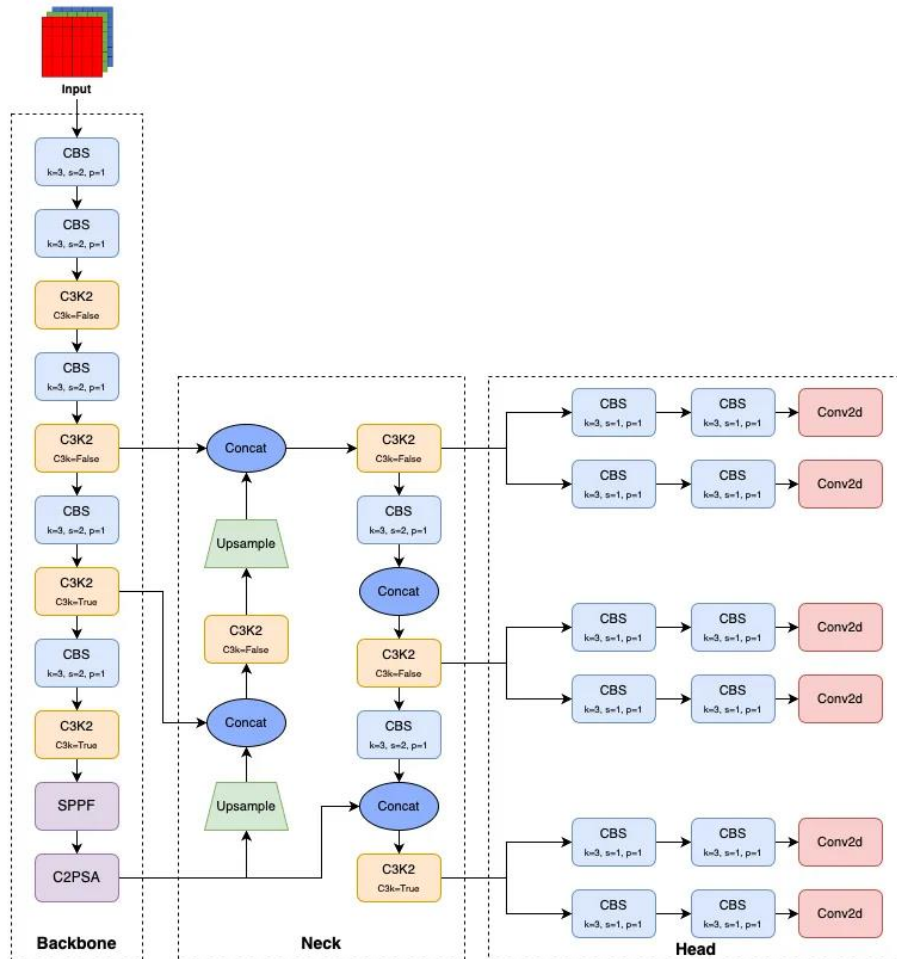


Fig. 1: YOLO11 Architecture Featuring New C3K2 Blocks and C2PSA Modules [9]

Fig. 1 above is the architecture of YOLO11 which includes Backbone, Neck and Head. YOLO11 introduces a C2PSA (Cross-Stage Partial with Self-Attention) module added behind the SPPF module, the C2PSA module combines the usefulness of cross-stage partial networks with self-attention mechanisms. The C2PSA module functions to enable the model to capture contextual information more effectively on multiple layers, and improve the accuracy of object detection. YOLO11 has a C3k2 block that blocks C2F, as well as a custom implementation of the CSP Bottleneck using two convolutions. The blocks in YOLO11 use a smaller kernel than YOLOv8 so that it can maintain accuracy while improving efficiency and speed [9][11]. Details of the structure of the SPPF module, the C2PSA module and the optimized C3k2 Block on YOLO11 are presented in Fig. 2 below.

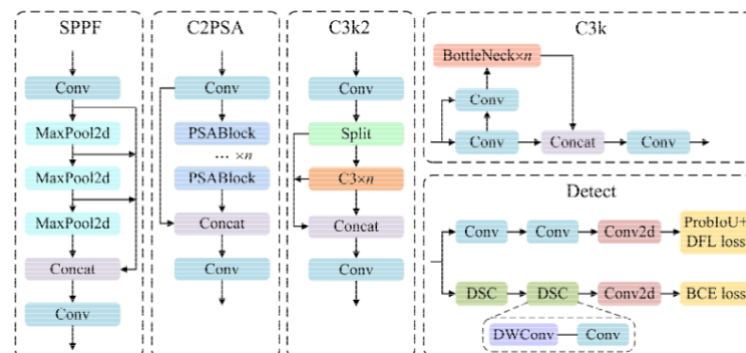


Fig. 2: The Optimized Module Structure of YOLO11 Network [11]

### 3. Research Method

The research method on machine learning is a scientific and systematic way of making machine learning models, where machine learning modeling is one part of machine learning system design to create and produce models that are trained and evaluated from data. The stages of the research method in detecting deepfake content in this study include data exploration from the image dataset used, creating data files, training custom objects using the YOLO11 algorithm, and producing models that have been trained, before being deployed, the model is evaluated first. Fig. 3 below is an overview of the stages of the research.

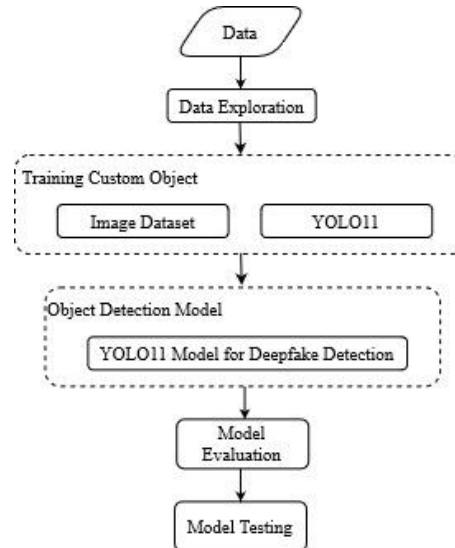


Fig. 3: Research Method

#### 3.1. Data

The dataset used is a secondary type of data because it is obtained from the Roboflow public dataset source. Two public dataset sources are used at the same time in order to have a large number of images and diverse images to obtain optimal model results, the dataset sources are EC2-Deepfake Dataset [12], and Deepfake Dataset [13]. The dataset is in the form of image data of a person (human) consisting of 'real' and 'fake' that have been annotated.

#### 3.2. Data Exploration

Data exploration is checking data related to the amount of data, image type, image resolution, and checking whether the data has been shared, data preprocessing, and data augmentation. This stage aims to understand the data used. Then combine data from the two dataset sources based on training data, validation data, and test data.

#### 3.3. Training Custom Object

Custom object training is the process of training a machine learning model using a ready-made model from an object detection algorithm on a fairly large amount of data. The custom object training in this study, namely the YOLO11 (You Only Live Once 11) algorithm model, will be trained using deepfake datasets on training data and validation data. The hyperparameters and values used for the model training process are presented in Table 1 below. These hyperparameters include using the YOLO11 version yolo11.pt model, data files in the form of YAML Ain't Markup Language, a training process with a target image size for training of 640, the number of processes that produce batches in parallel of 0, the value of the number of images processed before updating the model is 8, the number of times the learning algorithm will work to process the entire dataset is 30, and the epoch that must be waited for without any observable improvement for the early termination of training by 50.

Table 1: Hyperparameter and Value

Hyperparameter	Value
Model YOLO	Yolo11.pt
Data File	Data.yaml
Imgsz	640
Workers	0
Batch	8
Epoch	30
Patience	50

#### 3.4. Model Evaluation

Model evaluation metrics are used to measure the model's ability to detect positive samples using the Recall metrics formulated (1) [14][15]. Then the model evaluation metrics are used to measure the accuracy of the model in classifying the sample as positive using the Precision metrics formulated (2) [14][15].

Formula (3) Average Precision (AP) is a formula for calculating the area produced by the Precision-Recall curve with Recall as the X-axis and Precision as the Y-axis [11]. Formula (4) mean Average Precision (mAP) is a model evaluation metrics for object detection, which is calculated at several Intersection over Union (IoU) thresholds ranging from 0.5 to 0.95. Average Precision that occurs at each threshold of IoU [16].

$$\text{Recall} = \frac{TP}{TP+FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2}$$

$$AP = \int_0^1 \text{Precision}(\text{Recall})d(\text{Recall}) \tag{3}$$

$$mAP = \frac{1}{|\text{IoU thresholds}|} \sum_{\text{IoU thresholds}} AP \tag{4}$$

### 3.5. Model Testing

After the model has gone through the evaluation stage, the next process is to test the model in the form of object detection using images data which is actually known to be Real or Fake, so that the results of this model test will prove whether the model has successfully detected facial images according to the actual data. The image used at this stage is an image sourced from public data [17]. Model testing is also carried out on video data sourced from Youtube to identify whether the video contains a 'real' or 'fake' face object [18].

## 4. Result and Discussion

### 4.1. Data Exploration Result

In EC2-DeepFake Dataset, it is an RGB (Red, Green, Blue) image with various image resolutions, including 850x472 pixels, 426x448 pixels, 626x500 pixels, and so on. The dataset consists of a total of 844 images. The dataset has been divided into three parts, namely train data, validation data, and test data, each of which is 70% (591 images), 20% (170 images), and 10% (83 images). The results of the data exploration are presented in Table 2 below.

**Table 2:** Data Exploration Results

Data Source	Amount of Data	Image Type	Image Resolution (pixel)	Split Data		
				Train	Val	Test
EC2-DeepFake Dataset	844	RGB	850x472, 426x448, 626x500, dll	591 (70%)	170 (20%)	83 (10%)
Deepfake Dataset	47	RGB	640x640	99 (88%)	9 (8%)	5 (4%)

Then Table 2 on the Deepfake Dataset only has a total of 47 images of RGB type image data (Red, Green, Blue) with a resolution of only 640x640 pixels. Where the data is divided into 88% of the train data as many as 99 images, 8% of the validation data as many as 9 images, and 4% of the test data as many as 5 images. However, in this dataset, even though it only consists of 47 images, the data has gone through the data preprocessing stage in the form of resize the image into a resolution of 640x640 pixels. Then the data has also gone through the augmentation data stages in the form of Flip, Grayscale, Hue, Blur, Noise, Cutout, Bounding Box Noise, and output per sample. Details of the augmentation data process are presented in Table 3 below.

**Table 3:** Augmentation Data

Parameter	Value
Flip	Horizontal and Vertical
Grayscale	20% of images
Hue	Between -78 and +78 degree
Blur	Up to 10px
Noise	Up to 9% of pixels
Cutout	3 boxes with 18% size each
Boudix box noise	Up to 9% of pixels
Output per-training example	3

Table 4 below is the result of the total data from the merger of two dataset sources based on the division of train data, validation data, and test data. Produced 690 train data for the model training process, 179 validation data for the model validation process during the training process, and 88 test data for the model evaluation process.

**Table 4:** Amount of Data

Split Data	Data Source		Amount of Data
	EC2-DeepFake Dataset	Deepfake Dataset	
Train	591	99	690
Val	170	9	179
Test	83	5	88

## 4.2. Training Model Result

Model yolo111.pt with Ultralytics version 8.3.65, programming language, Python 3.11.11, torch 2.5.1 engine, Tesla T4 GPU, 15102MiB RAM. Resulting in model training with an architecture of 464 layers, showing a very deep architecture, this depth allows the model to capture more complex features but can also increase the risk of overfitting. With 25,280,854 parameters, it describes a sizable model, indicating the complexity of the model that determines the model's capacity to learn patterns from the training data. Based on this study, the number of parameters showed that it was too large when compared to the amount of data used, so there was a possibility that the model could be overfitted. The model generates 0 gradients, which indicates the model is ready to be used for prediction. 0 gradients means that there is no gradient that currently no changes have occurred to the model parameters. Generating 86.6 GFLOPs (Giga Floating Point Operations per Second) measures the number of floating-point operations performed per second in the model. The results show that the model is quite computationally-intensive but still within a reasonable range, it requires considerable computing resources.

Table 5 shows the results of training the YOLO11 algorithm model using train data and validation data with epochs of 30. The Images attribute refers to the number of classes detected by the model in the validation data that serves to validate the model during the training process, the number of images containing Real and Fake images is 179, consisting of Real images as many as 55 and Fake images as many as 124. The Instances attribute is the total object contained in the validation image, having a total of 182 Instances, consisting of 127 Fake images, and 55 Real images.

Training the model yielded a not-so-high accuracy value of only 51.5%, which illustrates that the model produces more incorrect predictions. Then a very high recall value of 95.5%, which illustrates that the model rarely misses existing objects. Details of the precision and recall value results of each class are presented in Table 5 below.

Furthermore, the model training produced a mean value of Average Precision at the Intersection over Union (IoU) threshold of 0.5 for all classes of 65.8% or 0.658, this result means that the model is not good enough in measuring how well the model recognizes objects with an overlap tolerance of 50%. Then it produces a mean Average Precision at various IoU values from 0.5 to 0.95 with an increase of 0.05, this mAP50-95 is stricter than mAP50 because it considers various overlap bounding box scenarios, resulting in an accuracy of 54.6% or 0.546. Details of the mAP50 and mAP50-95 grades in each class are presented in Table 5.

**Table 5:** Training Model Result

Class	Images	Instances	Box			
			Precision	Recall	mAP50	mAP50-95
All	179	182	<b>0.515</b>	<b>0.955</b>	<b>0.658</b>	<b>0.546</b>
Fake	124	127	0.696	1	0.828	0.702
Real	55	55	0.333	0.909	0.488	0.389

Fig. 4 below is a graph of the results of the model training process for 30 epoch using train data and validation data. The Loss Function parameter is the error measure used to optimize the model. The loss function results of this study in the train data decreased significantly with the increase in epoch value. Likewise, the validation data also experienced a decrease in the Loss Function value from epochs 1 to 10, but in epochs 10 to 30 there was no significant decrease in the Loss value.

Box\_loss that occurs both in the train data and the validation data describe the Loss value for the prediction of bounding box coordinates (x, y, w, h) where this measures how far the bounding box predicts from the ground truth. Based on the results of box\_loss on the train data and validation data, where the box\_loss in the validation data is slightly higher than the box\_loss the train data, this illustrates that the model is overfitting. Then cls\_loss describes the loss value for the classification of objects in the bounding box, where this uses the Binary Cross Entropy (BCE) function, if the model is wrong in classifying the object then this loss value will be high. Based on the results of cls\_loss on higher validation data and stagnant from the train data, this illustrates that the model will have difficulty in classifying objects, or the model will experience underfitting. Furthermore, the dfl\_loss (Distribution Focal Loss or DFL) describes the Loss value related to the scale distribution of the bounding box feature, this is used to improve the accuracy of object scale prediction. Based on the results of the Loss Function of this dfl\_loss which is higher than box\_loss and cls\_loss, it illustrates that the model will have difficulty in predicting the scale of the object.

Fig. 4 below also illustrates the results of model training using metrics precision, recall, and mean average precision in measuring the model's performance in detecting objects. This study produced a precision value of only 51.5% in the 30 epoch, meaning that it is not too high, where this illustrates how accurate the model is in detecting objects, the higher the precision value during training, the more accurate the model will be, the results of this study the model will sometimes give wrong predictions. However, resulting in a high recall value, which is 95.5%, which describes that the model can capture most objects, but with an increased recall value and a decrease in precision value, the model may have predicted a lot of False Positive objects.

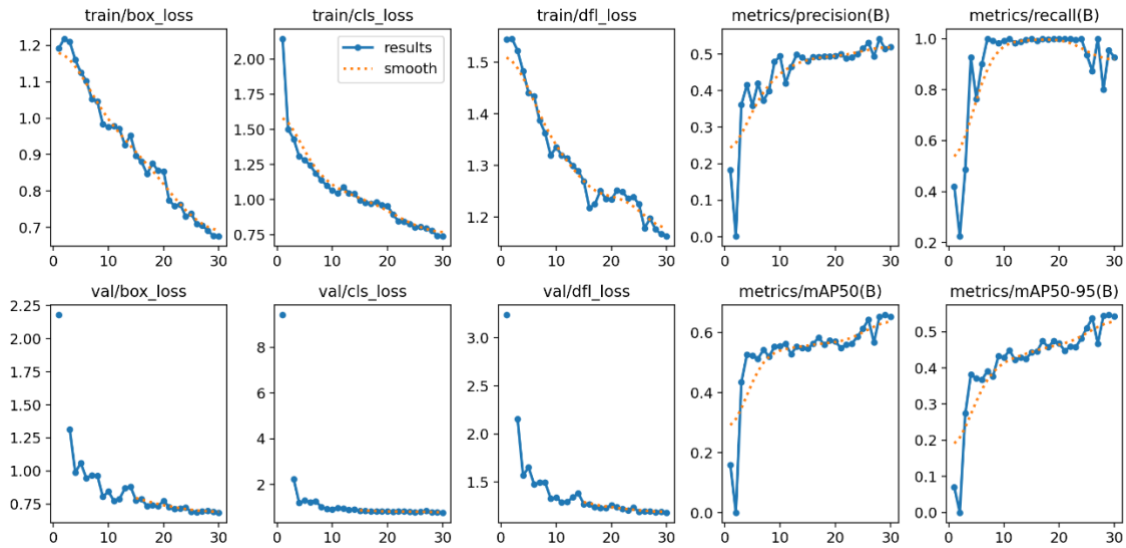


Fig. 4: Graph Training Model Result

### 4.3. Model Evaluation Result

The model evaluation uses test data of 88 images, consisting of 63 fake images, and 25 real images. The model evaluation produces 89 Instances, consisting of 64 Fake objects, and 25 Real objects. Based on the results of Table 6, it produces a model accuracy of 68.2% on mAP50, and 57.8% on the overlap bounding box that is mAP50-95, with a precision value of 58.1% and a recall value of 84.9%. Details of the model evaluation results are presented in Table 6 below.

Table 6: Evaluation Model Result



Class	Images	Instances	Box			
			Precision	Recall	mAP50	mAP50-95
All	88	89	<b>0.581</b>	<b>0.849</b>	<b>0.682</b>	<b>0.578</b>
Fake	63	64	0.72	1	0.863	0.736
Real	25	25	0.442	0.697	0.501	0.421

### 4.4. Model Testing Result

Table 7 below shows the results of testing the model using image data that is completely unknown to the model or new image data. In the first image, the model failed to recognize and provide a bounding box along with the prediction results, this happened because the image tested was a non-photo face image, while the data created for model training was data that was really a real or fake photo image. Then in the second image, it is known that the image is actually a fake image, but the model results provide prediction results in the form of real images, in this test the model fails to provide the correct prediction results, this can be because the training data has very few fake data images when compared to real images, so the model will have difficulty recognizing fake images.

Then in the third image, the model succeeded in giving the correct prediction result, namely that the image was indeed a fake image, the prediction result gave an accuracy level of 76%, this happened because the image used for this test was a photo image that was indeed a fake, not a non-photo image. Furthermore, in the fourth image, the test gave the correct prediction result, that the image is true real, with an accuracy level of 65%, this happens because it is true that the image is a real photo image.

Table 7: Model Testing Results on Image Data

No	Images	Actual Class	Resolution Images	Image Prediction	Result Class Prediction
1		Fake	2000 x 2000 pixel		Failed to detect


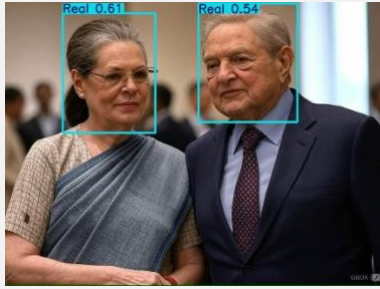



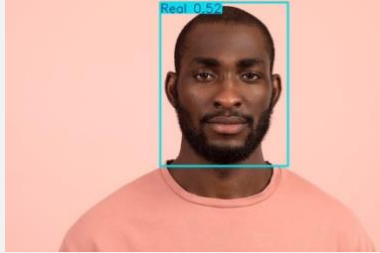
No	Images	Actual Class	Resolution Images	Image Prediction	Result Class Prediction
2		Fake	960 x 720 pixel		Real
3		Fake	626 x 417 pixel		Fake
4		Real	626 x 417 pixel		Real

Fig. 5 below is a screenshot of the model test using video data [18]. In the video is an example of Elon Musk deepfake video made by a group of people who are interested in the potential of deepfake, they make short videos to entertain to show the general public what deepfake may do. The video shows Elon Musk talking, where the video shows two sides of Elon Musk image, where the left side is Elon Musk original image and the right side is Elon Musk deepfake image. The test results show that, for the 'real' image side of Elon Musk, it is detected correctly as a Real label with an accuracy of 55%, and the deepfake image side of Elon Musk produces incorrect detection, namely the actual image in the form of a deepfake but predicted to be Real with an accuracy of 44%.

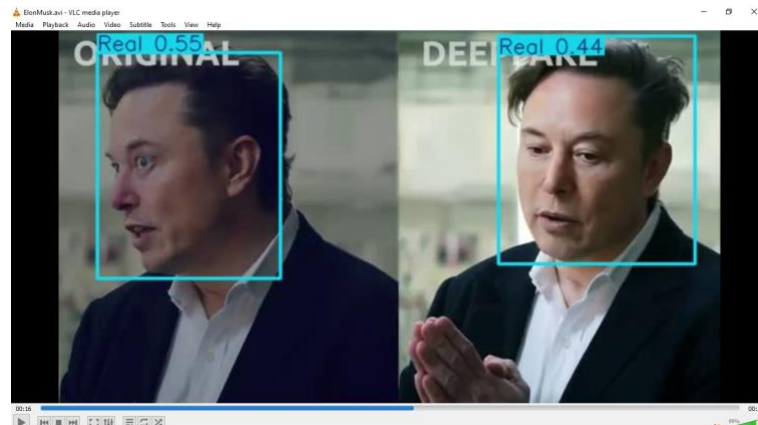


Fig. 5: Model Testing Results on Video Data

### 5. Conclusion

The YOLO11 algorithm demonstrates potential in detecting deepfake and real faces, achieving an evaluation accuracy of 68.2% (mAP50) and 57.8% (mAP50-95). While the model shows high recall (84.9%), indicating its ability to capture most objects, its precision is limited (58.1%), reflecting challenges in avoiding false predictions. Testing revealed that the model struggles with non-photographic images and imbalanced datasets. These results highlight the importance of utilizing larger and more diverse datasets, as well as refining training processes to improve overall performance. Despite its limitations, this study underscores the feasibility of using YOLO11 for deepfake detection, providing a foundation for further advancements in safeguarding digital media authenticity.

## References

- [1] C. T. Noerman and A. L. Ibrahim, "Kriminalisasi Deepfake Di Indonesia Sebagai Bentuk Pelindungan Negara," *J. Usm Law Rev.*, vol. 7, no. 2, pp. 603–621, 2024.
- [2] S. A. U. Sijabat and D. Lukitasari, "Konten Gambar dan Video Pornografi Deepfake Sebagai Suatu Bentuk Tindak Pidana Pencemaran Nama Baik," *J. Huk. Pidana dan Penanggulangan Kejahatan*, vol. 13, no. 2, pp. 179–194, 2024.
- [3] J. Mu, M. Adrezo, and A. N. Haikal, "Identifikasi Wajah Asli dan Buatan Deepfake Menggunakan Metode Convolutional Neural Network," *Teknika*, vol. 13, no. 1, pp. 45–50, 2024.
- [4] L. Hae-rin, "More Koreans fall victim to deepfake sex crimes," *The Korea Times*. [Online]. Available: [https://www.koreatimes.co.kr/www/nation/2025/01/113\\_381251.html](https://www.koreatimes.co.kr/www/nation/2025/01/113_381251.html). [Accessed: 02-Jan-2025].
- [5] "Polisi Ungkap Sisi Kelam AI, untuk Penipuan hingga Pelecehan Seksual," *CNN Indonesia*. [Online]. Available: <https://www.cnnindonesia.com/teknologi/20241129161114-185-1172121/polisi-ungkap-sisi-kelam-ai-untuk-penipuan-hingga-pelecehan-seksual>. [Accessed: 02-Jan-2025].
- [6] "Penipu di Lampung Pakai Deepfake AI Prabowo Raup Rp30 Juta," *CNN Indonesia*. [Online]. Available: <https://www.cnnindonesia.com/nasional/20250123165315-12-1190773/penipu-di-lampung-pakai-deepfake-ai-prabowo-raup-rp30-juta>. [Accessed: 26-Jan-2025].
- [7] G. N. Rizkatama, A. Nugroho, and A. F. Suni, "Sistem Cerdas Penghitung Jumlah Mobil untuk Mengetahui Ketersediaan Lahan Parkir berbasis Python dan YOLO v4," *Edu Komputika J.*, vol. 8, no. 2, pp. 91–99, 2021.
- [8] Darmatiasia, A. R. Ramli, A. Salsabila, and F. Adiba, "Analisis Performa Convolutional Neural Network dengan Hyperparameter Tuning dalam Mendeteksi Gambar Deepfake," *J. Insypro*, vol. 9, no. November, pp. 1–8, 2024.
- [9] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "Evaluating the Evolution of YOLO (You Only Look Once) Models: A Comprehensive Benchmark Study of YOLO11 and Its Predecessors," *arXiv*, no. October, pp. 1–20, 2024.
- [10] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements," *arXiv*, no. October, pp. 1–9, 2024.
- [11] J. Huang, K. Wang, Y. Hou, and J. Wang, "LW-YOLO11: A Lightweight Arbitrary-Oriented Ship Detection Method Based on Improved YOLO11," *Sensors*, vol. 25, no. 1, 2025.
- [12] EC2, "EC2-DeepFake Dataset," *Roboflow*. [Online]. Available: <https://universe.roboflow.com/ec2/ec2-deepfake-oxj9m>. [Accessed: 16-Dec-2024].
- [13] N. Workspace, "Deepfake Dataset," *Roboflow*. [Online]. Available: <https://universe.roboflow.com/new-workspace-ugbw4/deepfake-gglmq>. [Accessed: 16-Dec-2024].
- [14] M. Bakhara Alief Rachman, A. Kurniasih, A. Sundawijaya, and A. Nuraminah, "Penerapan Blok Se-Net Pada Deep Learning Inceptionv3 Untuk Meningkatkan Deteksi Penyakit Mpox Pada Manusia," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 10, no. 7, pp. 1447–1452, 2023.
- [15] A. Kurniasih and L. P. Manik, "On the Role of Text Preprocessing in BERT Embedding-based DNNs for Classifying Informal Texts," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 6, pp. 927–934, 2022.
- [16] M. A. R. Alif, "YOLOv11 for Vehicle Detection: Advancements, Performance, and Applications in Intelligent Transportation Systems," *arXiv*, no. October, pp. 1–16, 2024.
- [17] Saurabhbagchi, "Deepfake image detection," *Kaggle Dataset*, 2025. [Online]. Available: <https://www.kaggle.com/datasets/saurabhbagchi/deepfake-image-detection>. [Accessed: 29-Dec-2024].
- [18] LipSynthesis, "Deepfake Example. Original/Deepfake Elon Musk.," *LipSynthesis*. [Online]. Available: <https://www.youtube.com/watch?v=WFc6t-c892A>. [Accessed: 29-Dec-2024].