MEAT FRESHNESS PREDICTION USING YOLO 11, DETECTRON2 AND SWIN TRANSFORMERS

INTRODUCTION

Meat quality is one of the most important factors influencing consumer choice, but it is highly susceptible to deterioration during the post-harvest process. Freshness is a crucial indicator of meat quality, affecting not only its taste, texture, and nutritional value but also its safety. The deterioration of meat leads to the growth of pathogenic microorganisms, which can pose significant health risks. As a result, accurate and efficient methods for predicting meat freshness are essential for ensuring food safety and minimizing health risks. Additionally, the global meat industry faces challenges in reducing food waste due to improper storage, transportation, or delayed processing of meat. As meat products move through the supply chain, proper freshness monitoring becomes essential for reducing spoilage, ensuring safe consumption, and meeting consumer demand for high-quality products. Thus, there is an urgent need for an automated, reliable system capable of assessing meat freshness, which will enhance food quality control, improve supply chain efficiency, and promote sustainability within the industry.

ML algorithms can be trained on large datasets to detect patterns and correlations between visual characteristics like color, texture, and marbling, and meat freshness levels. These algorithms can also identify specific features that may not be easily discernible to the human eye, improving prediction accuracy. Deep learning, by learning hierarchical features from raw image data, CNNs can identify complex, subtle variations in meat's appearance that are indicative of its freshness. In addition, the introduction of transformer-based models, which excel in capturing long-range dependencies and contextual information, has further enhanced the predictive capabilities of these systems. Transformers are particularly useful in handling sequential or spatial data, making them ideal for tasks involving high-dimensional image data, such as analyzing meat freshness. These advanced techniques enable highly accurate and efficient freshness prediction systems that can be deployed in real-world applications, from quality control in processing plants to shelf-life monitoring in retail.

This research aligns primarily with Sustainable Development Goal (SDG) 12: Responsible Consumption and Production, which encourages sustainable production and consumption practices. In particular, the goal emphasizes reducing food waste and improving the efficiency of food production systems. Furthermore, this research also supports SDG 3: Good Health and Well-being, as it focuses on ensuring that consumers are provided with safe, high-quality food.

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

By ensuring meat freshness through automated prediction, the likelihood of foodborne illnesses due to improper storage or spoilage is significantly reduced. In this way, the project contributes to improving public health by preventing the consumption of unsafe meat and promoting responsible food handling practices throughout the supply chain.

I contribute to the field of meat freshness classification by employing cutting-edge object detection models, both one-stage and two-stage detectors, to classify meat freshness levels accurately. I specifically utilize the YOLOv11 model (a one-stage detector) and various two-stage detectors such as Detectron2, Mask R-CNN, Faster R-CNN, and the Swin Transformer, which have been adapted for this task. One-stage detectors like YOLOv11 are known for their speed and efficiency, allowing real-time freshness classification, while two-stage detectors offer higher accuracy through more refined region proposals and feature extraction. Detectron2, Mask R-CNN, and Faster R-CNN leverage region-based CNNs for object detection, making them suitable for handling complex image data in meat freshness prediction. The Swin Transformer, a novel transformer-based model, incorporates both vision and spatial awareness, offering significant improvements in handling spatial relationships in images.

This paper is organized into the following sections: Section 2 provides a comprehensive review of existing literature on meat freshness prediction, focusing on traditional methods, machine learning techniques, and advances in deep learning for food quality assessment. The section also explores the challenges and opportunities in applying computer vision techniques to meat freshness classification. Section 3 outlines the methodology, detailing the object detection models employed in the study—YOLOv11, Detectron2, Mask R-CNN, Faster R-CNN, and Swin Transformer—and explaining their selection for this task. Section 4 describes the experimental setup, including the dataset used, preprocessing steps, and evaluation metrics for assessing model performance. The section also presents the process of training and fine-tuning each model to ensure optimal results. Section 5 discusses the results, comparing the performance of the different models based on accuracy, precision, recall, and inference time. It also provides insights into the strengths and weaknesses of each approach in the context of meat freshness prediction. Finally, Section 6 concludes the paper, summarizing the findings and providing recommendations for future research directions, including potential improvements in model performance, real-world deployment, and broader applications within the food industry.

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

LITERATURE SURVEY

The quest to improve meat quality monitoring methods has evolved considerably, with various studies proposing innovative approaches ranging from traditional techniques to advanced machine learning and deep learning methods. This literature survey reviews the progression of technologies and methodologies used for assessing meat freshness, beginning with simpler, more traditional approaches, followed by the introduction of machine learning models and cutting-edge deep learning techniques.

Early Simple Approaches

The journey begins with fundamental techniques such as physical measurements and sensory evaluations to monitor meat freshness. Soo-Kyoung Lee et al. [1] conducted one of the earliest studies into classifying broiler breast meat using basic tools, such as the Torrymeter, to assess meat lightness (L* value). This method was effective for initial freshness assessment but lacked sophistication for broader, real-time applications across various meat types. Similarly, Qingying Luo et al. [2] developed a composite chitosan film embedded with clove essential oil to preserve pork and monitor its freshness. This material-based solution showed promise by extending pork's shelf life, but it highlighted the limitations of relying on chemical changes rather than dynamic, real-time monitoring techniques.

Simple Machine Learning Approaches

As technology advanced, Zheng-Xu Fu et al. [3] introduced a more data-driven approach to yak meat preservation by employing metabolomics to track meat composition changes during preservation. This shift from simple observations to data analysis paved the way for machine learning models to handle dynamic data more efficiently. D.-E. Kim, N.-D. Mai, and W.-Y. Chung [4] further advanced the field by creating an AIoT-based system for real-time meat quality monitoring. The system utilized multi-sensor data fusion, combining image analysis and gas emissions, and employed machine learning models such as SVM and decision trees. Although the system marked an important leap in real-time freshness classification, challenges like sensor calibration and environmental noise limited its scalability.

Recent Advances in Sensor Technologies and Machine Learning

In recent years, further innovations have emerged in the area of meat freshness monitoring. M. J. Oates et al. [5] developed a low-cost handheld electronic nose (e-nose) unit using MQ series gas sensors to assess food quality, including lamb, hake, salmon, beef, pork, and chicken. This system successfully achieved over 95% classification accuracy but revealed the need for further sensor calibration

and a broader dataset for optimal sensitivity and specificity. Similarly, Saman Abdanan Mehdizadeh et al. [6] proposed an AI-driven, non-destructive system for meat freshness detection using a multi-indicator sensor array and smartphone technology. Their approach demonstrated strong performance metrics, but challenges like sensor response consistency across varying environmental conditions underscored the need for a more diverse dataset to improve model generalization. Michela Albano-Gaglio et al. [7] explored the potential of visible and near-infrared (VNIR) spectral imaging for assessing pork belly quality, focusing on attributes like fatness and firmness. Their study achieved accurate predictive models using regression techniques and spectral imaging, with future potential for deep learning to optimize real-time quality control. Lastly, Eko Prasetyo et al. [8] developed a standardized method for classifying fish freshness during ice storage using clustering algorithms. Their approach achieved reliable freshness classification but suggested that further exploration into different storage conditions and species could enhance the robustness of the model. Taoping Liu et al. [9] proposed a novel, data-driven approach for meat freshness monitoring using a metal-oxide-semiconductor (MOS) sensor-based electronic nose, coupled with Hidden Markov Models (HMMs). Their system integrated a Segmental-Rapid Centroid Estimation (RCE) algorithm to optimize the HMM, allowing it to track freshness in real-time based on fresh meat samples. This method demonstrated excellent sensitivity and specificity across fish, beef, and chicken, showcasing its robustness for various meat types. However, they recognized the need for further validation with larger datasets and different environmental conditions, a limitation also seen in subsequent studies. These concerns about dataset and environmental variability are echoed in the work of Hui Lu et al. [10], who applied machine learning models to predict the freshness of pork patties using Total Volatile Basic Nitrogen (TVB-N) levels and an electronic nose.

Hui Lu et al. [10] utilized machine learning models, including BP-ANN, SVM, and PLS-DA, to assess pork patty freshness stored at 4°C. Their study found that BP-ANN achieved the highest prediction accuracy (94%), aligning well with actual measurements of TVB-N. This research highlighted the effectiveness of machine learning, particularly deep learning, in predicting freshness and parallels Liu et al.'s emphasis on real-world application. However, Lu et al. also recognized the need for further validation across a broader range of samples and storage conditions, similar to the need identified by Liu et al. for more extensive testing in different environmental settings. In a more visual approach, Jie Wang et al. [11] introduced a colorimetric microneedle sensor (CMS) integrated with deep learning algorithms for real-time, visual monitoring of meat freshness. Their CMS, which changes color as spoilage progresses, coupled with a convolutional neural network (CNN), enabled classification of meat into categories such as

"fresh," "less fresh," and "spoiled." This system offered a user-friendly, portable solution via a smartphone app. Although this method provided high accuracy, the study, like others, noted limitations due to image quality and potential variations based on meat type and storage conditions. This aligns with the challenges highlighted by both Liu et al. and Lu et al., where environmental factors and sample variability affected the system's performance. The research of Wei Gong et al. [12] built up deep learning methods by creating a smartphone system for meat freshness detection. GelMA colorimetric indicator bars and CNN model created a system that reached 96.2% accuracy when analyzing meat freshness images. Similar to the Civilized Monitoring System used by Wang et al., the study depended on visual indicators which presented possible constraints in monitoring spoiling foods under different light conditions or those exhibiting small color variations during spoilage. The visual sensitivity problem during different conditions points to one of the system limitations Wang et al. discussed about image quality variations.

The determination of large yellow croaker freshness through CNN and computer vision analysis is presented in Yao Zheng et al.'s work [13]. The modified ResNeXt CNN model enabled researchers to reach 84% classification success at 24-hour examination points. CNNs demonstrated their real-time capability for detecting fish freshness changes at their early stages according to the study. The research brought attention to elements that influence model accuracy including image clarity and storage variables which matched the findings of Liu et al. and Lu et al. The detection system needs additional improvement to accurately process multiple fish standards and storage conditions based on their findings about prior research limitations. Fish freshness monitoring is performed by the fluorescence-based sensor array developed by Xia Xu et al. [14] which combines lanthanide metal-organic frameworks (Ln-MOFs) and deep learning technology. The technique displayed freshness detection through a fluorescent progress bar which CNNs improved accuracy rates. The research method recorded an outstanding classification accuracy of 98.97% which established an efficient fish freshness monitor although additional testing for species varieties as well as specific environmental conditions would be beneficial. The examined challenges match the spoilage condition variability and environmental factor concerns that Zheng et al. specified along with the prior research. The combination of AI technology with robotic arming systems for sorting chicken meat based on freshness was reported by Mahamudul Hasan et al. [15]. The combination of InceptionV3 CNN with LIME framework delivered sorting accuracy of 96.3% according to their results. The decision-making process of AI was made more transparent by this system which improved interpretability during meat sorters operations. The research documented various problems which would affect real-world deployment including the variations in meat quality and environmental

factors. The system stability across diverse conditions as mentioned in Liu et al. and Gong et al. creates operational challenges and environmental limitations because of its unclear scalability potential across different settings.

Advances in Deep Learning and Computer Vision approaches

Dayuan Wang et al. utilized DCNN alongside a fluorescence-based sensor array for vegetable freshness monitoring [16]. Their system reveals high accuracy for detecting yardlong beans freshness along with spinach and sweet corn freshness by employing curcumin together with puerarin and fisetin for pH-sensitive fluorescence analysis. The analysis system showed an exceptional ability to detect spoilage which resulted in 98.58% precision when monitoring yardlong beans. The system's widespread use could be enhanced by testing with additional vegetable types to address sample generalization limitations just like problems encountered when applying meat freshness detection methods. The scalability and generalizability issue persists in similar fashion in the research published by Yuandong Lin et al. [17]. Yuandong Lin et al. [17] used a paper-based colorimetric sensor array (CSA) as a freshness detector of beef through deep learning algorithm implementation. A network model named ResNet34 led to 98.0% accuracy in the detection of amine gases during spoilage conditions based on their developed system. The CSA technology demonstrated promising capabilities for real-time freshness monitoring of beef similar to the research conducted by Wang et al. on vegetable freshness. The detection system designed by Lin et al. showed successful results for beef freshness analysis but faced a limitation because it needed particular dye and amine gas types which reduced its ability to measure other kinds of spoiled foods. The food freshness monitoring field often encounters this problem because approaches commonly remain limited to one specific type of food substance as reported in various existing research studies.Xinyi Liu et al. [18] designed a ratiometric fluorescence platform which detects hypoxanthine (Hx) in order to perform meat freshness assessments. The detection accuracy was enhanced by their system which processed colorimetric images through deep learning YOLOv5 using a smartphone-assisted device. The system operated with high sensitivity but had potential scalability problems and environmental factors that could affect its accuracy according to Liu et al. All research investigations addressing field adaptation and consistent results across storage environments share the common problem of achieving robustness thus requiring additional field testing for enhancement.

A seafood freshness detection system for shrimp was developed by Mingxin Hou et al. [19] who integrated YOLO-Shrimp with visual information combined with biochemical and microorganisms data. Multiple freshness indicators including TVB-N and TVC delivered the system better recall and precision rates than conventional methods in shrimp freshness classification tasks. Similar to Liu et

al.'s work the model developed by Hou et al.'s had constraints owing to its application on a single shrimp species therefore needed additional validation for broader application. The issues of specificity toward individual species along with variations in environmental factors also affect the research conducted by Wang et al. for vegetables and Lin et al. for beef freshness assessment. MRI analysis supported by computer vision methods let Trinidad Perez-Palacios et al. [20] identify and forecast beef quality while separating raw and frozen-thawed samples. Through their research the authors confirmed MRI potential for non-invasive testing which led to high accuracy freshness classification of beef products. The research of Perez-Palacios et al. shared the same observation as Liu et al. and Hou et al. about selecting appropriate algorithms yet indicated that customized solutions should be created based on sample types and objectives. Real-time implementation of MRI remained unexamined in this study which indicates that effectiveness does not equal practicality when used in hectic everyday conditions. Z. W. Bhuiyan explains a system combining deep learning and gas sensors for beef and mutton freshness and species classification in "IoT Based Meat Freshness Classification Using Deep Learning" [21]. The system relies on a particular Convolutional Neural Network (CNN) trained with 9,928 images to attain 99% accuracy in meat freshness classification. The system includes gas sensory components that detect ammonia and methane when spoilage occurs. The system provides real-time freshness evaluation but its performance reliability depends significantly on the dataset quality and sensor calibration may encounter issues from environmental conditions.

The detection of meat freshness by Min Li et al. [22] incorporates deep learning assistance but employs a technique based on flavonoid-based fluorescent sensors array assessment. Freshness-dependent fluorescence changes of the sensors can be analyzed through smartphone imaging before deep convolutional neural network (DCNN) processing occurs. The system delivered 97.1% precision which established it as another real-time non-destructive method for meat freshness assessment. Just like Waqas et al.'s system Li et al.'s device faces performance challenges due to sensor drifts across prolonged use. The system utilizes an effective dataset but researchers should consider adding more diverse samples encompassing multiple meat kinds stored under different environmental conditions. This calls for extensive testing to enhance system scalability and generalization capabilities. Yuandong Lin et al. [23] constructed a system which utilized a fluorescent sensor array (FSA) produced from copper metal nanoclusters (CuNCs) along with fluorescent dyes through integration with SqueezeNet deep learning model. Using their system operators can execute real-time detection rapidly while it maintains a 98.17% precision rate for detecting meat freshness levels. The researchers improved model interpretation capacity through Grad-CAM alongside UMAP visualization methods. Its quick

operational speed poses limitations because the system works best with limited compatibility for various types of meat while handling different storage environments. The same food freshness detection challenge was addressed by Guangzhi Wang et al. [24] while focusing specifically on pork evaluation. Scientists created colorimetric sensor array (CSA) with deep learning models for freshness measurement through color variation identification which performed at 99.5% accuracy rate. The installed system performs real-time detection in addition to mobile deployment through a 53-point colorimetric sensor array (CSA) with analysis capability from ColorNet. The pork quality overview application developed by the researchers demonstrates system potential in industrial controls but still needs improvements because it fails to achieve identical detection effectiveness over all pork food types. Real-world applications require additional research to boost system reliability since the present challenges affect all systems.

Approach using Machine Transformer for freshness monitoring

The smart seafood freshness monitoring solution was developed by Muhammad Waqas et al. [25]. A pH-sensitive film functions within their system that detects spoilage-related compounds including Total Volatile Basic Nitrogen (TVB-N) through color changes and is processed with their Visual Encoder Transformer (VET) model. The detection system achieves exceptional results by maintaining an accuracy level of 98.44% and provides an efficient sustainable non-invasive seafood freshness monitoring solution. The seafood freshness solution from their research works exceptionally well but requires advanced testing on diverse food products and exhibits the same focus on a single food category. The system effectiveness depends on careful sensor calibration alongside environmental conditions because these factors impact its long-term reliability as well as scalability like other discussed alternatives.

RESEARCH SUMMARY FOR MEAT FRESHNESS

The field of meat freshness monitoring has evolved through time as researchers move past fundamental physical and sensory methods toward complex ML and DL-based systems according to literary research. Initial detection approaches functioned well but failed to provide suitable solutions for live continuous meat quality assessment across different contexts. The adoption of Support Vector Machines (SVM) together with decision trees represented an advancement which made data-based solutions possible. The realization of widespread sensor systems remained restricted because of persistent technical barriers with sensor tuning along with environmental interferences and limited data size factors. Convolutional Neural Networks (CNNs) together with sensor arrays demonstrate excellent potential for enhancing real-time non-invasive freshness detection in

recent years. The implementation of meat freshness detection systems continues to face general obstacles because the techniques struggle to adapt to various meat species under different storage conditions and changing environmental factors. Existing detection systems experience difficulties when maintaining real-time operations that achieve high accuracy levels across different environmental conditions. Models which enhance both fast detection capabilities and precise performance standards need development for dependable action across different scenarios. Current models possess limitations when it comes to extracting meaningful features from disorderly and high-dimensional information. The addition of attention mechanisms into advanced models enables more precise identification of essential freshness indicators leading to better detection precision. Existing systems work within restricted boundaries because they were developed to handle particular species alongside particular storage conditions. The widespread use of spoilage detection models would rise substantially if these models displayed capabilities to recognize diverse meat products under various spoilage environments. The need emerges for sensors which demonstrate resilience to environmental changes such as lighting conditions and temperature variations since these elements affect measurement precision. The current systems base their information on a single data source that produces restricted results. The combination of different data types including visual, chemical and sensory data would generate more precise freshness evaluation techniques to enhance overall monitoring system performance. The development of superior deep learning frameworks requires ability to process multiple data modalities while handling sensor data complexity along with successful operation across diverse meats under different environmental conditions. Models which possess superior capabilities to analyze space and time data structures along with real-time detection functions and attention system integration will substantially boost freshness monitoring performance. Implementation of adaptable model structures must occur to make freshness detection systems robust for real-world applications when facing various spoilage situations and storage environments.

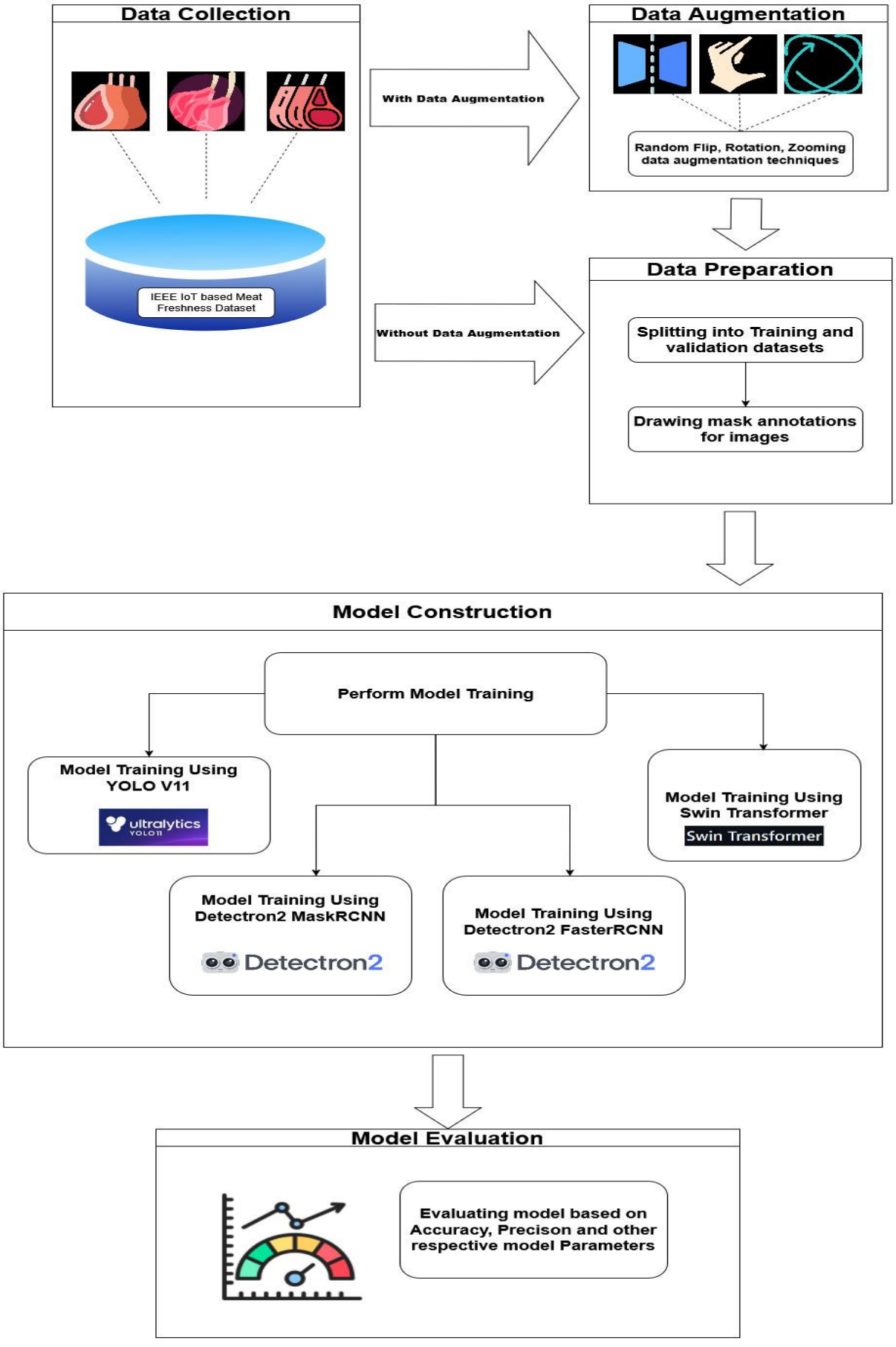
RESEARCH GAP IDENTIFIED FOR MEAT FRESHNESS MONITORING

The literature reveals significant advancements in meat freshness monitoring, progressing from traditional methods based on physical measurements and sensory evaluations to more sophisticated machine learning (ML) and deep learning (DL) approaches. While early techniques were useful in controlled environments, their scalability for real-time, dynamic monitoring was limited. The introduction of machine learning models, such as Support Vector Machines (SVM) and decision trees, marked a progression, allowing for more data-driven

solutions. However, challenges related to sensor calibration, environmental noise, and dataset limitations continued to hinder widespread implementation. In more recent years, deep learning methods, including Convolutional Neural Networks (CNNs) and sensor arrays, have shown potential in improving real-time, non-invasive freshness detection. Despite this, common barriers remain, such as difficulties in adapting systems to diverse meat types, storage conditions, and environmental variations.

Many existing systems face challenges in balancing real-time detection with high accuracy, particularly under varying environmental conditions. There is a need for models that can improve both speed and precision, ensuring reliable performance in diverse settings. Additionally, current models often struggle with effectively extracting relevant features from noisy or high-dimensional data. Advanced models, particularly those utilizing attention mechanisms, could better identify key freshness indicators, enhancing detection accuracy. Furthermore, most systems are tailored to specific meats and conditions, limiting their versatility. Models that can generalize across different meat types and spoilage scenarios would significantly increase their applicability. Environmental factors, such as lighting and temperature, can also negatively affect sensor accuracy, emphasizing the need for more robust models that are less sensitive to these variations. Finally, current methods often rely on a single data source, which limits their accuracy. Integrating multiple data types, such as visual, chemical, and sensor data, would offer a more comprehensive and accurate approach to freshness assessment, improving the overall performance of the monitoring systems. The need for the development of advanced deep learning frameworks that can integrate multiple data modalities, handle complex, noisy sensor data, and generalize well across different meats and environments. Models that excel in both spatial and temporal data analysis, such as those with real-time detection capabilities and attention-based mechanisms, could significantly enhance freshness monitoring. Furthermore, the adoption of more flexible and scalable models that can adapt to different spoilage types and storage conditions will be essential to the development of robust, real-world freshness detection systems.

METHODOLOGY



Name: PRANAY GORANTLA
REG.NO: 21MIS0123

This research approach expands from collecting datasets to model assessment stages with fresh meat detection as the main focus. Multiple deep learning detectors such as two-stage and one-stage along with machine transformers worked on analyzing the IoT-based meat freshness dataset.

Data Collection

The "IEEE Dataport IoT-Based Meat Freshness Dataset" supplied the data for this research work. The dataset consists of multiple images together with sensor records from beef and chicken and pork meat types. Researchers used freshness indicators derived from spoilage level observations at different time stages and gathered a broad variety of samples that incorporated environmental changes such as storage conditions and temperature and lighting variables. These images were captured using IoT-enabled sensors and cameras, providing valuable real-time data that reflects the dynamic changes in meat freshness.

Data Augmentation

The data augmentation techniques helped overcome dataset size limitations along with its variability issues. The system applied multiple techniques which included random rotations and flips together with scaling and cropping for creating simulated environmental conditions. Through this process we expanded the diversity of training data which improved both generalization ability and testing capability for the models.

Model Construction

This research applies three main deep learning architectures including the one-stage detector YOLOv11 functions as the selected model for conducting real-time freshness detection. With a single pass through the detector system you obtain both detection and classification operations which render it efficient for real-time monitoring needs. The YOLOv11 model achieves high accuracy on big datasets at rapid speeds. Two stage detectors MaskRCNN and FasterRCNN operate within detectron2 to first suggest region proposals and later classify these proposals thereby providing accurate detection of freshness indicators. MaskRCNN improves detection capabilities through pixelwise segmentation since it delivers detailed freshness analysis results. The region proposal network from FasterRCNN provides fast detection that enhances meat freshness detection performance. Swin Transformer operates as a Machine Transformer because it manages to detect intricate spatial-temporal data relationships. Swin Transformer performs better than typical convolutional models because it employs attention mechanisms which efficiently analyze images that show complicated spoilage patterns alongside diverse textures in meat freshness monitoring.

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

Evaluation of Model Performances

The performance evaluation of all models employed accuracy together with precision and recall and F1-score validity metrics. Model accuracy was the main assessment metric for determining general model validity but precision and recall measurements analyzed how well the models detected freshness during different stages of spoilage.

RESULTS:

The IEEE Dataport IoT-Based Meat Freshness Dataset is designed to evaluate the freshness of meat products based on various IoT-based measurements. This dataset specifically focuses on mutton (sheep meat), with images captured at different time intervals after slaughter. The freshness of the meat is determined through visual inspection and analysis, which can be correlated with the time elapsed since slaughter. This dataset contains images that represent the freshness of mutton at various stages. The key feature of this dataset is the categorization of mutton images according to the time elapsed since slaughter, with each category representing a different level of meat freshness. The data is divided into the classes as 0-hour Mutton Images – Images taken immediately after slaughter. 12-hour Mutton Images – Images taken 12 hours after slaughter. 24-hour Mutton Images – Images taken 24 hours after slaughter. 36-hour Mutton Images – Images taken 36 hours after slaughter. 48-hour Mutton Images – Images taken 48 hours after slaughter. These images are used to assess the freshness of the mutton by applying various image classification techniques and potentially IoT-based sensors for quality monitoring. The implementation system setup consists of amd 7 cpu with Jupyter Notebook and Google Colab for execution of code. The dataset includes two versions of image distribution: Before Data Augmentation: This represents the original count of images available in each class before any augmentation techniques are applied. After Data Augmentation: This represents the distribution of images after augmentations are applied to artificially increase the size and variability of the dataset. This often helps to enhance the generalizability of models when training classification.

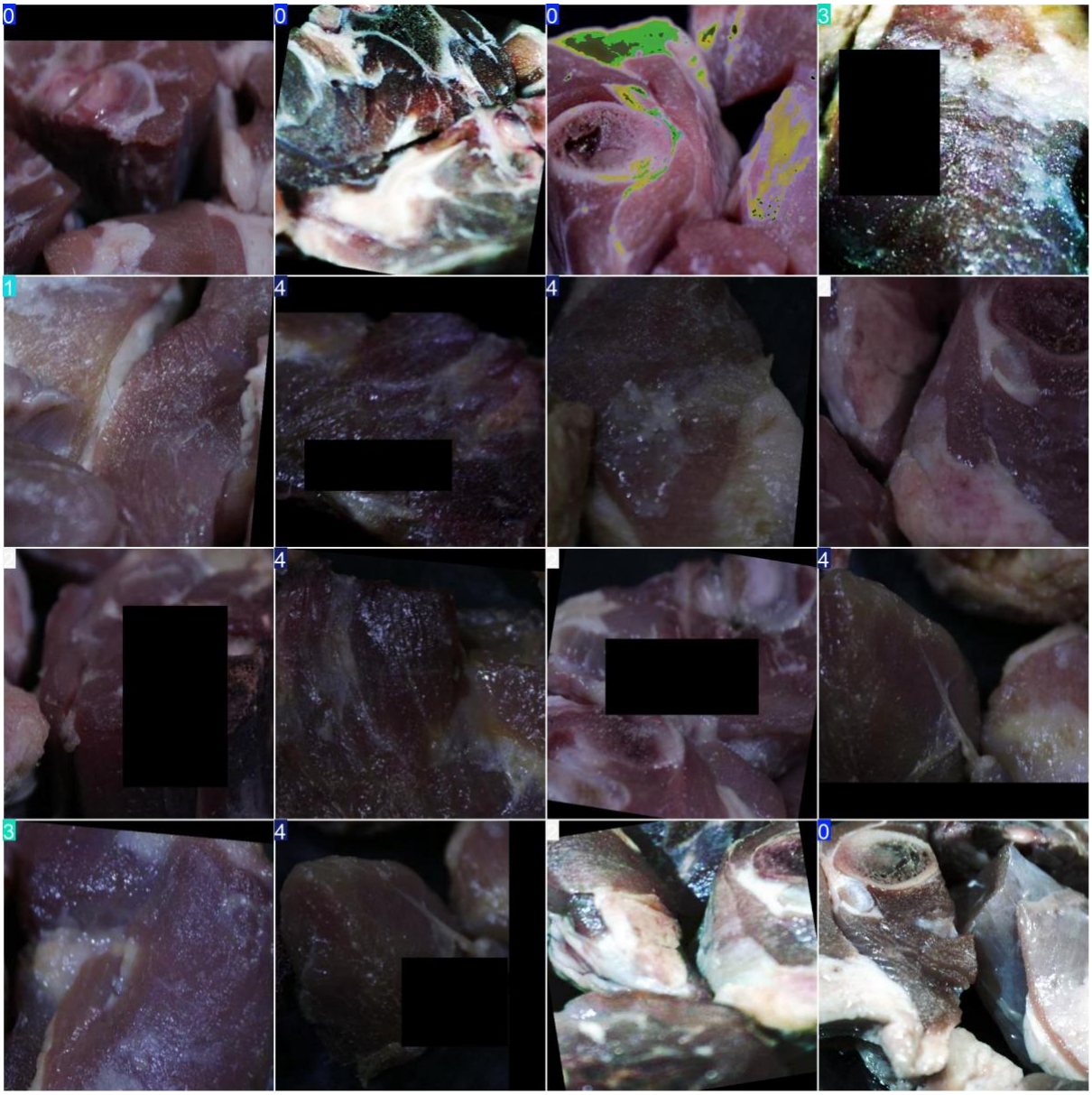
DISTRIBUTION OF IMAGES

CLASS NAMES/ TECHNIQUE APPLIED	Before Data Augmentation	After Data Augmentation
0 hour Mutton images	206	494
12 hour Mutton images	209	501
24 hour Mutton images	216	500
36 hour Mutton images	148	460
48 hour Mutton images	213	493

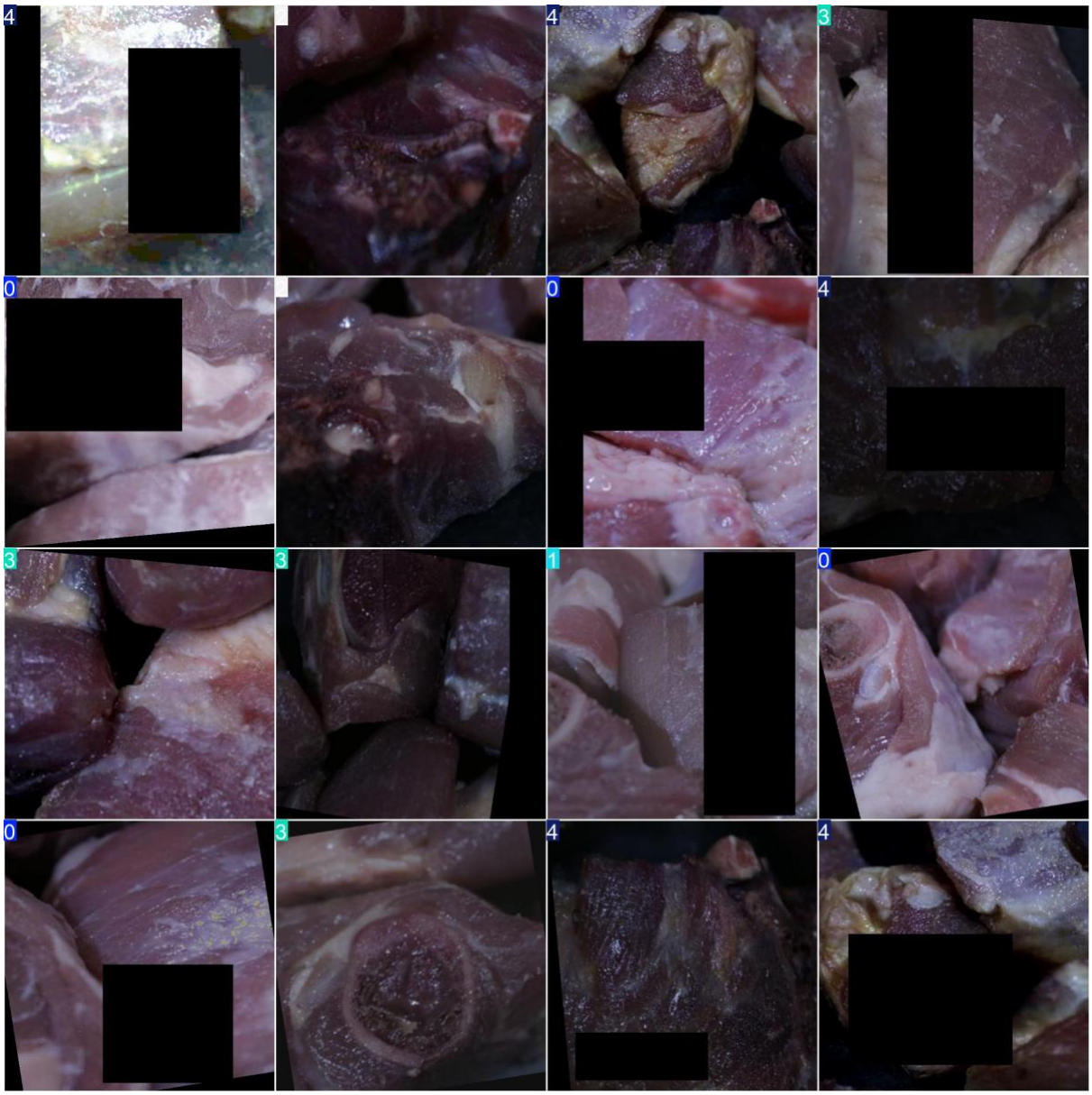
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

Comparision of model performances before data augmentation:

YOLO V11 RESULTS:

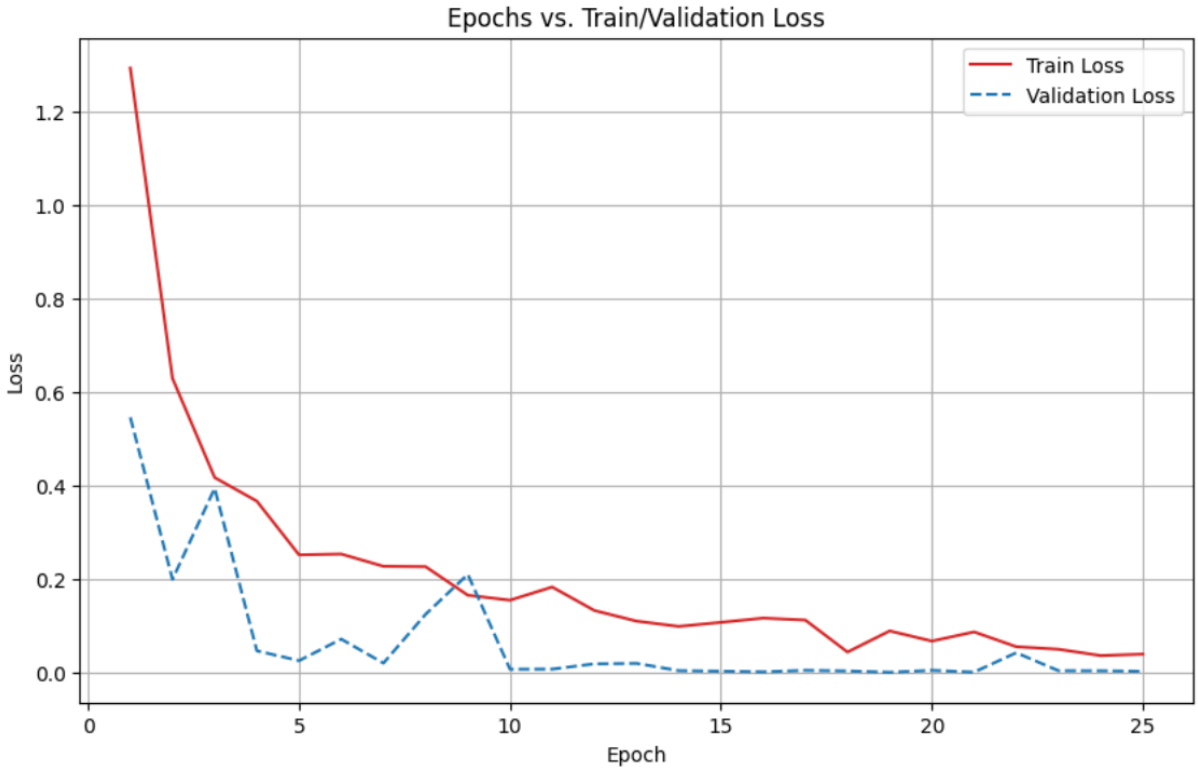


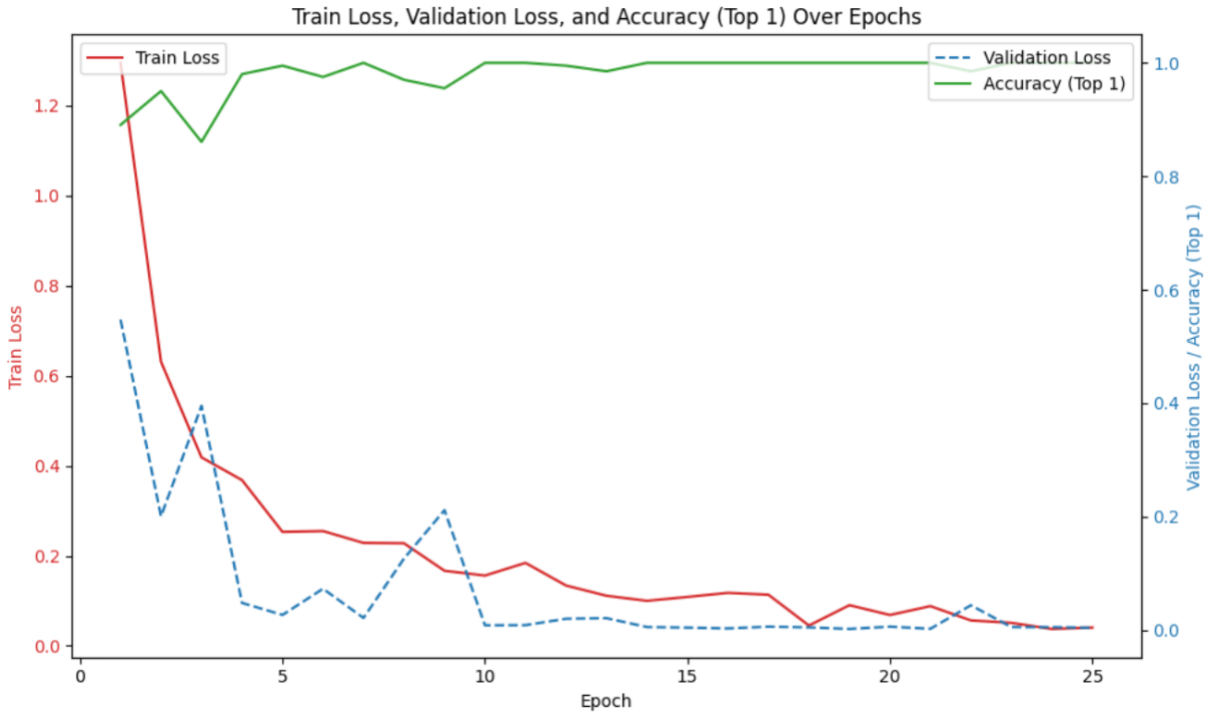
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



Name: PRANAY GORANTLA
REG.NO: 21MIS0123

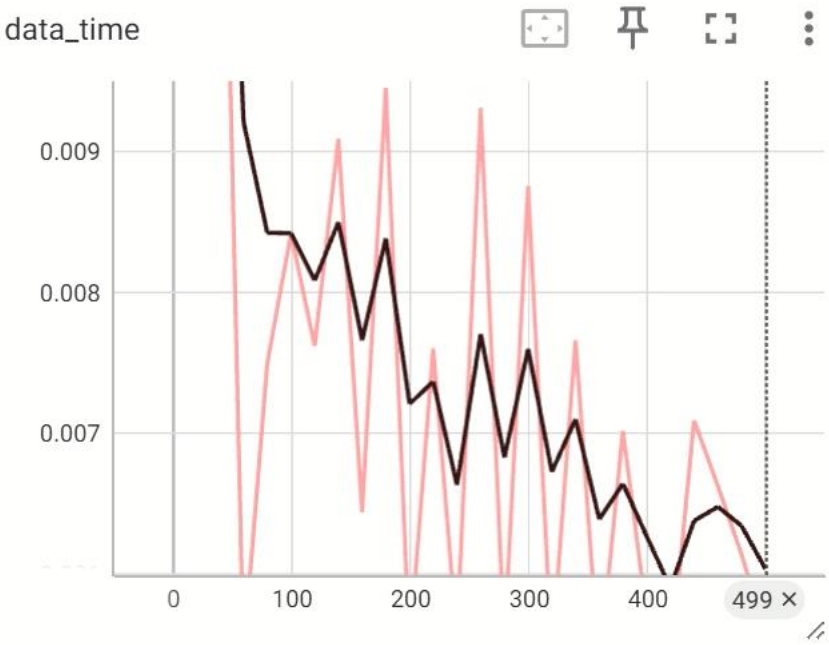
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



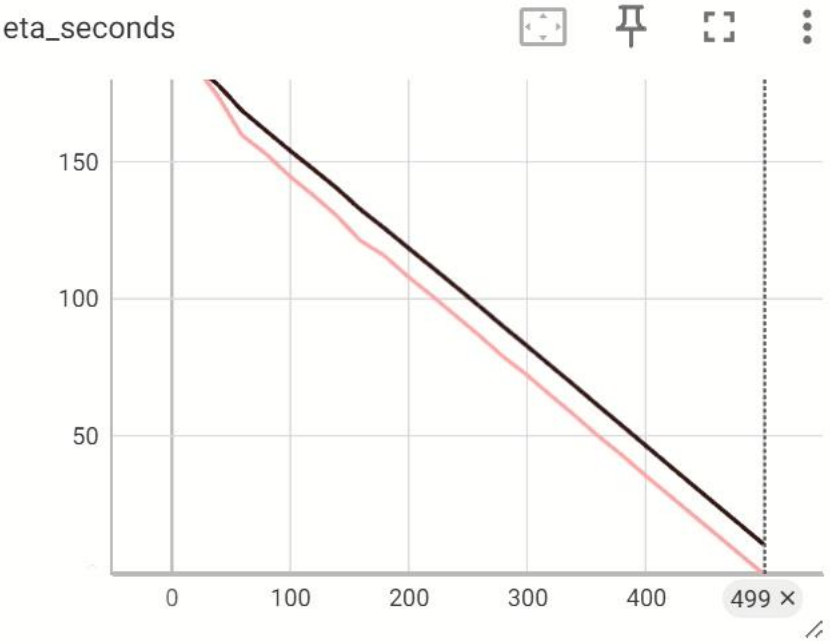


Name: PRANAY GORANTLA
REG.NO: 21MIS0123

Faster RCNN:



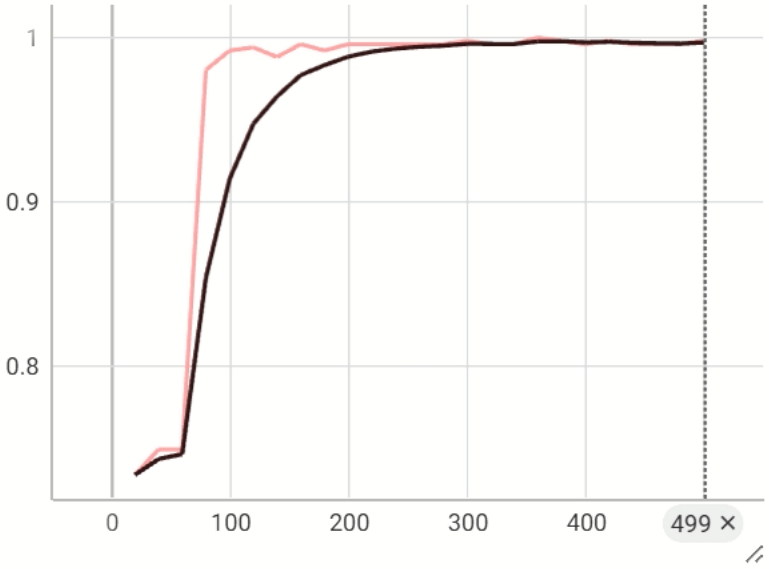
Run ↑	Smoothed	Value	Step	Relative
.	0.006	0.0056	499	2.948 min



Run ↑	Smoothed	Value	Step	Relative
.	10.8303	0	499	2.948 min

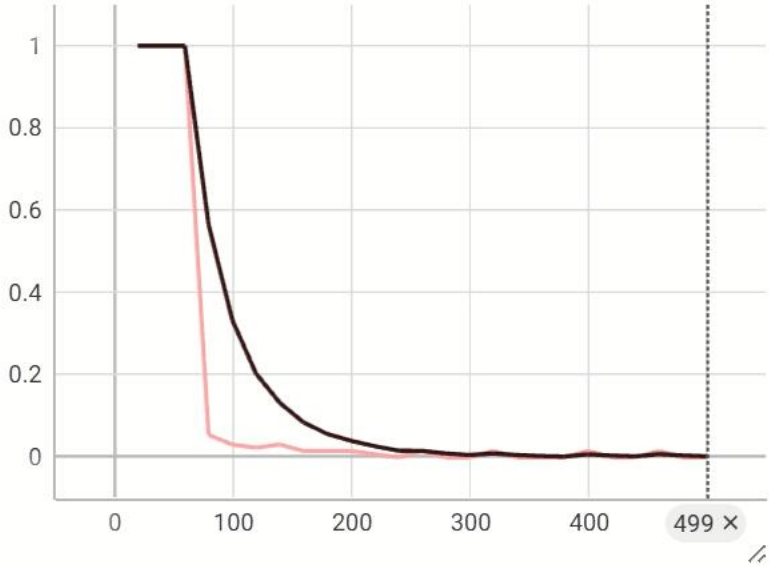
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

fast_rcnn/cls_accuracy



Run ↑	Smoothed	Value	Step	Relative
.	0.9971	0.998	499	2.948 min

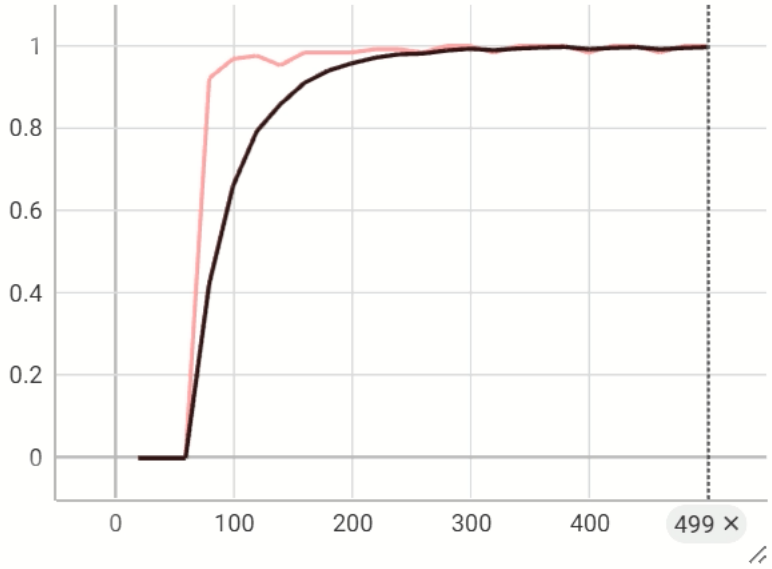
fast_rcnn/false_negative



Run ↑	Smoothed	Value	Step	Relative
.	0.0028	0	499	2.948 min

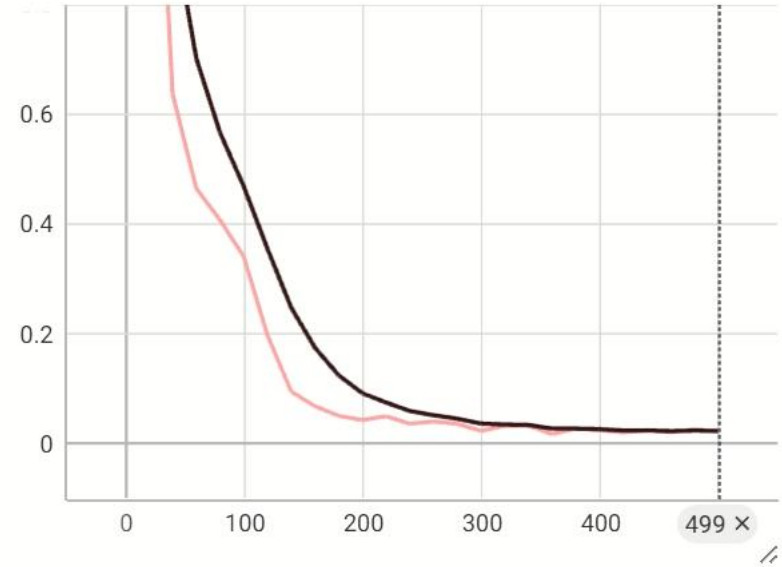
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

fast_rcnn/fg_cls_accuracy



Run ↑	Smoothed	Value	Step	Relative
.	0.9972	1	499	2.948 min

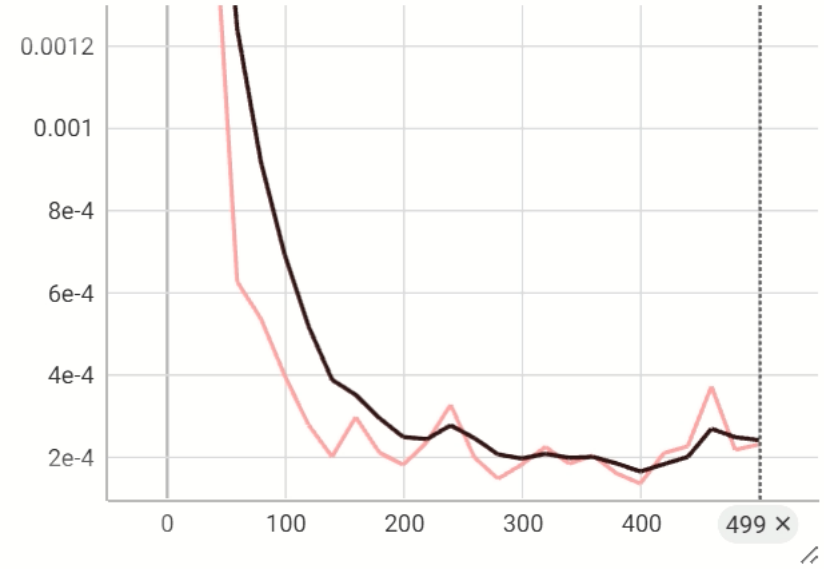
loss_cls



Run ↑	Smoothed	Value	Step	Relative
.	0.0246	0.0236	499	2.948 min

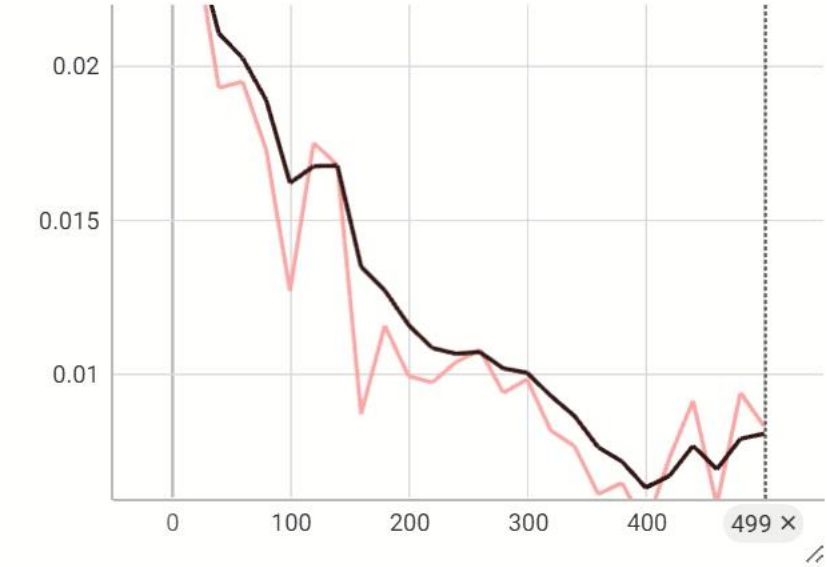
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

loss_rpn_cls



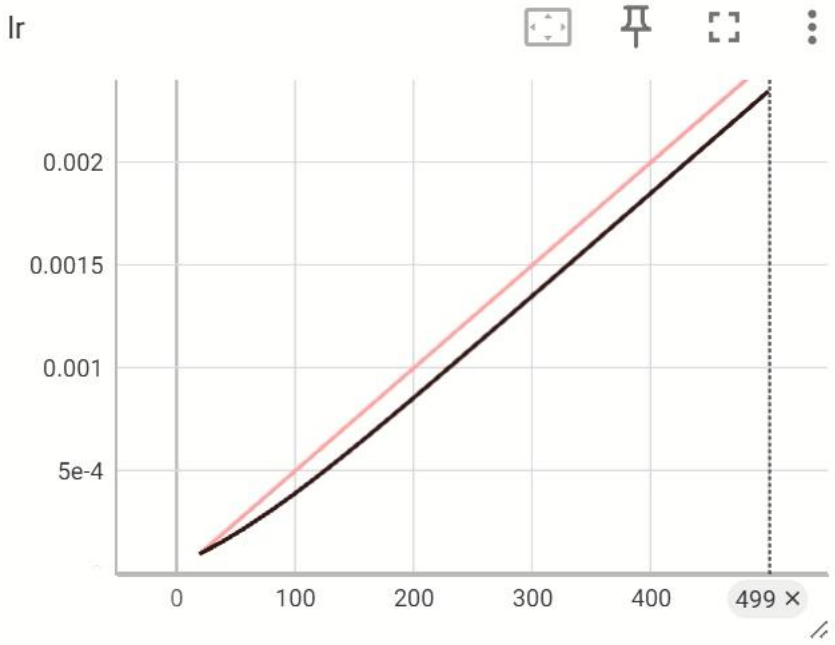
Run ↑	Smoothed	Value	Step	Relative
.	0.0002	0.0002	499	2.948 min

loss_rpn_loc

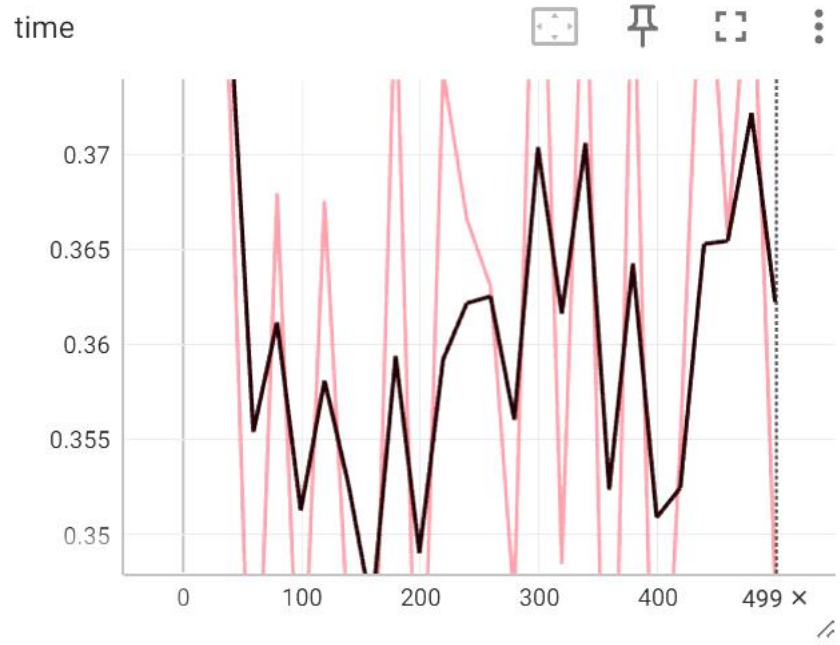


Run ↑	Smoothed	Value	Step	Relative
.	0.0081	0.0083	499	2.948 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

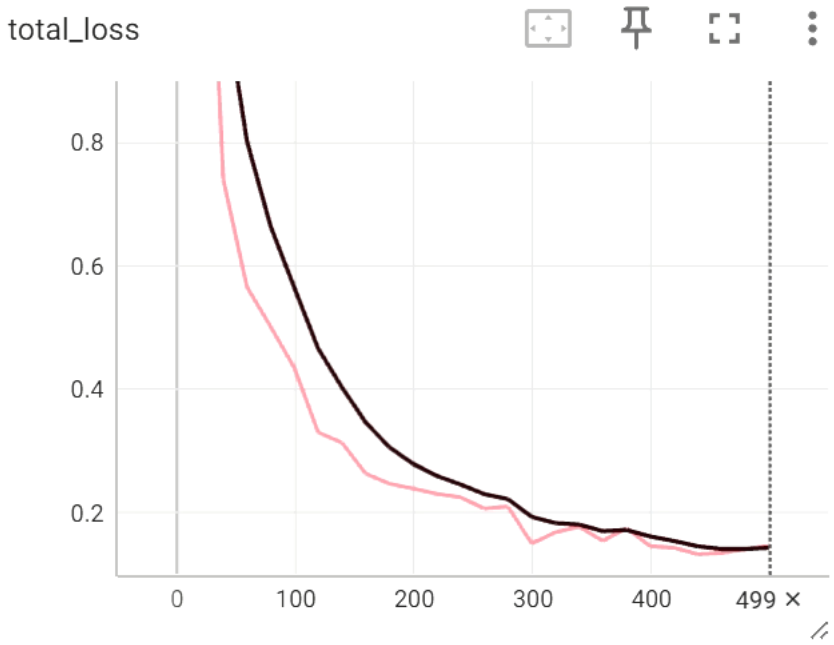


Run ↑	Smoothed	Value	Step	Relative
.	0.0023	0.0025	499	2.948 min



Run ↑	Smoothed	Value	Step	Relative
.	0.3623	0.3473	499	2.948 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123



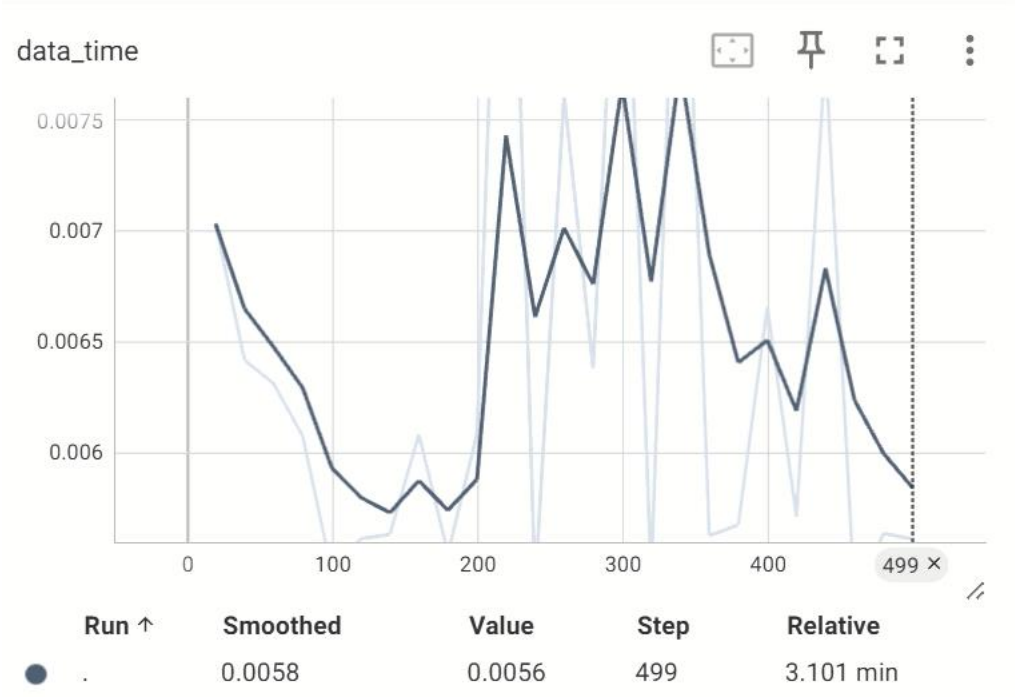
Run ↑	Smoothed	Value	Step	Relative
.	0.1442	0.1473	499	2.948 min

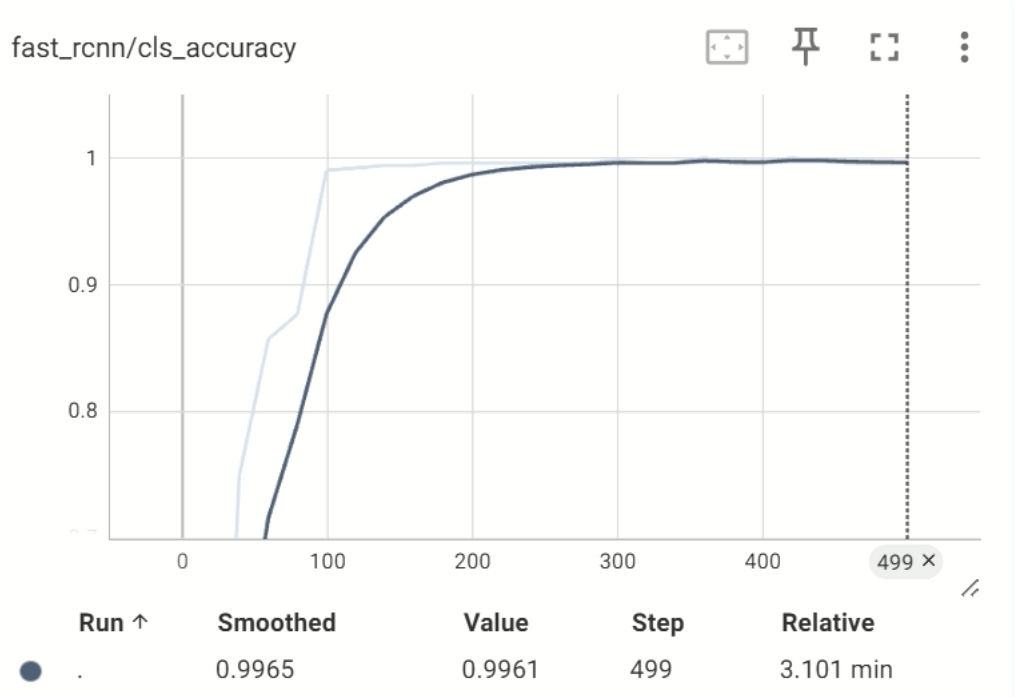
```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.004
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.009
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.005
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.005
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.005
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.005
[01/16 07:55:46 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|  AP   |  AP50  |  AP75  |  APs  |  APm  |  APl  |
|:-----:|:------:|:------:|:-----:|:-----:|:-----:|
| 0.446 | 0.891  | 0.000  |  nan  |  nan  | 0.446 |
[01/16 07:55:46 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN.
[01/16 07:55:46 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category   | AP     | category   | AP     | category   | AP     |
|:-----------|:-------|:-----------|:-------|:-----------|:-------|
|            | 1.485  |            | 0.000  |            | 0.000  |
|            | 0.000  |            | 0.743  |            |        |
```

```
OrderedDict([('bbox', {'AP': 0.4455445544554455, 'AP50': 0.891089108910891, 'AP75': 0.0, 'APs': nan, 'APm': nan,
'APs': nan, 'APm': nan, 'APl': 0.4455445544554455, 'AP-': 0.7425742574257426})])
```

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

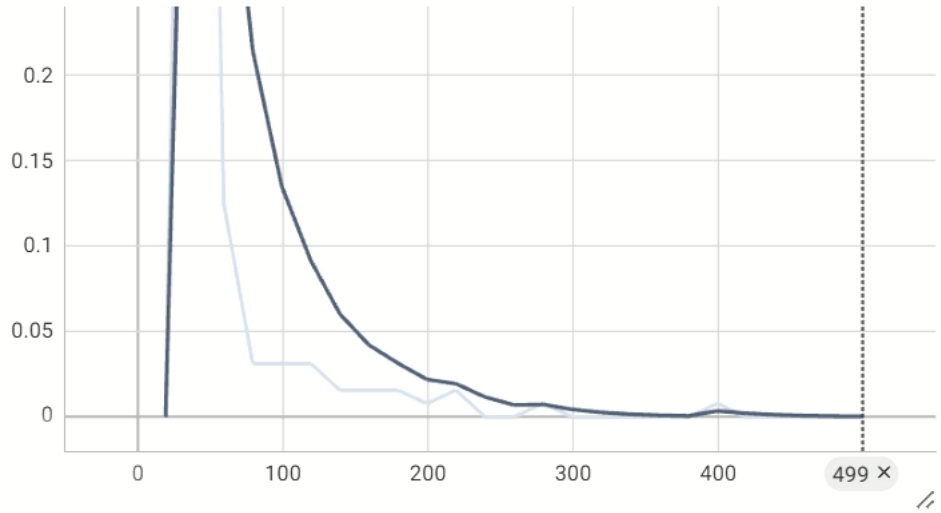
MASK RCNN:





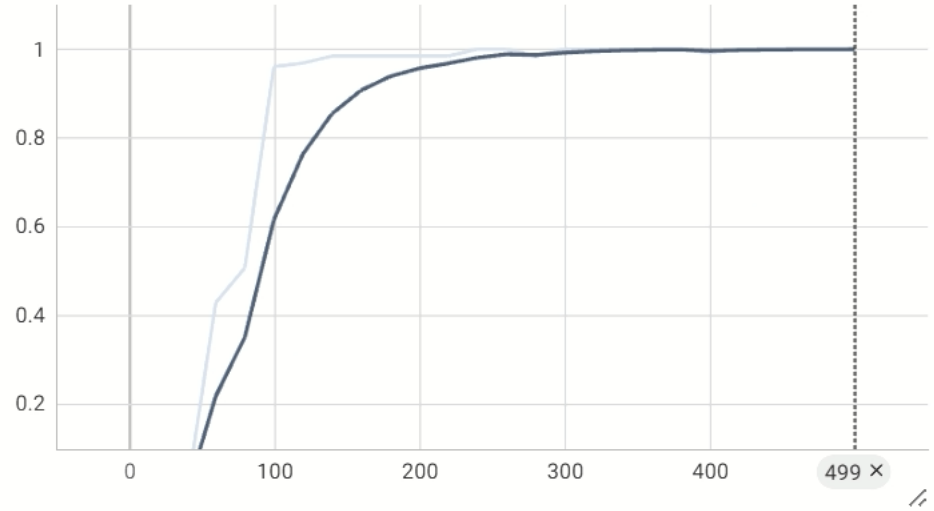
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

fast_rcnn/false_negative



Run ↑	Smoothed	Value	Step	Relative
.	0.0003	0	499	3.101 min

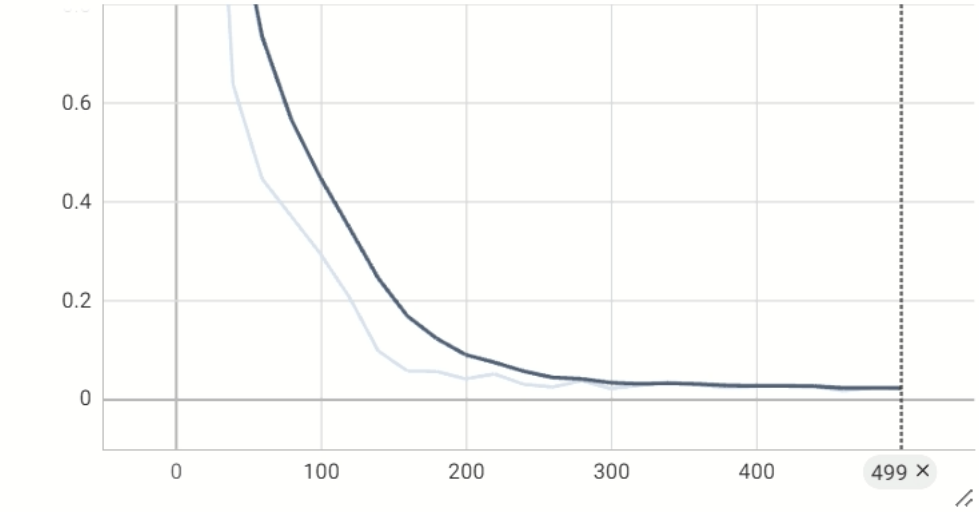
fast_rcnn/fg_cls_accuracy



Run ↑	Smoothed	Value	Step	Relative
.	0.9997	1	499	3.101 min

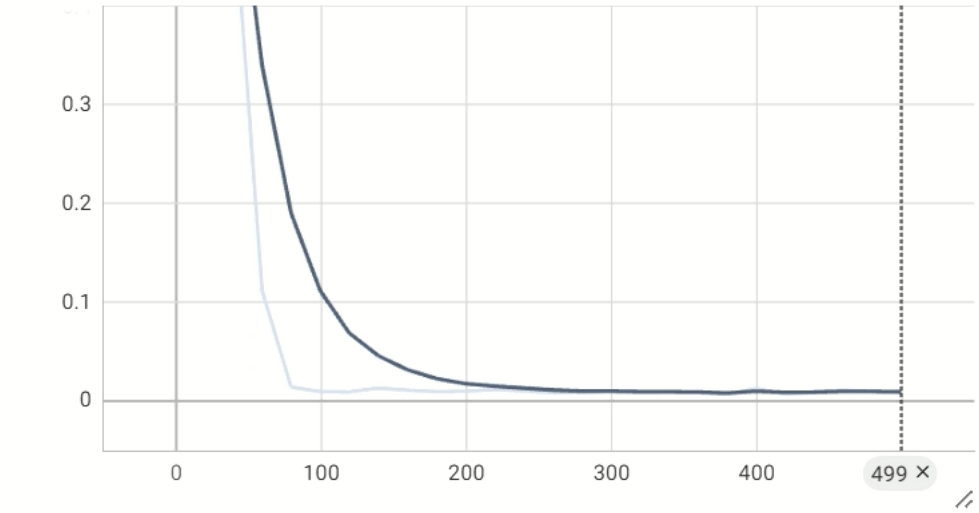
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

loss_cls



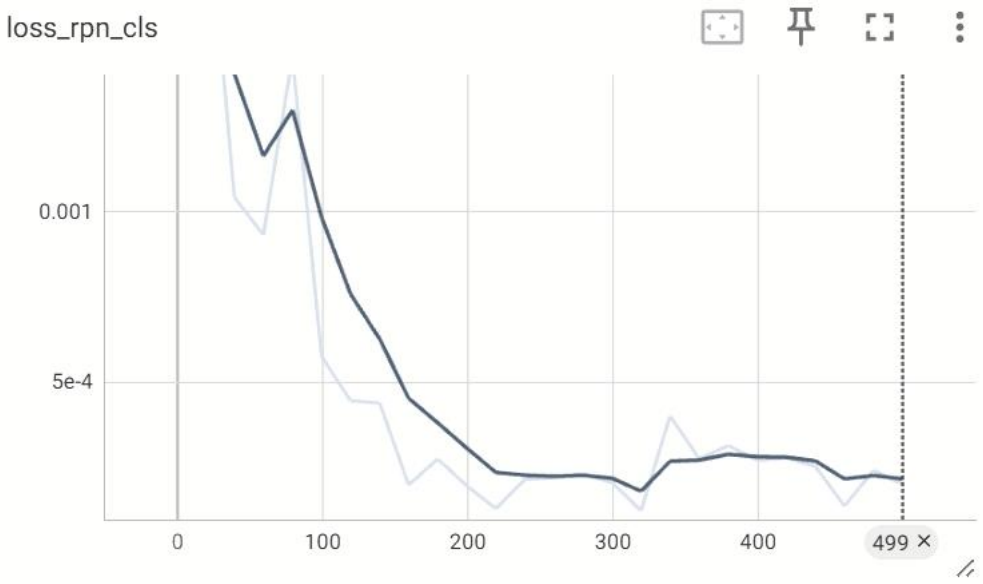
Run ↑	Smoothed	Value	Step	Relative
.	0.0238	0.0235	499	3.101 min

loss_mask

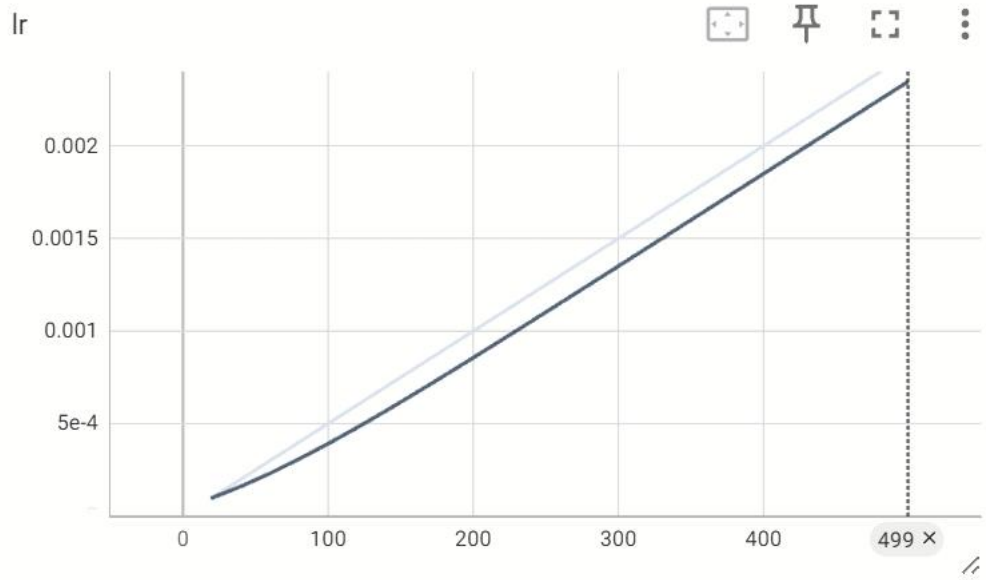


Run ↑	Smoothed	Value	Step	Relative
.	0.0094	0.0088	499	3.101 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123



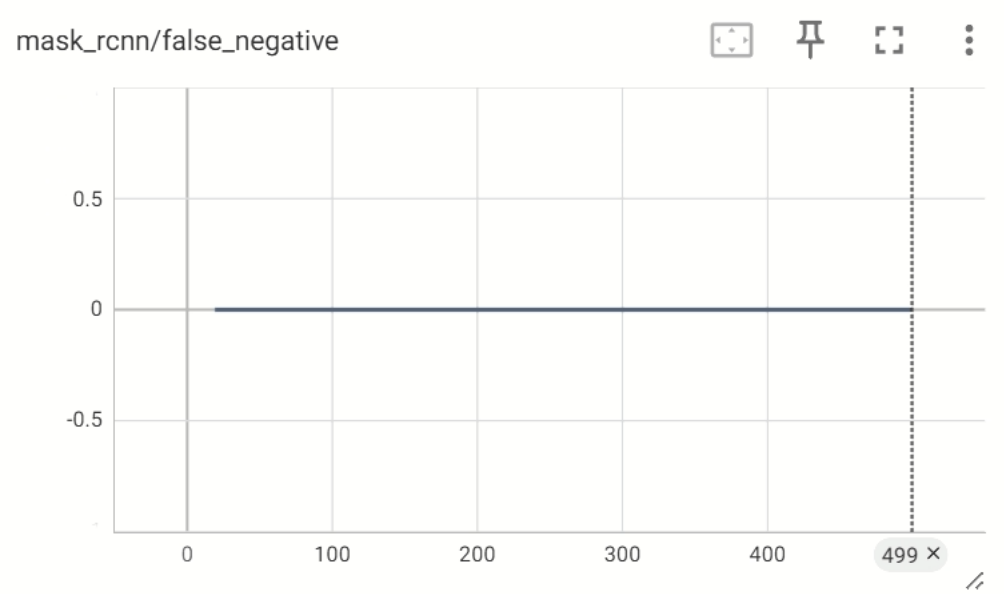
Run ↑	Smoothed	Value	Step	Relative
.	0.0002	0.0002	499	3.101 min



Run ↑	Smoothed	Value	Step	Relative
.	0.0023	0.0025	499	3.101 min

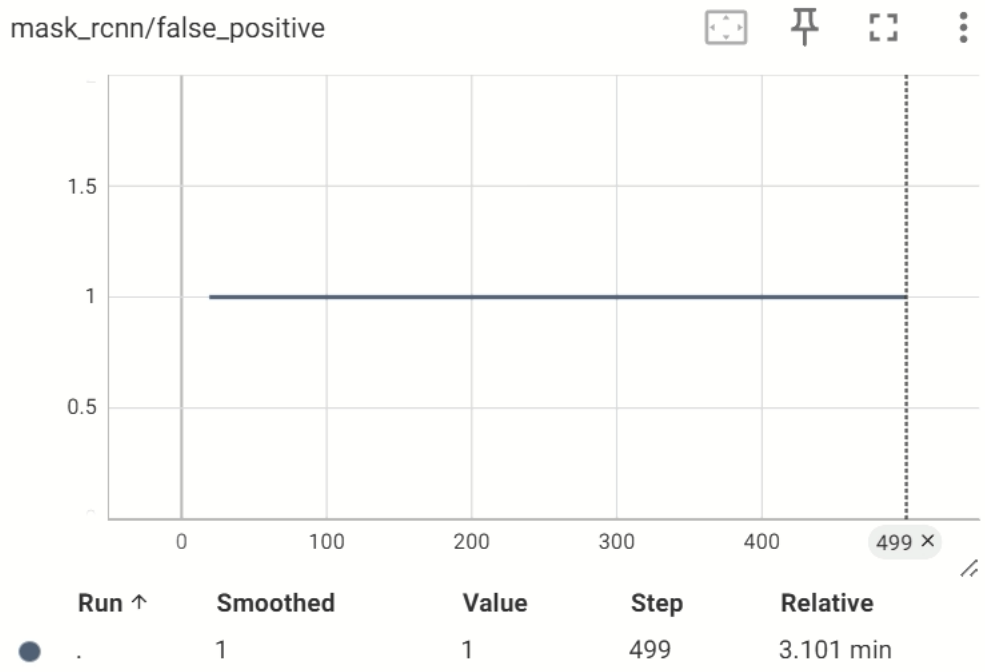
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

Run ↑	Smoothed	Value	Step	Relative
.	0.9981	0.9983	499	3.101 min

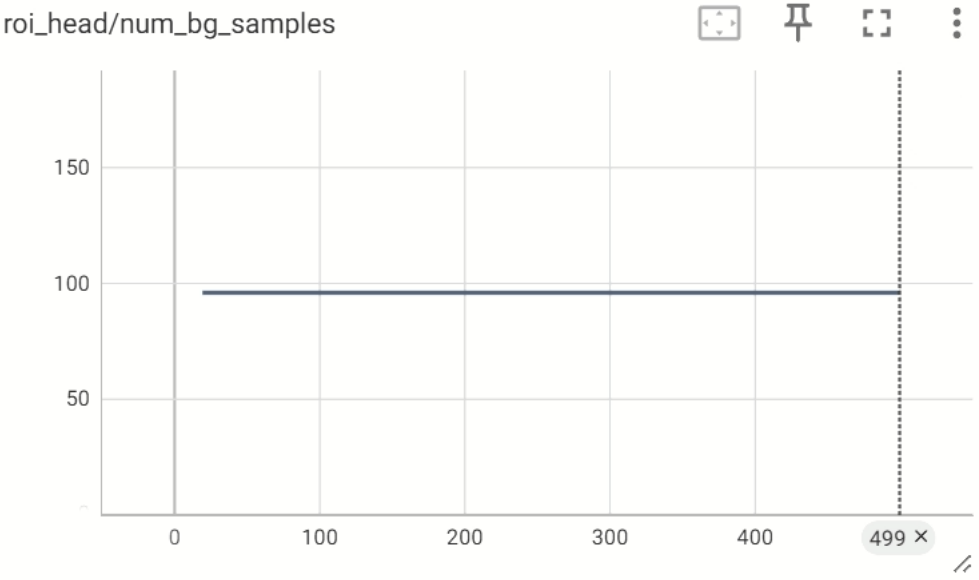


Run ↑	Smoothed	Value	Step	Relative
.	0	0	499	3.101 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

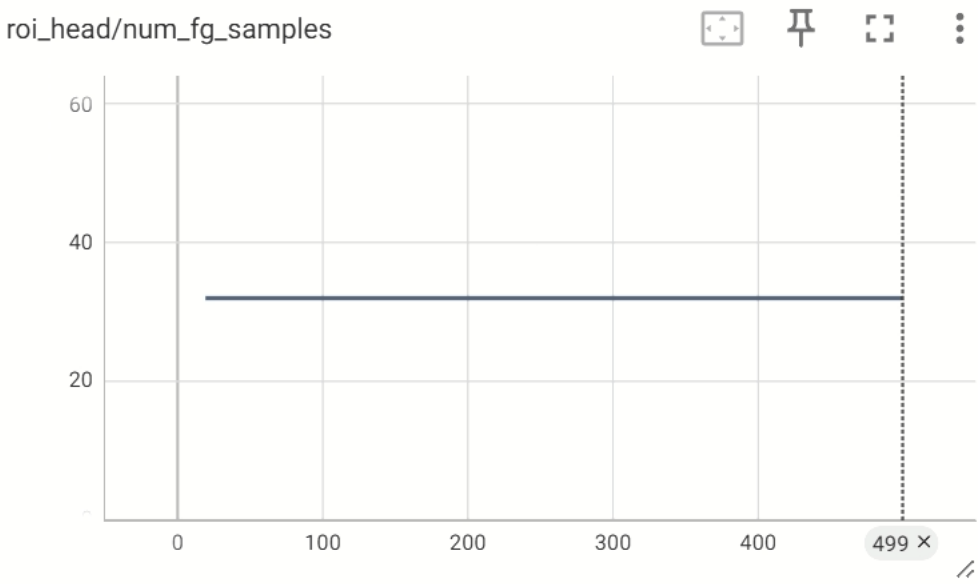


Run ↑	Smoothed	Value	Step	Relative
.	1	1	499	3.101 min

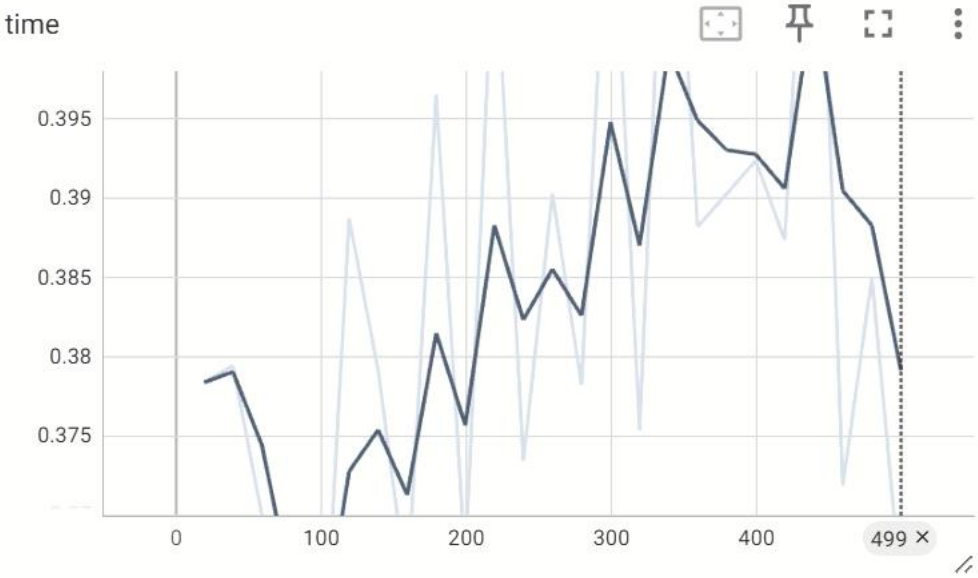


Run ↑	Smoothed	Value	Step	Relative
.	96	96	499	3.101 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

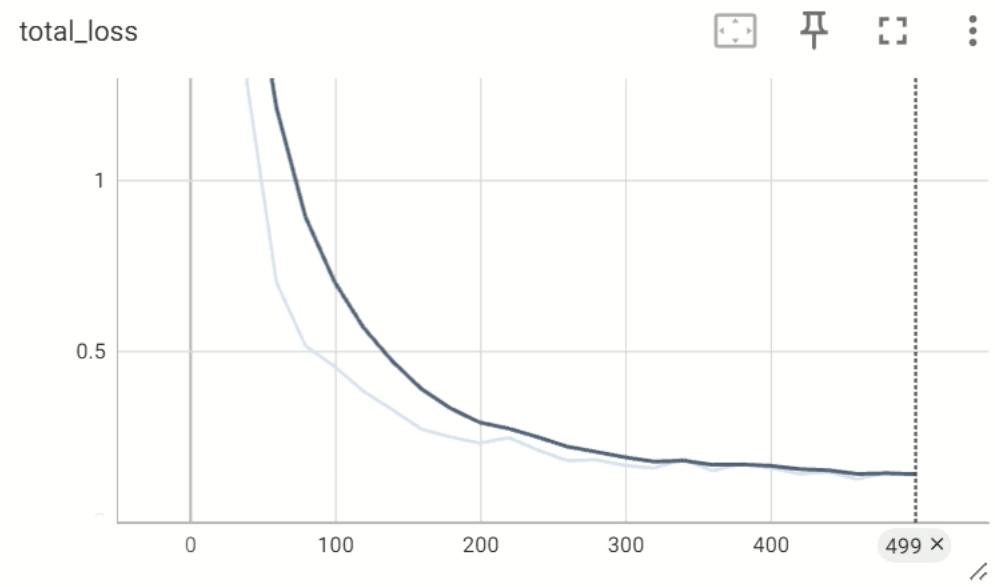


Run	Smoothed	Value	Step	Relative
.	32	32	499	3.101 min



Run	Smoothed	Value	Step	Relative
.	0.3792	0.3655	499	3.101 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123



Run	Smoothed	Value	Step	Relative
.	0.1424	0.139	499	3.101 min

```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.002
 Average Precision  (AP) @[ IoU=0.50       | area=   all | maxDets=100 ] = 0.003
 Average Precision  (AP) @[ IoU=0.75       | area=   all | maxDets=100 ] = 0.003
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.002
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.004
[01/16 09:48:46 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|  AP   |  AP50  |  AP75  |  APs  |  APm  |  APl  |
|:-----:|:------:|:------:|:-----:|:-----:|:-----:|
| 0.238 | 0.297  | 0.297  |  nan  |  nan  | 0.238 |
```

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

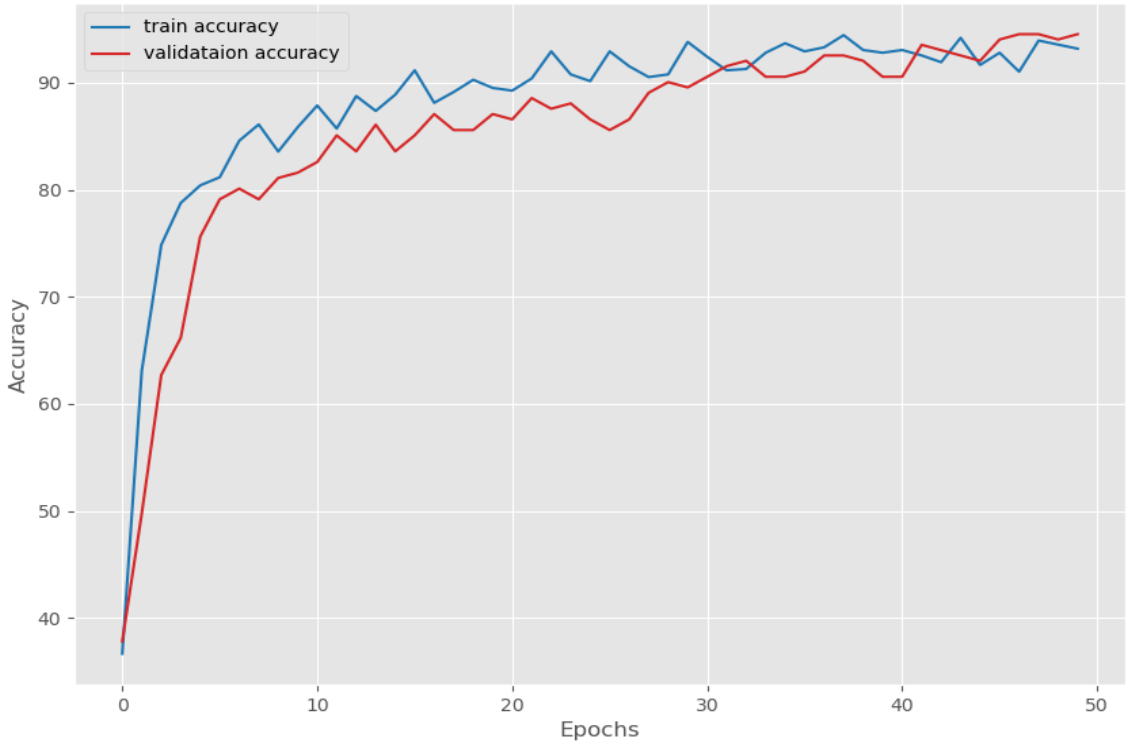
```
[01/16 09:48:46 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|  AP   |  AP50  |  AP75  |  APs  |  APm  |  APl  |
|:-----:|:------:|:------:|:-----:|:-----:|:-----:|
| 0.238 | 0.297  | 0.297  |  nan  |  nan  | 0.238 |
[01/16 09:48:46 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN.
[01/16 09:48:46 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category   | AP     | category   | AP     | category   | AP     |
|:-----------|:-------|:-----------|:-------|:-----------|:-------|
|            | 0.000  |            | 0.000  |            | 0.000  |
|            | 0.000  |            | 1.188  |            |        |
```

```
[01/16 09:48:46 d2.evaluation.fast_eval_api]: Evaluate annotation type *segm*
[01/16 09:48:46 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.02 seconds.
[01/16 09:48:46 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[01/16 09:48:46 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.01 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.002
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.003
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.003
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.002
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.004
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.004
[01/16 09:48:47 d2.evaluation.coco_evaluation]: Evaluation results for segm:
|  AP   |  AP50  |  AP75  |  APs  |  APm  |  APl  |
|:-----:|:------:|:------:|:-----:|:-----:|:-----:|
| 0.238 | 0.297  | 0.297  |  nan  |  nan  | 0.238 |
[01/16 09:48:47 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN.
[01/16 09:48:47 d2.evaluation.coco_evaluation]: Per-category segm AP:
| category   | AP     | category   | AP     | category   | AP     |
|:-----------|:-------|:-----------|:-------|:-----------|:-------|
|            | 0.000  |            | 0.000  |            | 0.000  |
|            | 0.000  |            | 1.188  |            |        |
OrderedDict([('bbox', {'AP': 0.2376237623762376, 'AP50': 0.297029702970297, 'AP75': 0.297029702970297, 'APs': nan, 'APm': nan, 'APl
```

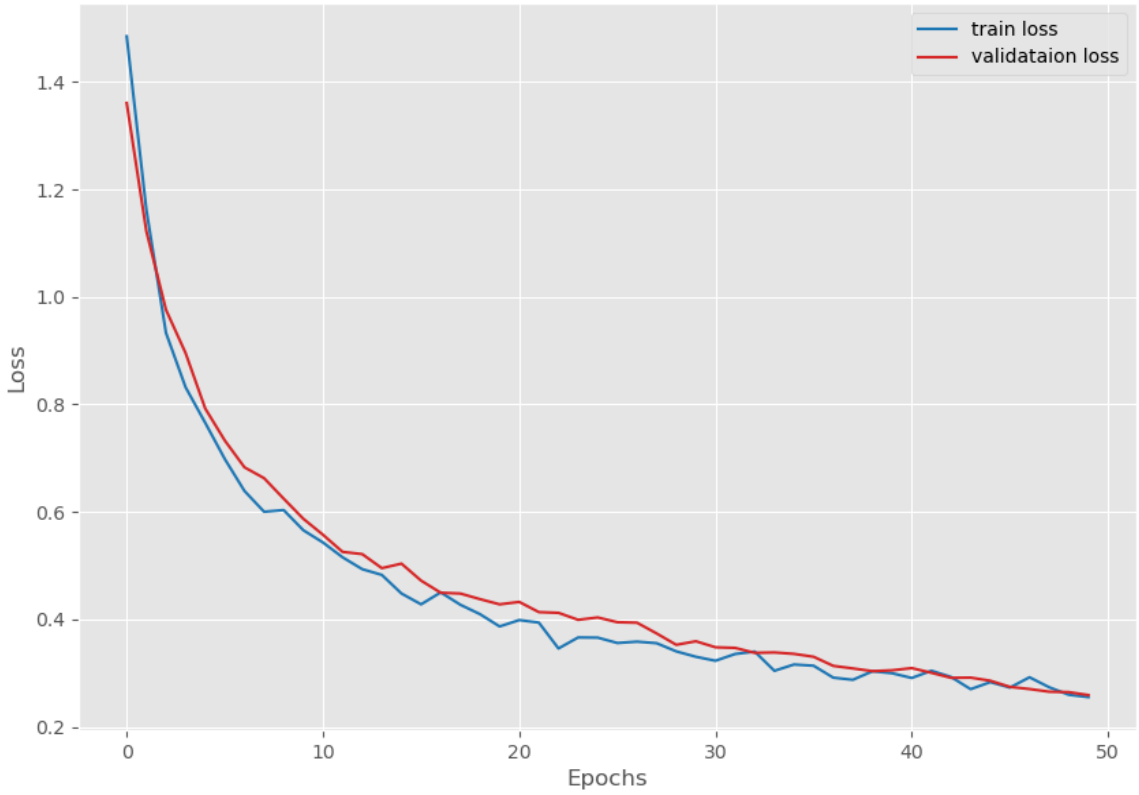
```
'APl': 0.2376237623762376, 'AP-': 1.188118811881188}), ('segm', {'AP': 0.2376237623762376, 'AP50': 0.297029702970297,
```

```
'AP75': 0.297029702970297, 'APs': nan, 'APm': nan, 'APl': 0.2376237623762376, 'AP-': 1.188118811881188})])
```

SWIN TRANSFORMER:



Name: PRANAY GORANTLA
REG.NO: 21MIS0123



[INFO]: Number of training images: 791

[INFO]: Number of validation images: 201

[INFO]: Classes: ['0hrMutton', '12hrMutton', '24hrMutton', '36hrMutton', '48hrMutton']

Computation device: cuda

Learning rate: 0.001

Epochs to train for: 50

Training loss: 0.256, training acc: 93.173

Validation loss: 0.259, validation acc: 94.527

Best validation loss: 0.2593913104917322

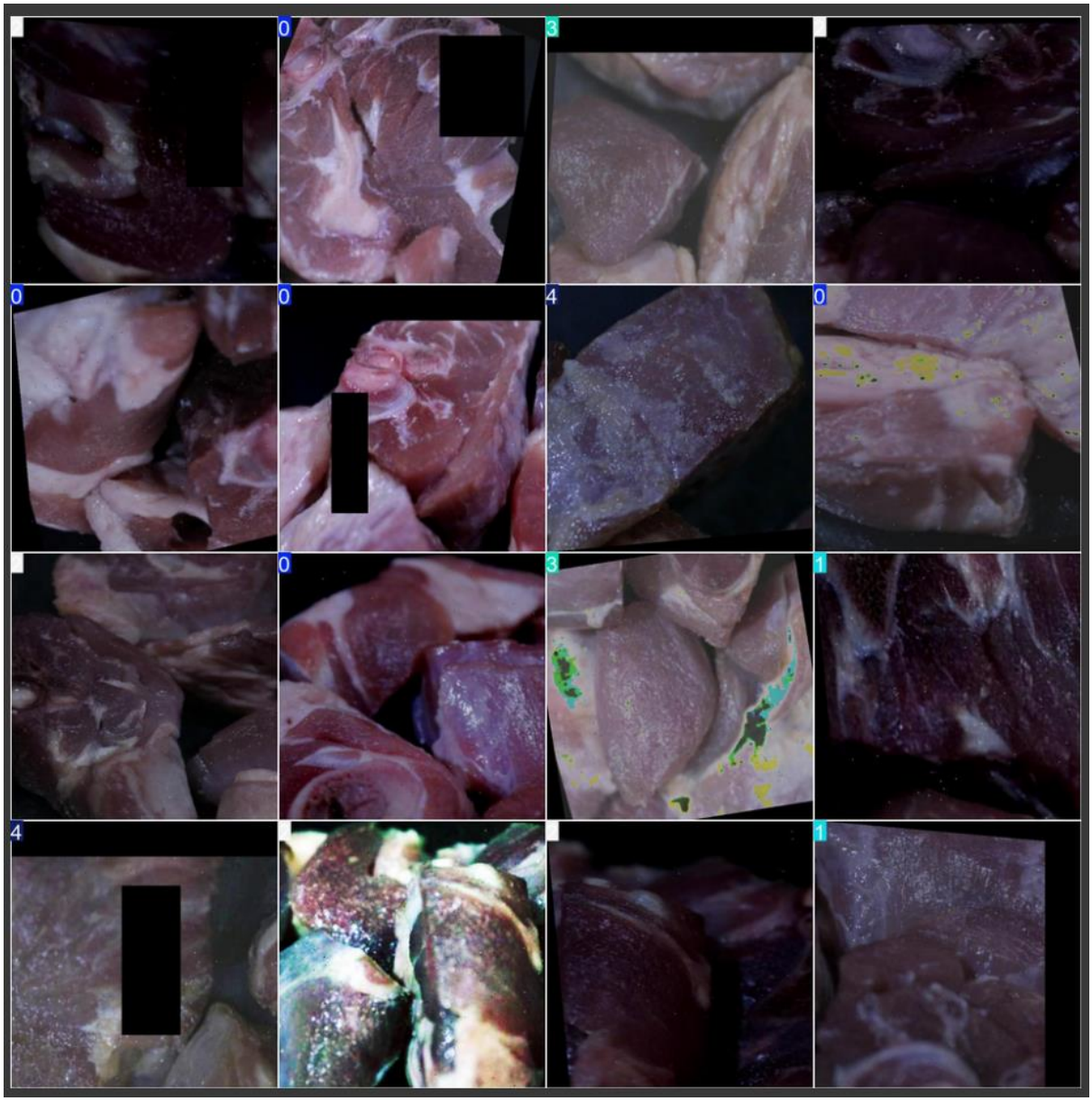
Saving best model for epoch: 50

Number of classes in the dataset: 5

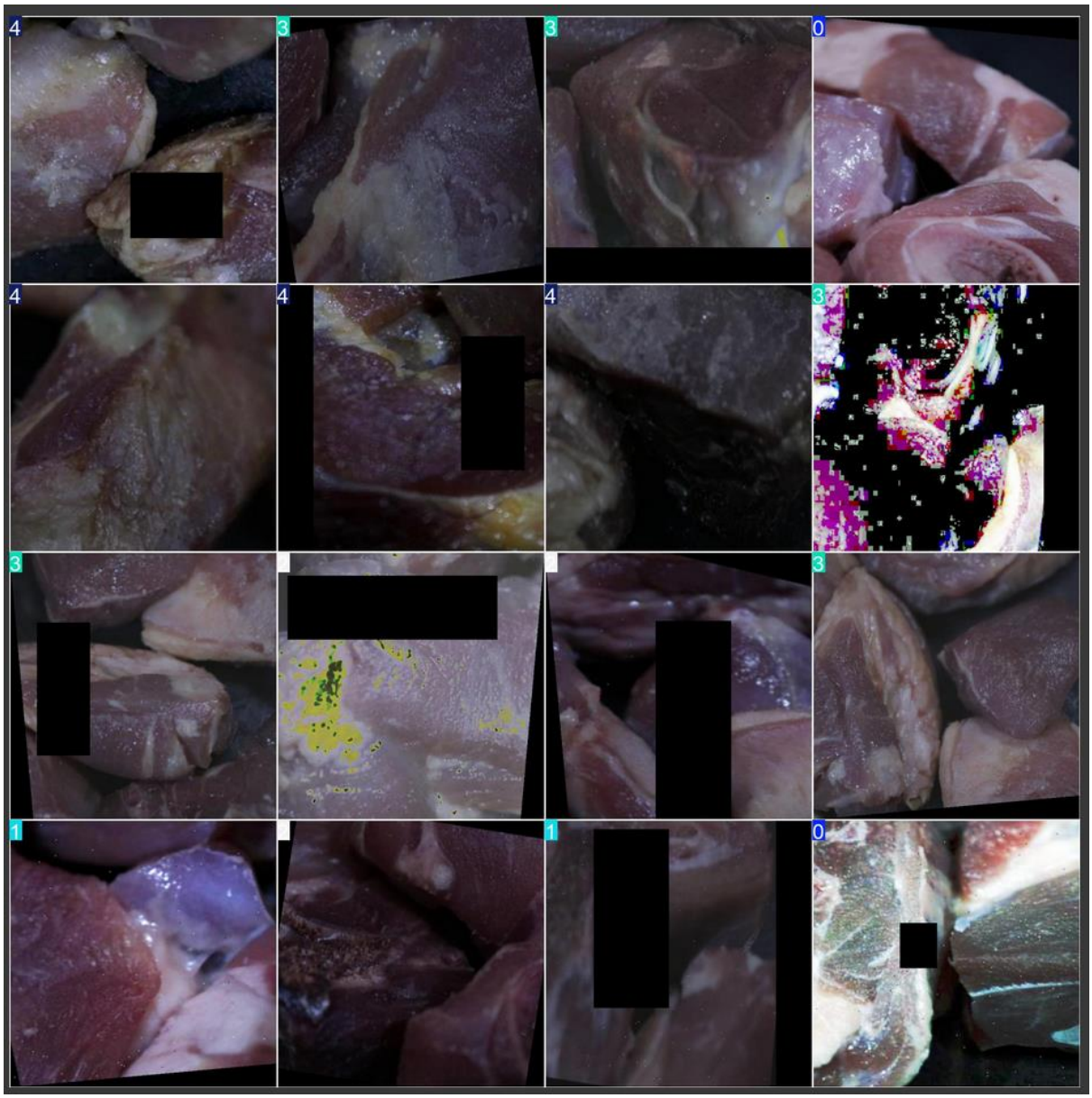
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

Comparision of model performances after data augmentation:

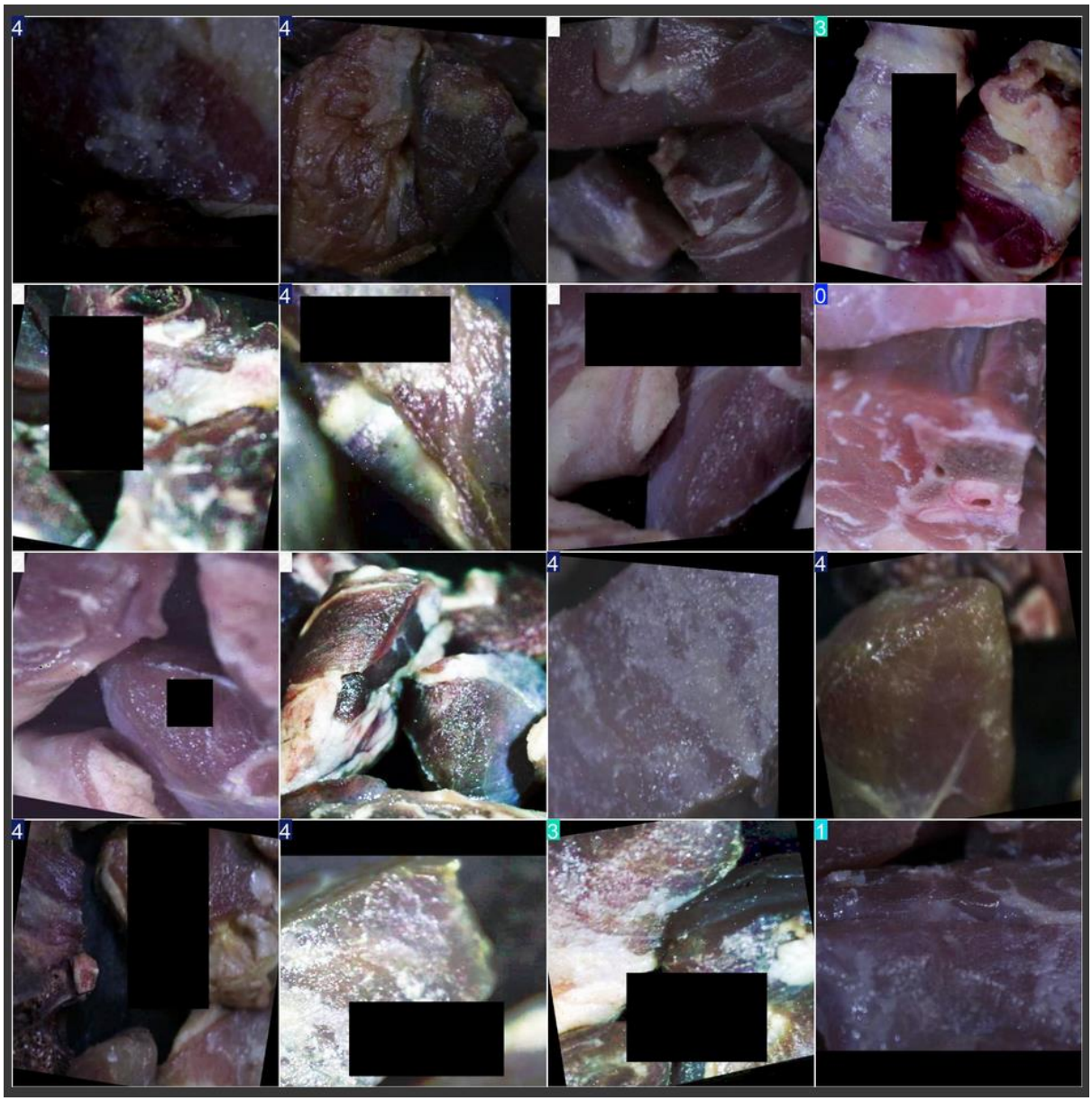
YOLO v11 RESULT:



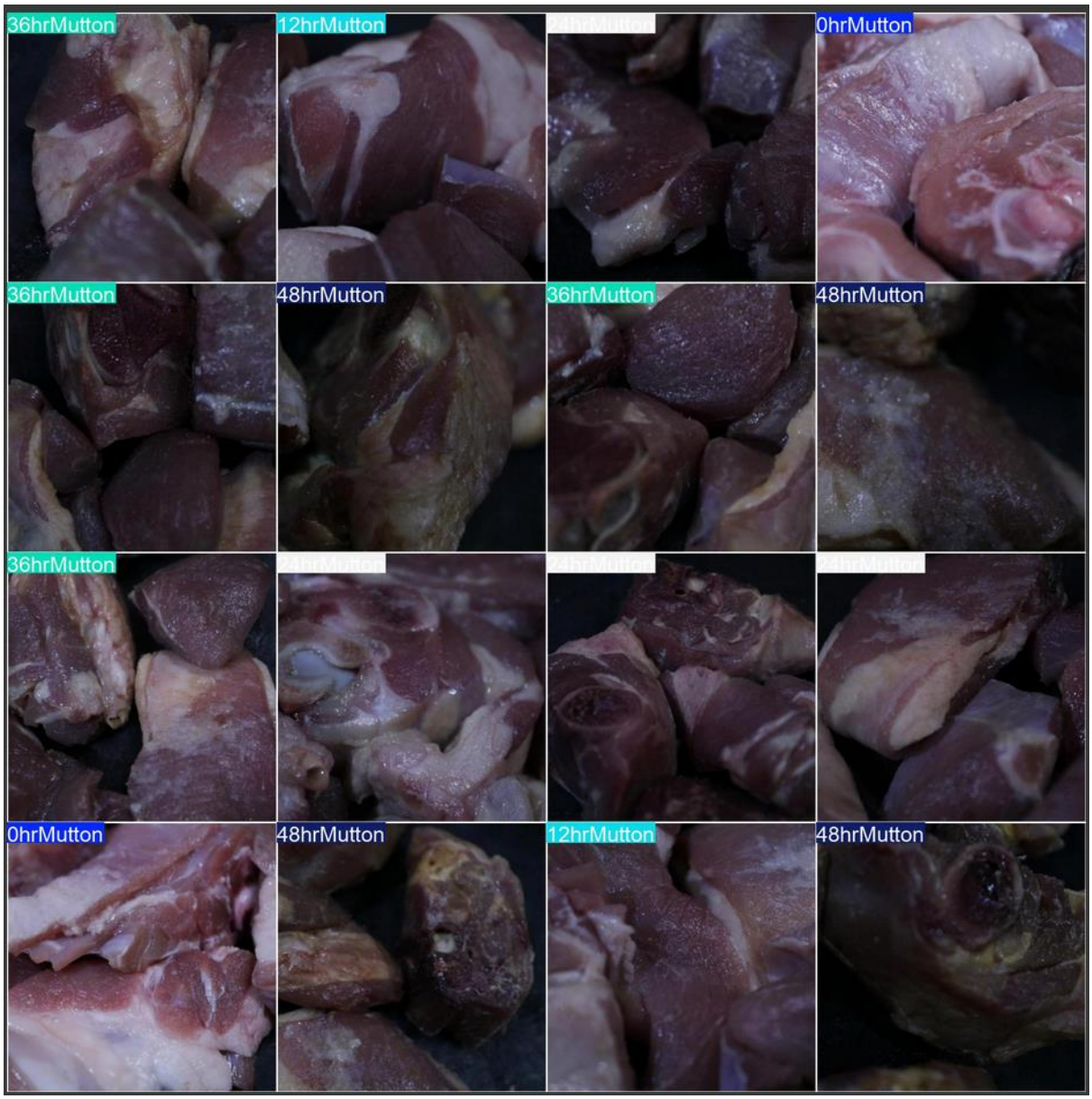
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



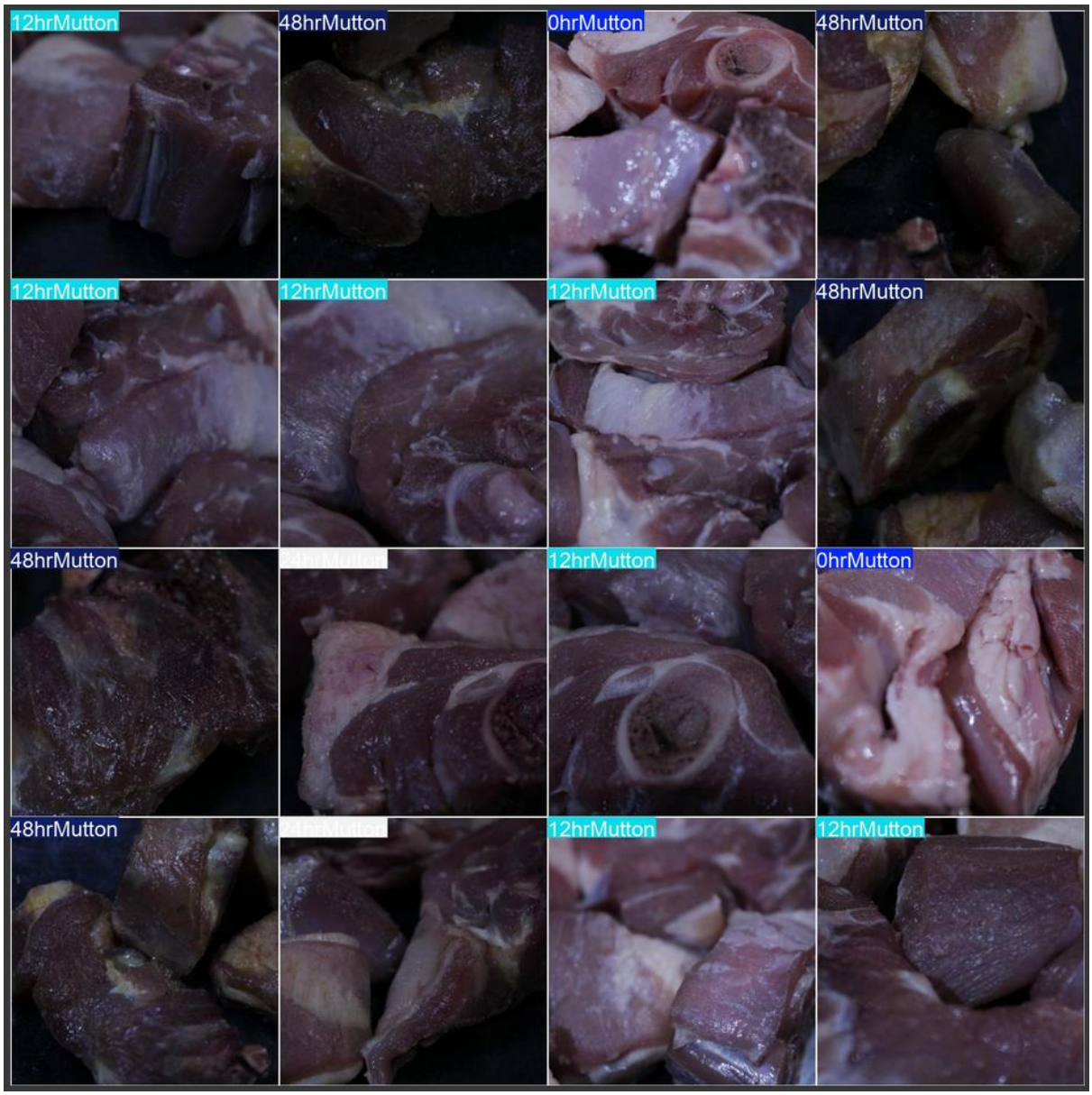
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



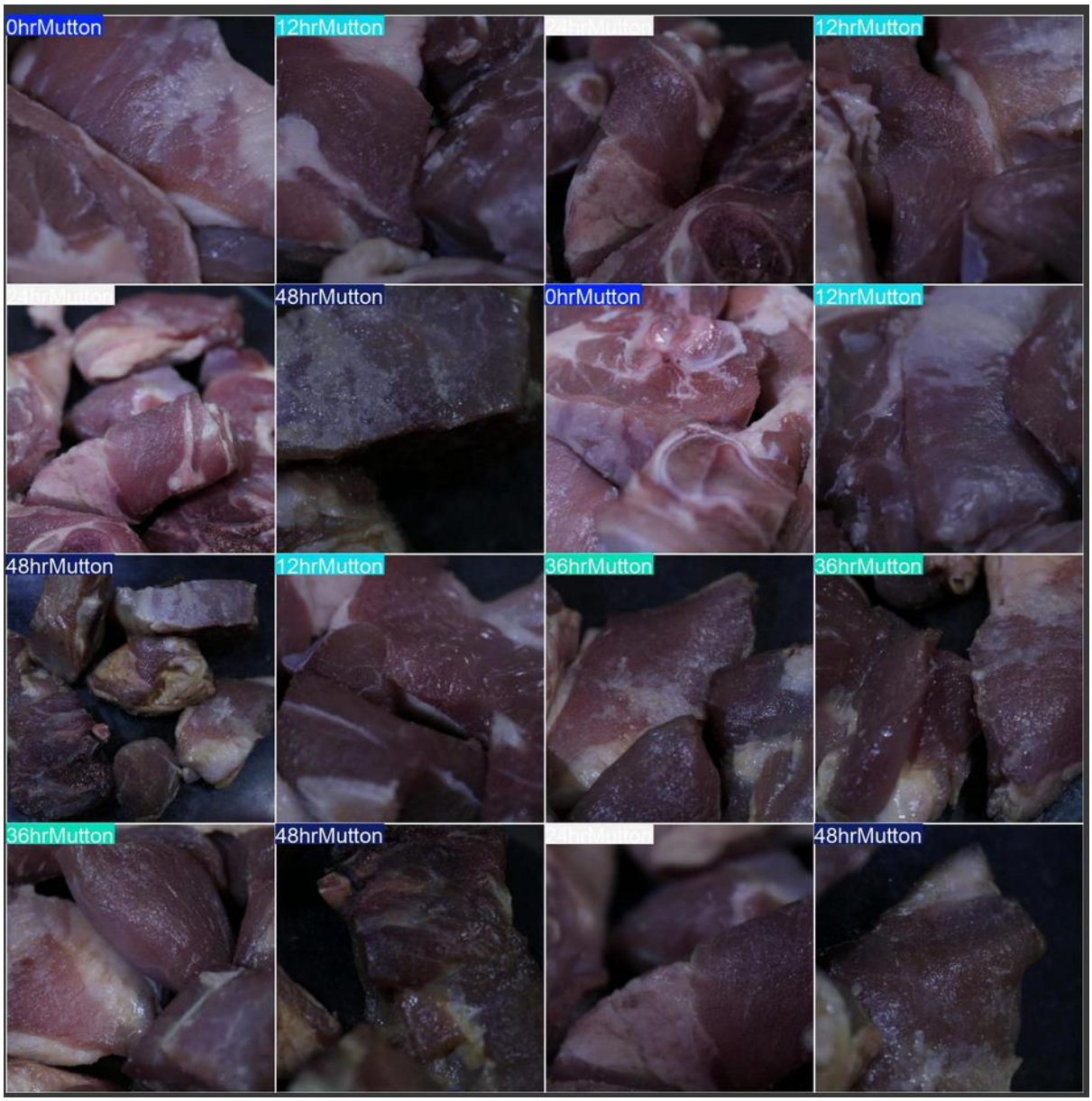
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



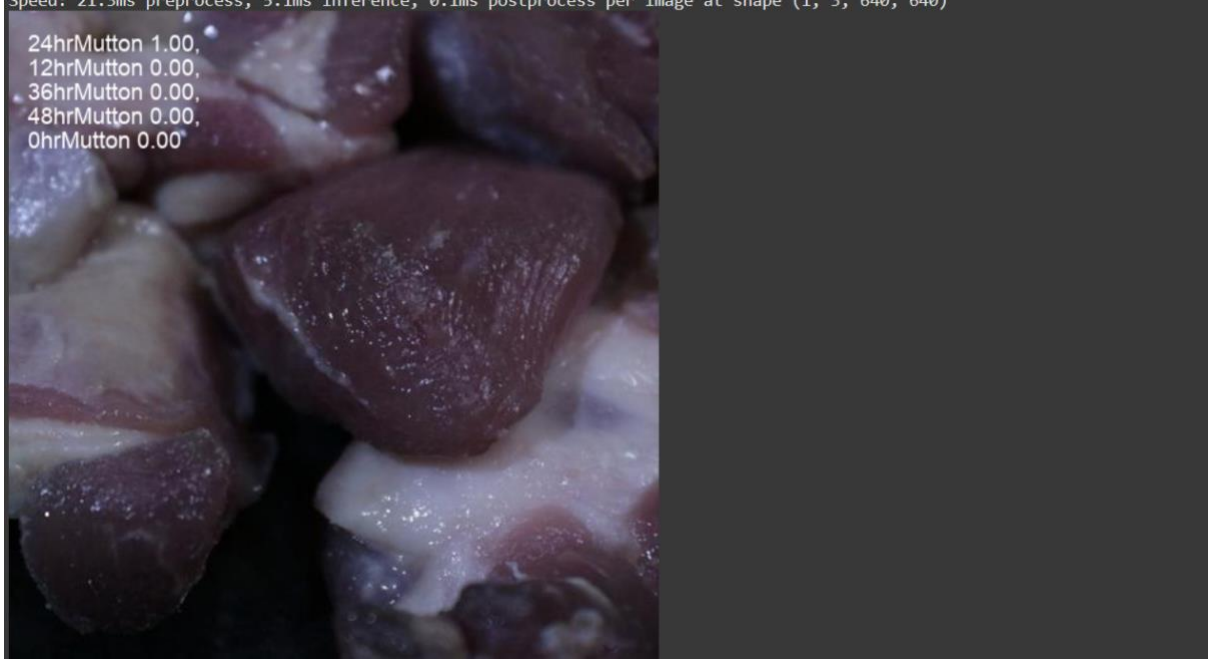
Name: PRANAY GORANTLA
REG.NO: 21MIS0123



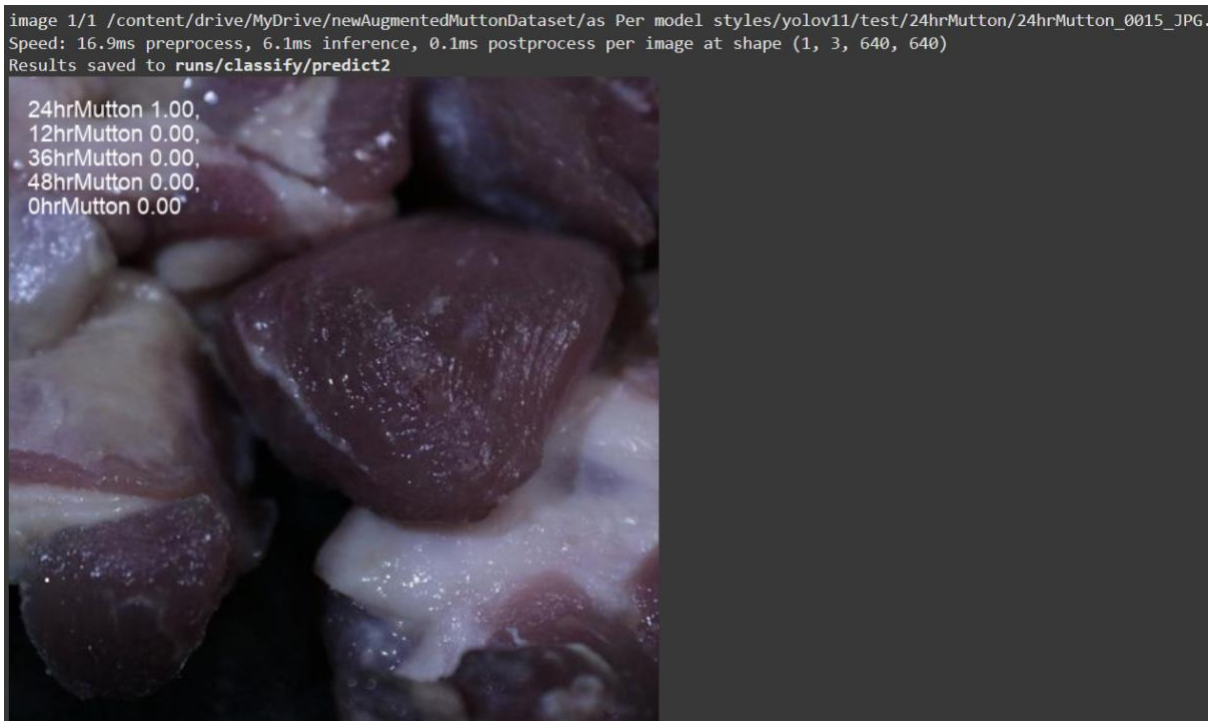
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

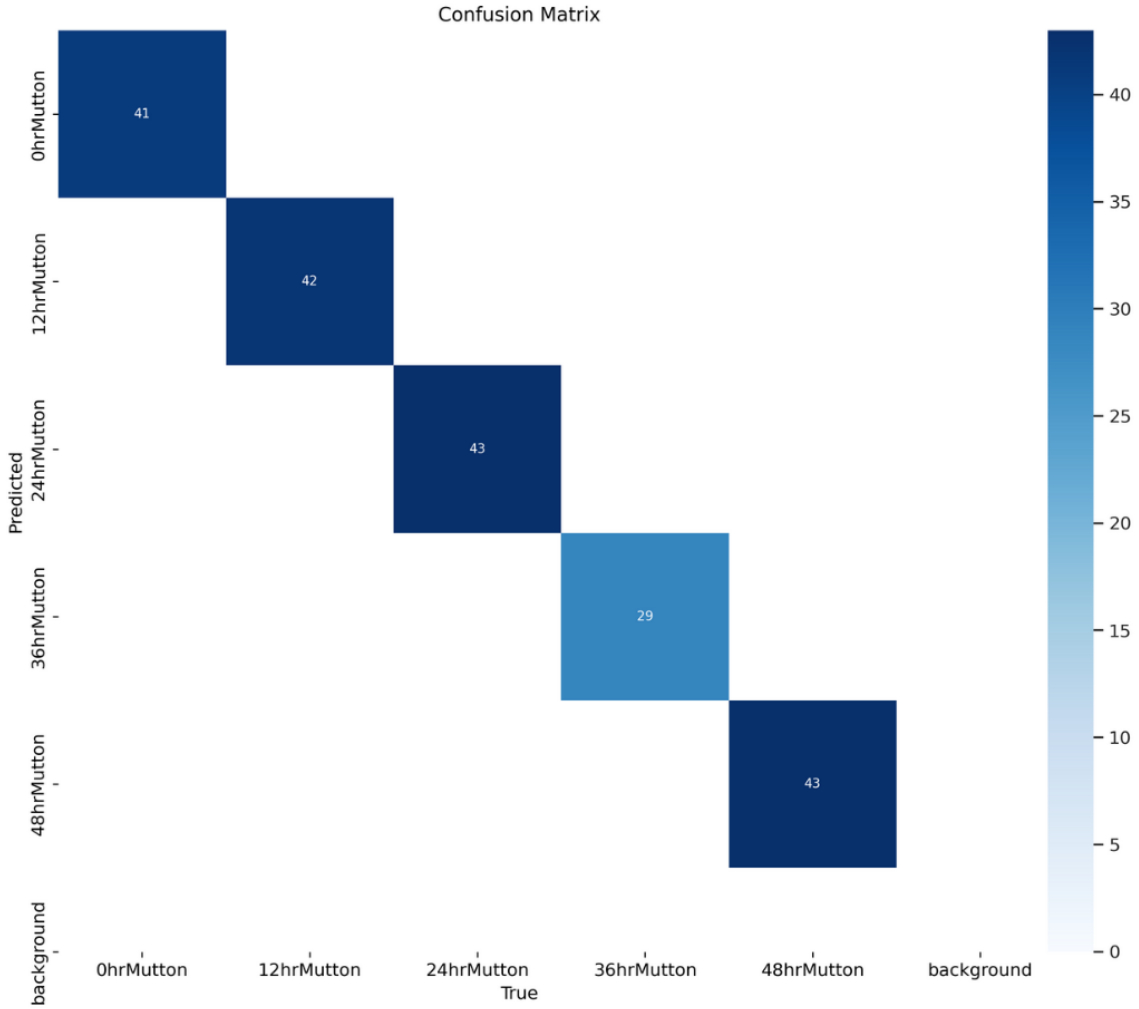



```
image 1/1 /content/drive/MyDrive/newAugmentedMuttonDataset/as Per model styles/yolov11/test/24hrMutton/24hrMutton_0015_JPG.
Speed: 21.3ms preprocess, 5.1ms inference, 0.1ms postprocess per image at shape (1, 3, 640, 640)
```

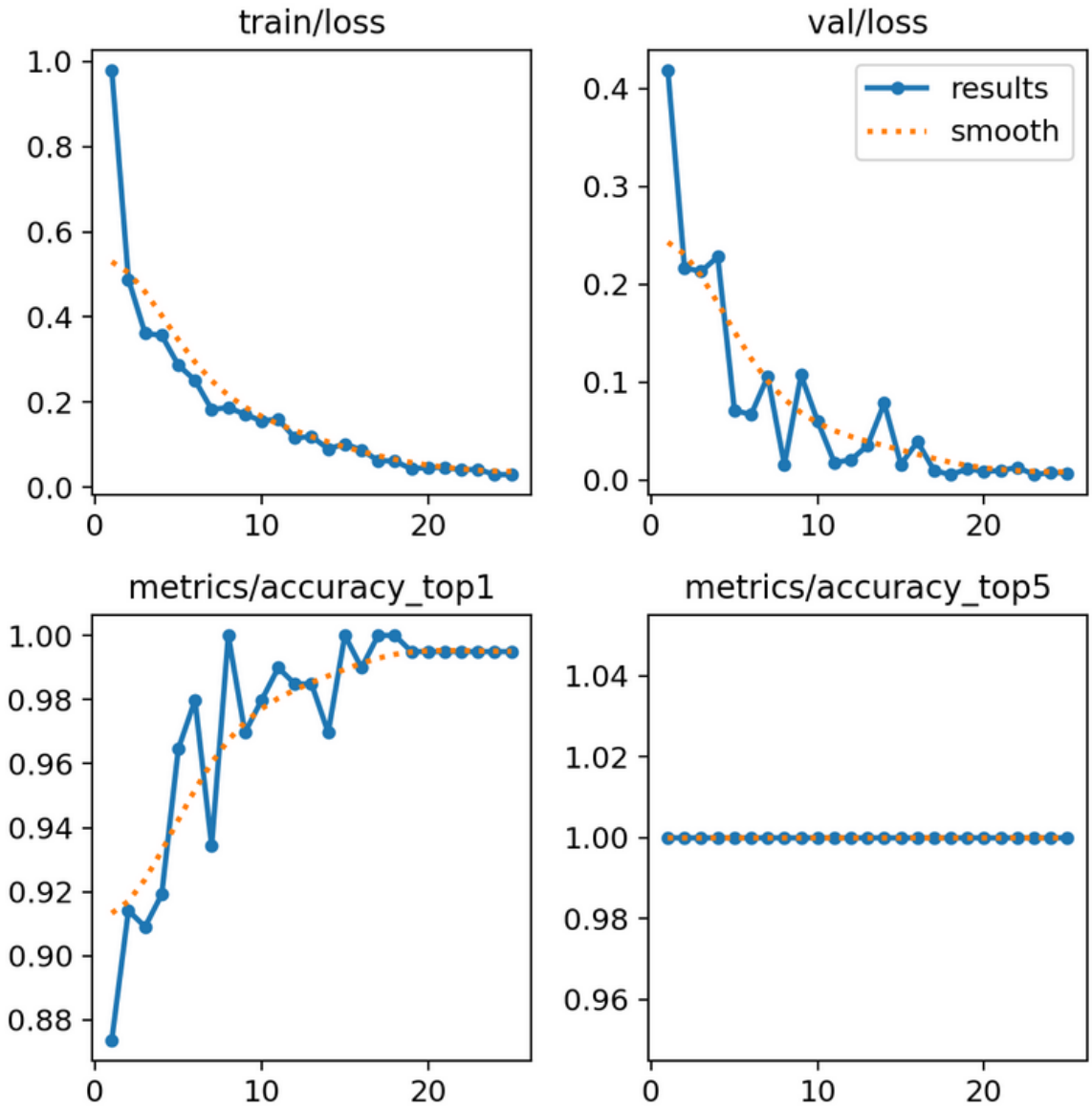


Name: PRANAY GORANTLA
REG.NO: 21MIS0123





Name: PRANAY GORANTLA
REG.NO: 21MIS0123



FASTER RCNN:

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

```
[02/13 17:04:06 d2.data.datasets.coco]: Loaded 1727 images in COCO format from /content/drive/MyDrive/newAugmentedMuttonDataset/as Pe
[02/13 17:04:06 d2.data.build]: Removed 1 images with no usable annotations. 1726 images left.
[02/13 17:04:06 d2.data.dataset_mapper]: [DatasetMapper] Augmentations used in training: [ResizeShortestEdge(short_edge_length=(640,
[02/13 17:04:06 d2.data.build]: Using training sampler TrainingSampler
[02/13 17:04:06 d2.data.common]: Serializing the dataset using: <class 'detectron2.data.common._TorchSerializedList'>
[02/13 17:04:06 d2.data.common]: Serializing 1726 elements to byte tensors and concatenating them all ...
[02/13 17:04:06 d2.data.common]: Serialized dataset takes 0.66 MiB
[02/13 17:04:06 d2.data.build]: Making batched data loader with batch_size=2
WARNING [02/13 17:04:06 d2.solver.build]: SOLVER.STEPS contains values larger than SOLVER.MAX_ITER. These values will be ignored.
[02/13 17:04:06 d2.checkpoint.detection_checkpoint]: [DetectionCheckpointer] Loading from https://dl.fbaipublicfiles.com/detectron2/C
model_final_280758.pkl: 167MB [00:00, 252MB/s]
WARNING:fvcore.common.checkpoint:Skip loading parameter 'roi_heads.box_predictor.cls_score.weight' to the model due to incompatible s
WARNING:fvcore.common.checkpoint:Skip loading parameter 'roi_heads.box_predictor.cls_score.bias' to the model due to incompatible sha
WARNING:fvcore.common.checkpoint:Skip loading parameter 'roi_heads.box_predictor.bbox_pred.weight' to the model due to incompatible s
WARNING:fvcore.common.checkpoint:Skip loading parameter 'roi_heads.box_predictor.bbox_pred.bias' to the model due to incompatible sha
WARNING:fvcore.common.checkpoint:Some model parameters or buffers are not found in the checkpoint:
roi_heads.box_predictor.bbox_pred.{bias, weight}
roi_heads.box_predictor.cls_score.{bias, weight}
```

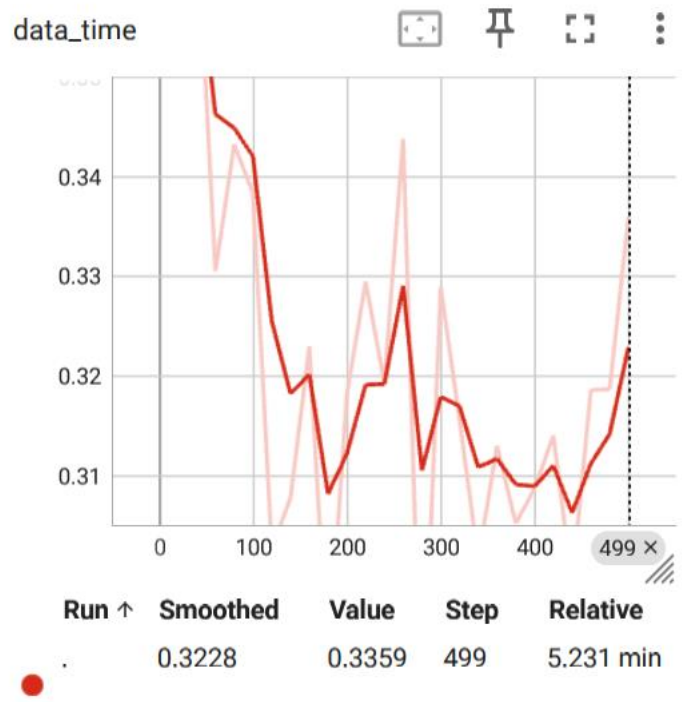
```
[02/13 17:04:07 d2.engine.train_loop]: Starting training from iteration 0
[02/13 17:04:14 d2.utils.events]:  eta: 0:02:28  iter: 19  total_loss: 1.776  loss_cls: 1.565  loss_box_reg: 0.1553  loss_rpn_cls: 0.001921  loss_rpn_loc: 0.01995    ti
[02/13 17:04:20 d2.utils.events]:  eta: 0:02:21  iter: 39  total_loss: 0.5521  loss_cls: 0.2463  loss_box_reg: 0.2702  loss_rpn_cls: 0.001117  loss_rpn_loc: 0.01308
[02/13 17:04:26 d2.utils.events]:  eta: 0:02:14  iter: 59  total_loss: 0.3677  loss_cls: 0.1588  loss_box_reg: 0.2055  loss_rpn_cls: 0.0007667  loss_rpn_loc: 0.0137
[02/13 17:04:32 d2.utils.events]:  eta: 0:02:08  iter: 79  total_loss: 0.4143  loss_cls: 0.1456  loss_box_reg: 0.2568  loss_rpn_cls: 0.0006243  loss_rpn_loc: 0.009493
[02/13 17:04:39 d2.utils.events]:  eta: 0:02:03  iter: 99  total_loss: 0.3234  loss_cls: 0.1122  loss_box_reg: 0.2003  loss_rpn_cls: 0.0005413  loss_rpn_loc: 0.007937
[02/13 17:04:45 d2.utils.events]:  eta: 0:01:56  iter: 119  total_loss: 0.3813  loss_cls: 0.1242  loss_box_reg: 0.2414  loss_rpn_cls: 0.0002867  loss_rpn_loc: 0.007567
[02/13 17:04:51 d2.utils.events]:  eta: 0:01:50  iter: 139  total_loss: 0.311  loss_cls: 0.1154  loss_box_reg: 0.1887  loss_rpn_cls: 0.0005282  loss_rpn_loc: 0.01049
[02/13 17:04:58 d2.utils.events]:  eta: 0:01:44  iter: 159  total_loss: 0.2265  loss_cls: 0.08866  loss_box_reg: 0.1281  loss_rpn_cls: 0.0008027  loss_rpn_loc: 0.01081
[02/13 17:05:04 d2.utils.events]:  eta: 0:01:38  iter: 179  total_loss: 0.1859  loss_cls: 0.09291  loss_box_reg: 0.09341  loss_rpn_cls: 0.0004187  loss_rpn_loc: 0.00998
[02/13 17:05:10 d2.utils.events]:  eta: 0:01:32  iter: 199  total_loss: 0.1829  loss_cls: 0.1126  loss_box_reg: 0.04808  loss_rpn_cls: 0.0007826  loss_rpn_loc: 0.0122
[02/13 17:05:16 d2.utils.events]:  eta: 0:01:26  iter: 219  total_loss: 0.1842  loss_cls: 0.1167  loss_box_reg: 0.05468  loss_rpn_cls: 0.0006222  loss_rpn_loc: 0.01457
[02/13 17:05:22 d2.utils.events]:  eta: 0:01:19  iter: 239  total_loss: 0.1914  loss_cls: 0.1208  loss_box_reg: 0.05159  loss_rpn_cls: 0.0002704  loss_rpn_loc: 0.01006
[02/13 17:05:29 d2.utils.events]:  eta: 0:01:13  iter: 259  total_loss: 0.15  loss_cls: 0.09902  loss_box_reg: 0.03325  loss_rpn_cls: 0.000147  loss_rpn_loc: 0.009066
[02/13 17:05:35 d2.utils.events]:  eta: 0:01:07  iter: 279  total_loss: 0.1362  loss_cls: 0.09227  loss_box_reg: 0.03451  loss_rpn_cls: 0.0002166  loss_rpn_loc: 0.00829
[02/13 17:05:41 d2.utils.events]:  eta: 0:01:01  iter: 299  total_loss: 0.1318  loss_cls: 0.09129  loss_box_reg: 0.02814  loss_rpn_cls: 0.0001434  loss_rpn_loc: 0.00714
[02/13 17:05:47 d2.utils.events]:  eta: 0:00:55  iter: 319  total_loss: 0.151  loss_cls: 0.1047  loss_box_reg: 0.02981  loss_rpn_cls: 0.0002909  loss_rpn_loc: 0.007978
[02/13 17:05:53 d2.utils.events]:  eta: 0:00:48  iter: 339  total_loss: 0.1222  loss_cls: 0.09689  loss_box_reg: 0.02433  loss_rpn_cls: 0.0001567  loss_rpn_loc: 0.00749
[02/13 17:05:59 d2.utils.events]:  eta: 0:00:42  iter: 359  total_loss: 0.1163  loss_cls: 0.09097  loss_box_reg: 0.01958  loss_rpn_cls: 0.0001619  loss_rpn_loc: 0.00765
[02/13 17:06:05 d2.utils.events]:  eta: 0:00:36  iter: 379  total_loss: 0.1408  loss_cls: 0.1064  loss_box_reg: 0.02571  loss_rpn_cls: 0.0002915  loss_rpn_loc: 0.0083
[02/13 17:06:12 d2.utils.events]:  eta: 0:00:30  iter: 399  total_loss: 0.1369  loss_cls: 0.09776  loss_box_reg: 0.0306  loss_rpn_cls: 0.0002964  loss_rpn_loc: 0.007621
[02/13 17:06:18 d2.utils.events]:  eta: 0:00:24  iter: 419  total_loss: 0.1272  loss_cls: 0.08924  loss_box_reg: 0.02638  loss_rpn_cls: 0.0002199  loss_rpn_loc: 0.00943
[02/13 17:06:25 d2.utils.events]:  eta: 0:00:18  iter: 439  total_loss: 0.1275  loss_cls: 0.08854  loss_box_reg: 0.02945  loss_rpn_cls: 0.0002339  loss_rpn_loc: 0.00815
[02/13 17:06:31 d2.utils.events]:  eta: 0:00:12  iter: 459  total_loss: 0.1175  loss_cls: 0.07873  loss_box_reg: 0.02431  loss_rpn_cls: 0.000163  loss_rpn_loc: 0.008102
[02/13 17:06:37 d2.utils.events]:  eta: 0:00:06  iter: 479  total_loss: 0.1119  loss_cls: 0.07851  loss_box_reg: 0.02353  loss_rpn_cls: 0.0004063  loss_rpn_loc: 0.00831
[02/13 17:06:44 d2.utils.events]:  eta: 0:00:00  iter: 499  total_loss: 0.1176  loss_cls: 0.07018  loss_box_reg: 0.03333  loss_rpn_cls: 0.0003752  loss_rpn_loc: 0.00769
[02/13 17:06:45 d2.engine.hooks]: Overall training speed: 498 iterations in 0:02:34 (0.3111 s / it)
[02/13 17:06:45 d2.engine.hooks]: Total training time: 0:02:36 (0:00:01 on hooks)
```

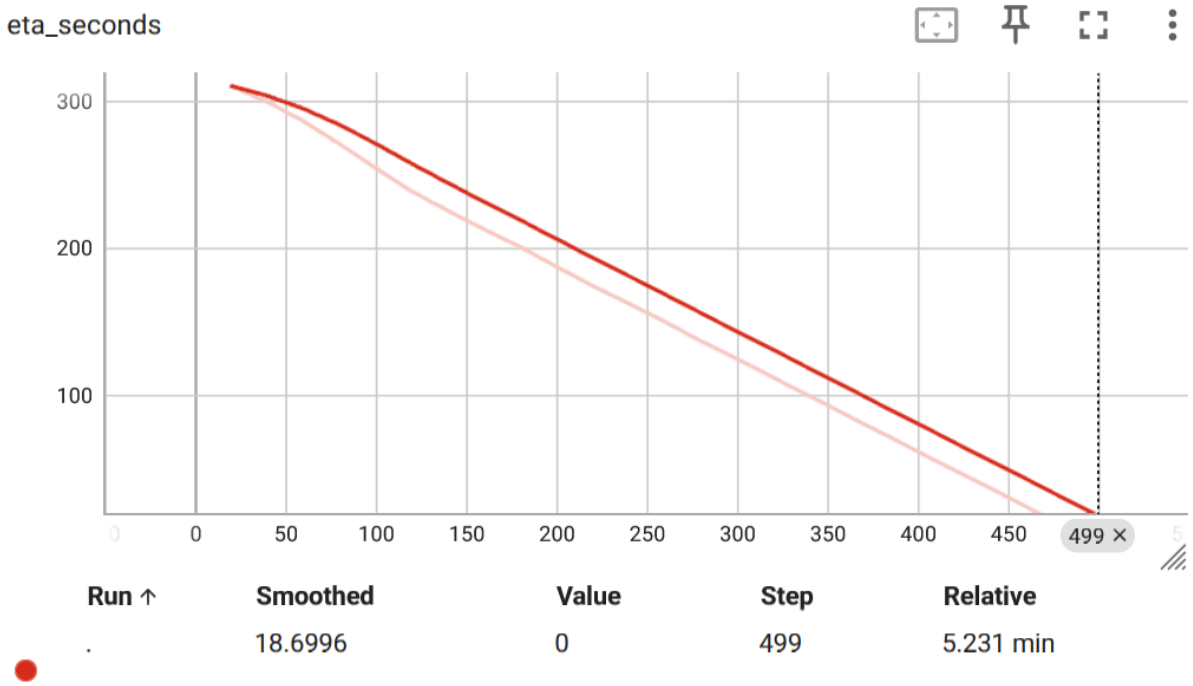
MASK RCNN:

```
[02/13 16:32:09 d2.data.datasets.coco]: Loaded 1727 images in COCO format from /content/drive/MyDri
[02/13 16:32:09 d2.data.build]: Removed 1 images with no usable annotations. 1726 images left.
[02/13 16:32:09 d2.data.build]: Distribution of instances among all 6 categories:
|  category   | #instances   |  category   | #instances   |  category   | #instances   |
|:-----------:|:-------------|:-----------:|:-------------|:-----------:|:-------------|
|  objects    | 0            | 0hrMutton   | 346          | 12hrMutton  | 351          |
| 24hrMutton  | 363          | 36hrMutton  | 322          | 48hrMutton  | 344          |
|             |              |             |              |             |              |
|   total     | 1726         |             |              |             |              |
```

```
oop]: Starting training from iteration 0
ckages/torch/functional.py:534: UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Triggered internally at ../aten/src/ATen/native/Te
kwargs)  # type: ignore[attr-defined]
  eta: 0:05:10  iter: 19  total_loss: 2.425  loss_cls: 1.522  loss_box_reg: 0.2017  loss_mask: 0.6816  loss_rpn_cls: 0.002351  loss_rpn_loc: 0.01739    time: 0.6718  last_time: 0.6222  da
  eta: 0:05:00  iter: 39  total_loss: 0.9206  loss_cls: 0.2469  loss_box_reg: 0.2603  loss_mask: 0.4479  loss_rpn_cls: 0.001458  loss_rpn_loc: 0.01627    time: 0.6555  last_time: 0.7196
  eta: 0:04:46  iter: 59  total_loss: 0.5201  loss_cls: 0.1667  loss_box_reg: 0.2143  loss_mask: 0.1074  loss_rpn_cls: 0.001207  loss_rpn_loc: 0.01219    time: 0.6552  last_time: 0.8587
  eta: 0:04:31  iter: 79  total_loss: 0.4164  loss_cls: 0.1379  loss_box_reg: 0.2413  loss_mask: 0.01079  loss_rpn_cls: 0.00102  loss_rpn_loc: 0.01017    time: 0.6487  last_time: 0.5669
  eta: 0:04:15  iter: 99  total_loss: 0.3199  loss_cls: 0.1113  loss_box_reg: 0.1879  loss_mask: 0.002382  loss_rpn_cls: 0.0007902  loss_rpn_loc: 0.009166    time: 0.6471  last_time: 0.63
  eta: 0:03:59  iter: 119  total_loss: 0.2996  loss_cls: 0.1118  loss_box_reg: 0.18  loss_mask: 0.001618  loss_rpn_cls: 0.0011  loss_rpn_loc: 0.009203    time: 0.6473  last_time: 0.5997
  eta: 0:03:46  iter: 139  total_loss: 0.2359  loss_cls: 0.09911  loss_box_reg: 0.1285  loss_mask: 0.001308  loss_rpn_cls: 0.0006212  loss_rpn_loc: 0.01076    time: 0.6445  last_time: 0.5
  eta: 0:03:33  iter: 159  total_loss: 0.1788  loss_cls: 0.09583  loss_box_reg: 0.06358  loss_mask: 0.0008123  loss_rpn_cls: 0.0008911  loss_rpn_loc: 0.01022    time: 0.6423  last_time: 0
  eta: 0:03:21  iter: 179  total_loss: 0.1275  loss_cls: 0.08379  loss_box_reg: 0.0277  loss_mask: 0.0006224  loss_rpn_cls: 0.0003985  loss_rpn_loc: 0.01329    time: 0.6444  last_time: 0.
  eta: 0:03:08  iter: 199  total_loss: 0.1485  loss_cls: 0.1003  loss_box_reg: 0.02913  loss_mask: 0.001762  loss_rpn_cls: 0.0003022  loss_rpn_loc: 0.008506    time: 0.6411  last_time: 0.
  eta: 0:02:55  iter: 219  total_loss: 0.1671  loss_cls: 0.09986  loss_box_reg: 0.04457  loss_mask: 0.00307  loss_rpn_cls: 0.0003451  loss_rpn_loc: 0.01273    time: 0.6458  last_time: 0.5
  eta: 0:02:42  iter: 239  total_loss: 0.1502  loss_cls: 0.09191  loss_box_reg: 0.04289  loss_mask: 0.002542  loss_rpn_cls: 0.0004754  loss_rpn_loc: 0.01128    time: 0.6499  last_time: 0.
  eta: 0:02:30  iter: 259  total_loss: 0.1435  loss_cls: 0.09681  loss_box_reg: 0.03779  loss_mask: 0.0023  loss_rpn_cls: 0.0001784  loss_rpn_loc: 0.009733    time: 0.6520  last_time: 0.5
  eta: 0:02:17  iter: 279  total_loss: 0.1572  loss_cls: 0.08994  loss_box_reg: 0.02744  loss_mask: 0.003972  loss_rpn_cls: 0.0002197  loss_rpn_loc: 0.008887    time: 0.6504  last_time: 0
  eta: 0:02:05  iter: 299  total_loss: 0.1317  loss_cls: 0.08593  loss_box_reg: 0.02553  loss_mask: 0.003115  loss_rpn_cls: 0.0003891  loss_rpn_loc: 0.00829    time: 0.6527  last_time: 0.
  eta: 0:01:52  iter: 319  total_loss: 0.1421  loss_cls: 0.09322  loss_box_reg: 0.03031  loss_mask: 0.002378  loss_rpn_cls: 0.0001317  loss_rpn_loc: 0.008616    time: 0.6516  last_time: 0
  eta: 0:01:40  iter: 339  total_loss: 0.1221  loss_cls: 0.08543  loss_box_reg: 0.0254  loss_mask: 0.001866  loss_rpn_cls: 0.0004828  loss_rpn_loc: 0.01183    time: 0.6491  last_time: 0.5
  eta: 0:01:27  iter: 359  total_loss: 0.123  loss_cls: 0.08399  loss_box_reg: 0.02325  loss_mask: 0.001008  loss_rpn_cls: 0.0002752  loss_rpn_loc: 0.0108    time: 0.6510  last_time: 0.51
  eta: 0:01:14  iter: 379  total_loss: 0.141  loss_cls: 0.09137  loss_box_reg: 0.02478  loss_mask: 0.001621  loss_rpn_cls: 0.0004382  loss_rpn_loc: 0.007484    time: 0.6494  last_time: 0.
  eta: 0:01:02  iter: 399  total_loss: 0.1476  loss_cls: 0.1026  loss_box_reg: 0.03733  loss_mask: 0.002992  loss_rpn_cls: 0.0003133  loss_rpn_loc: 0.007473    time: 0.6502  last_time: 0.
  eta: 0:00:49  iter: 419  total_loss: 0.1344  loss_cls: 0.08802  loss_box_reg: 0.03069  loss_mask: 0.001709  loss_rpn_cls: 0.0002886  loss_rpn_loc: 0.009848    time: 0.6499  last_time: 0
  eta: 0:00:37  iter: 439  total_loss: 0.132  loss_cls: 0.08595  loss_box_reg: 0.03097  loss_mask: 0.001141  loss_rpn_cls: 0.0004163  loss_rpn_loc: 0.00765    time: 0.6489  last_time: 0.5
  eta: 0:00:24  iter: 459  total_loss: 0.1096  loss_cls: 0.07108  loss_box_reg: 0.02455  loss_mask: 0.001486  loss_rpn_cls: 0.0001455  loss_rpn_loc: 0.008161    time: 0.6496  last_time: 0
  eta: 0:00:12  iter: 479  total_loss: 0.1067  loss_cls: 0.06646  loss_box_reg: 0.02146  loss_mask: 0.0009339  loss_rpn_cls: 0.0005197  loss_rpn_loc: 0.007793    time: 0.6503  last_time:
  eta: 0:00:00  iter: 499  total_loss: 0.08971  loss_cls: 0.05991  loss_box_reg: 0.02367  loss_mask: 0.0007648  loss_rpn_cls: 0.0004396  loss_rpn_loc: 0.006739    time: 0.6505  last_time:
Overall training speed: 498 iterations in 0:05:23 (0.6505 s / it)
Total training time: 0:05:30 (0:00:06 on hooks)
```

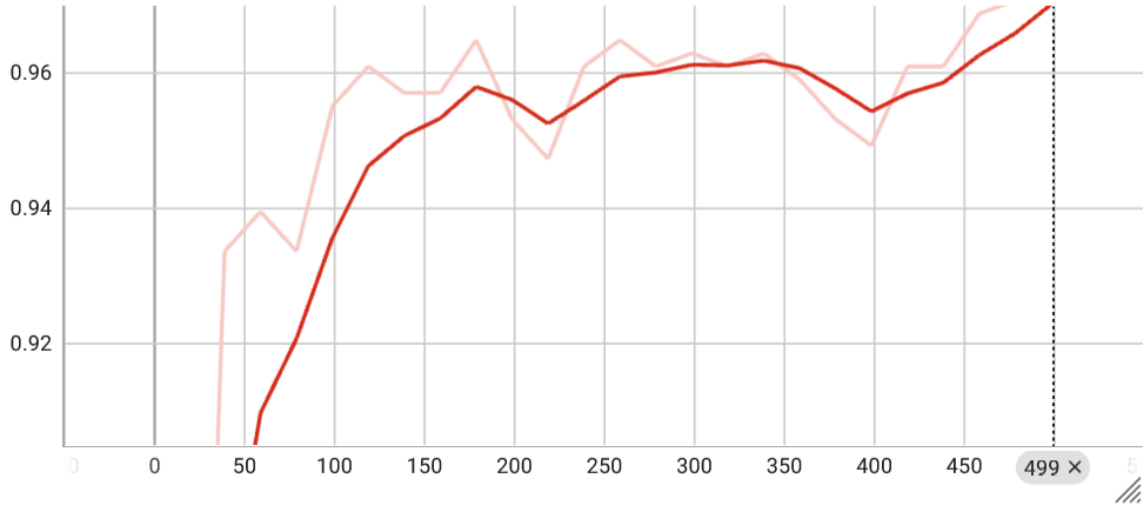
Name: PRANAY GORANTLA
REG.NO: 21MIS0123





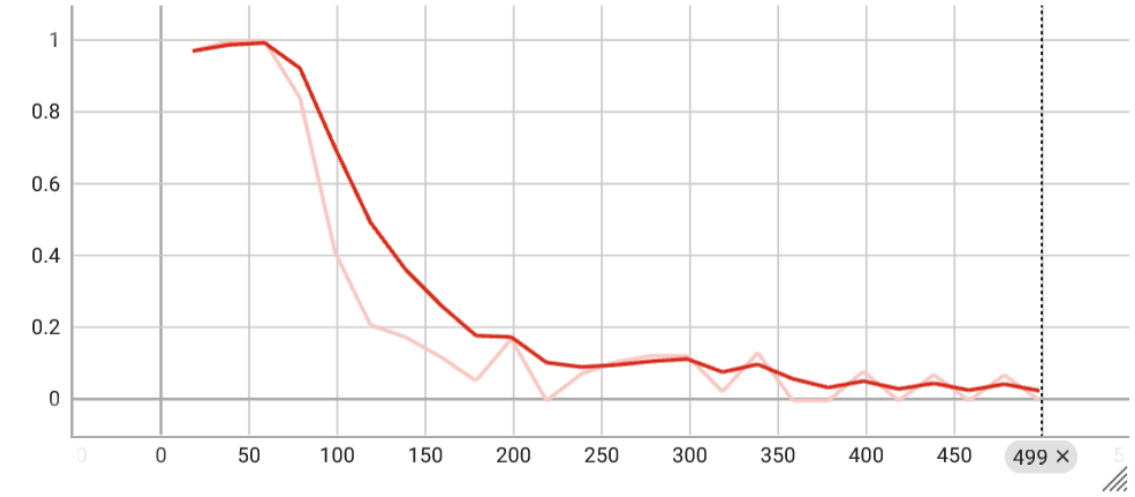
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

fast_rcnn/cls_accuracy



Run	Smoothed	Value	Step	Relative
.	0.9701	0.9766	499	5.231 min

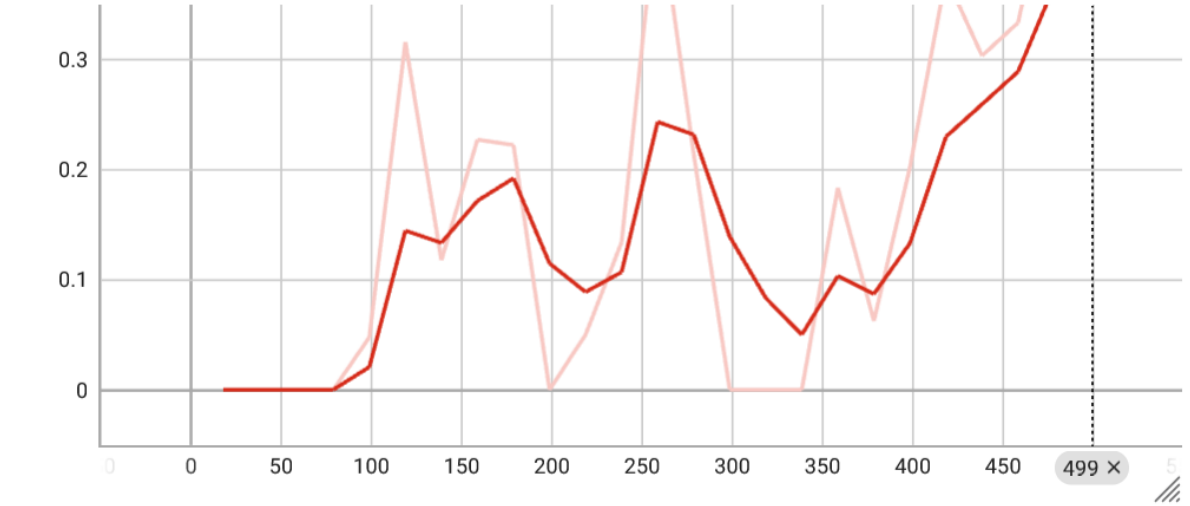
fast_rcnn/false_negative



Run	Smoothed	Value	Step	Relative
.	0.0276	0	499	5.231 min

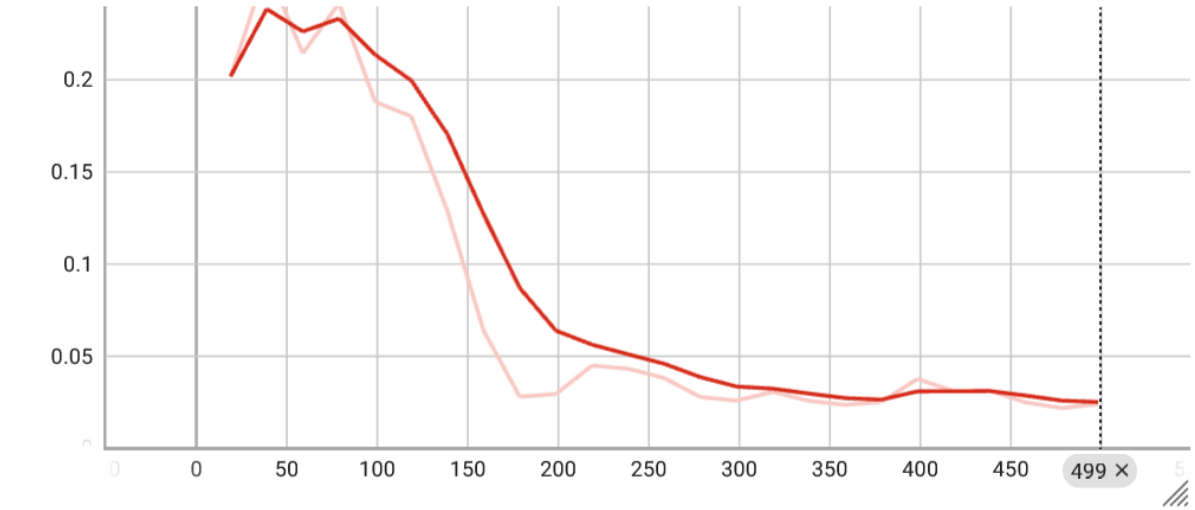
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

fast_rcnn/fg_cls_accuracy



Run ↑	Smoothed	Value	Step	Relative
.	0.4354	0.5385	499	5.231 min

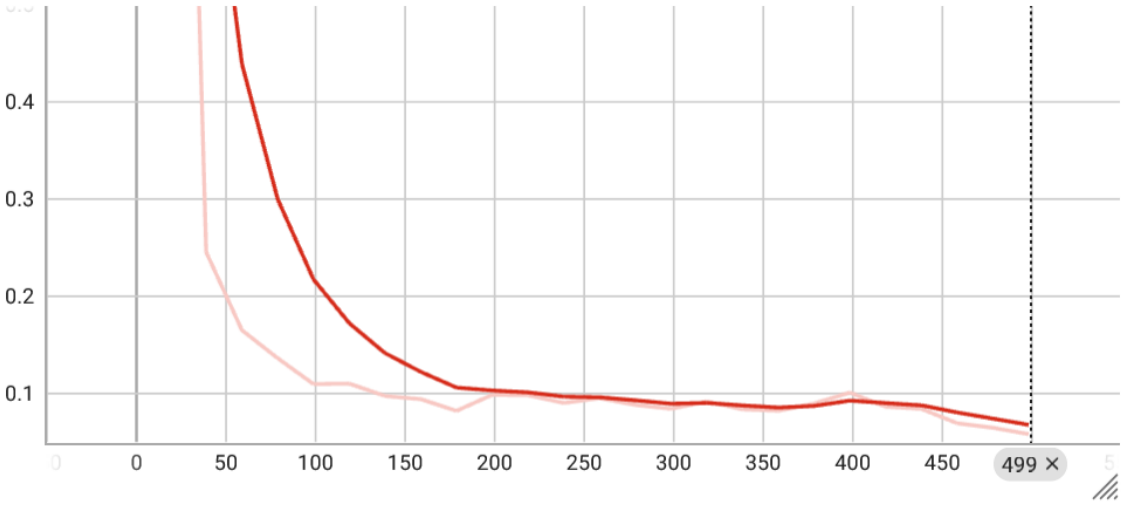
loss_box_reg



Run ↑	Smoothed	Value	Step	Relative
.	0.0248	0.0237	499	5.231 min

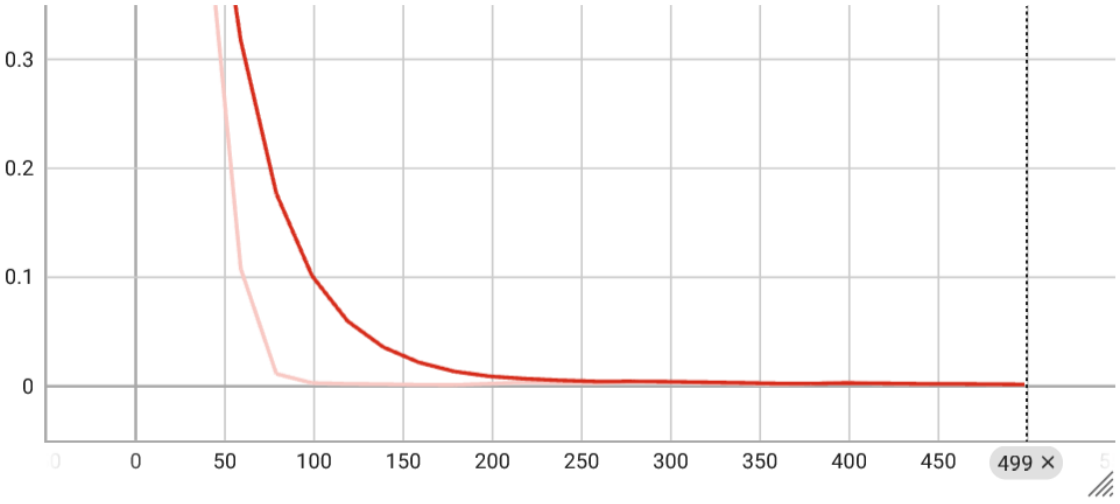
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

loss_cls



Run ↑	Smoothed	Value	Step	Relative
.	0.0695	0.0599	499	5.231 min

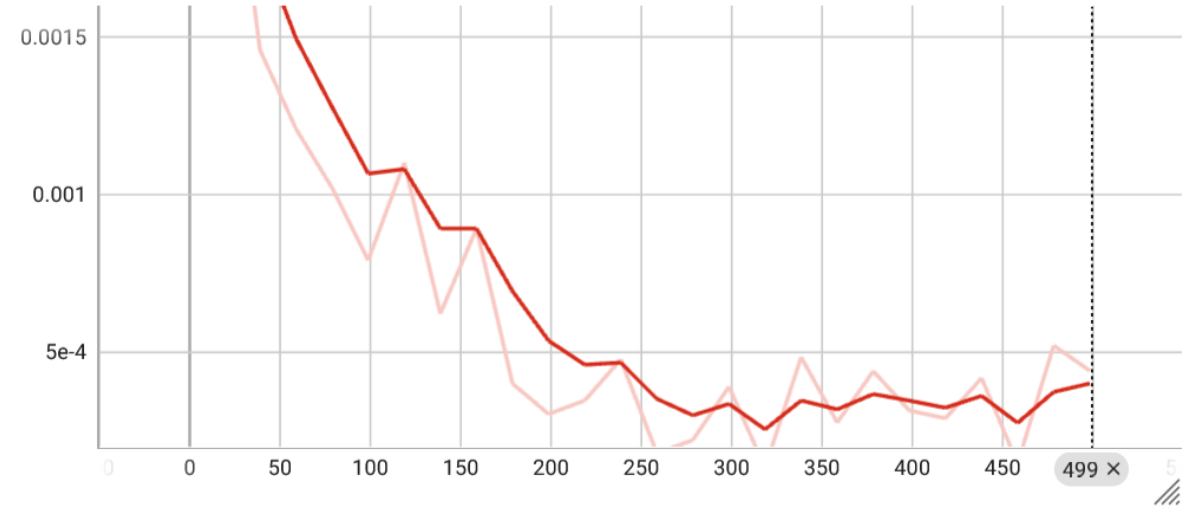
loss_mask



Run ↑	Smoothed	Value	Step	Relative
.	0.0011	0.0008	499	5.231 min

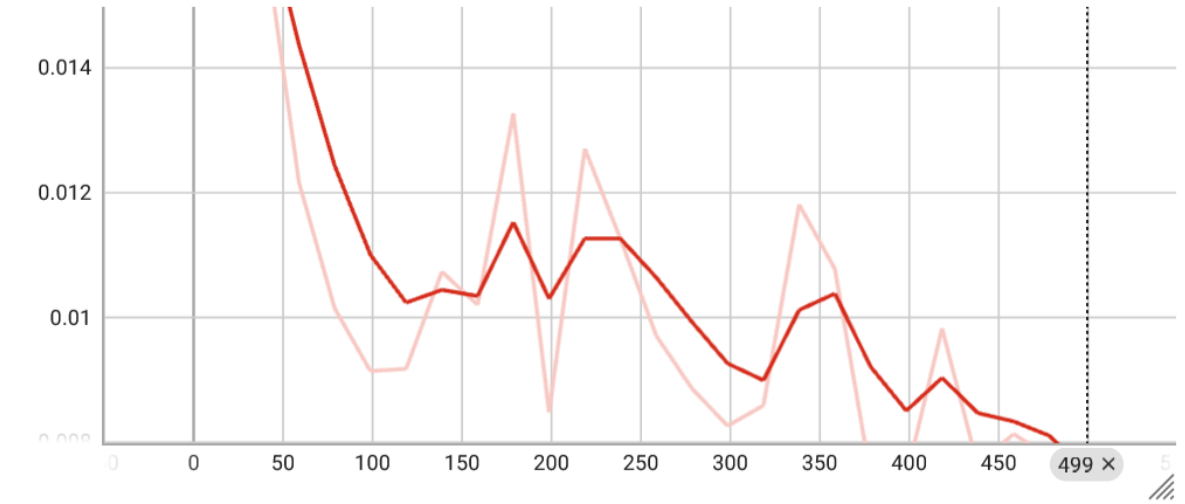
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

loss_rpn_cls



Run	Smoothed	Value	Step	Relative
.	0.0004	0.0004	499	5.231 min

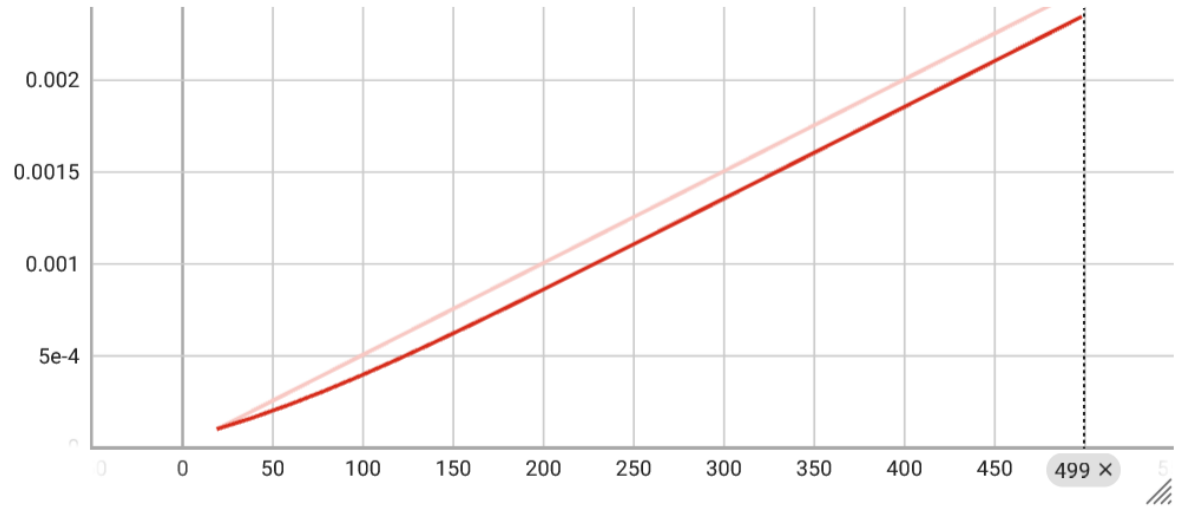
loss_rpn_loc



Run	Smoothed	Value	Step	Relative
.	0.0076	0.0067	499	5.231 min

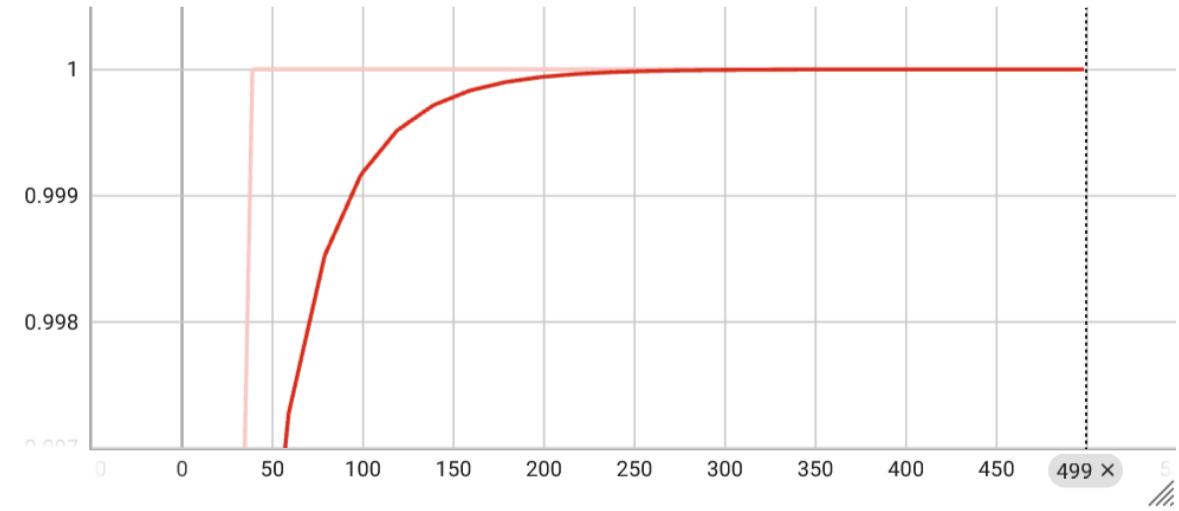
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

lr



Run	Smoothed	Value	Step	Relative
.	0.0023	0.0025	499	5.231 min

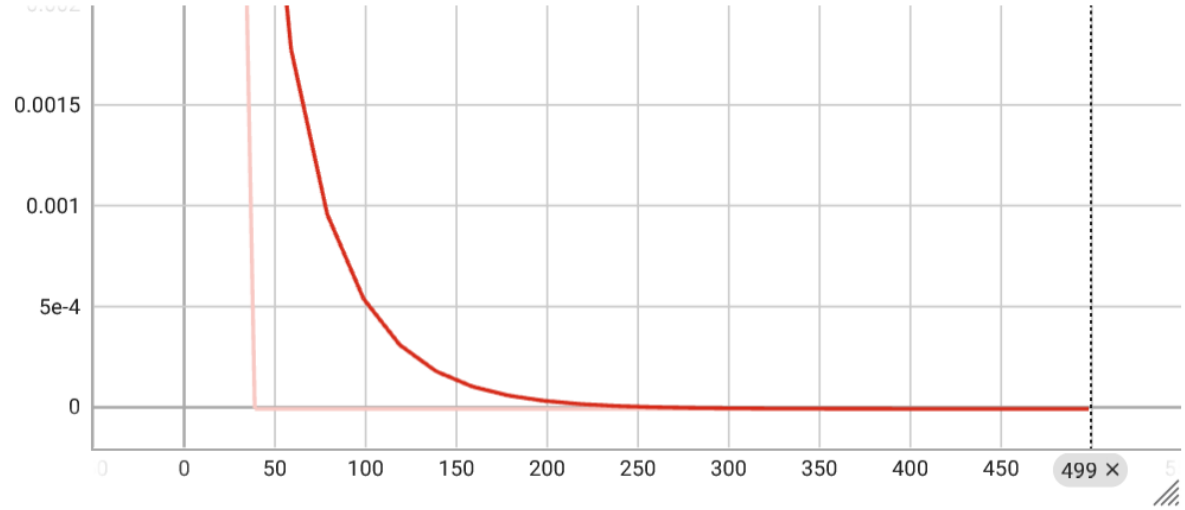
mask_rcnn/accuracy



Run	Smoothed	Value	Step	Relative
.	1	1	499	5.231 min

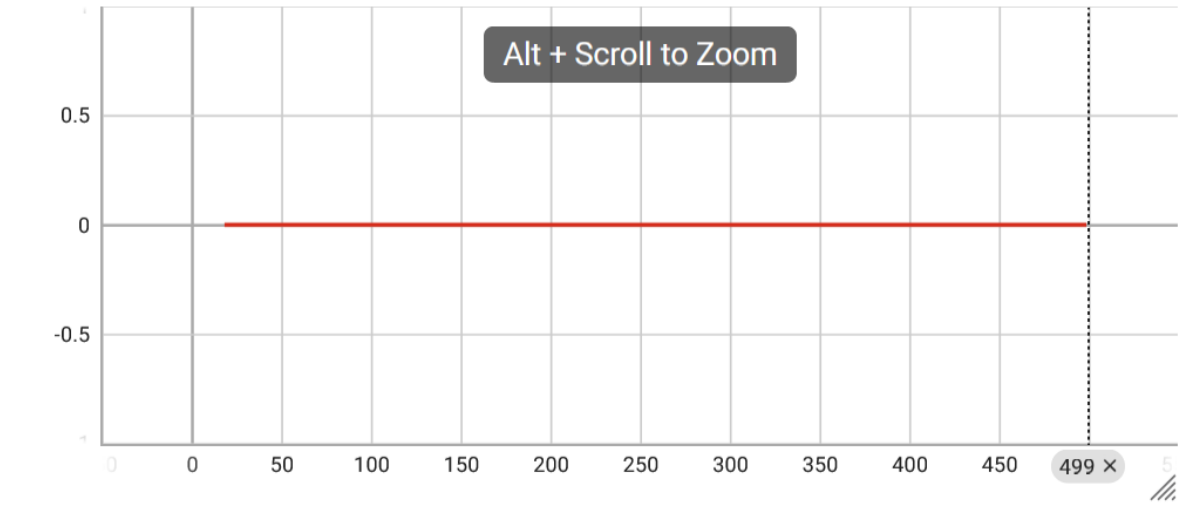
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

mask_rcnn/false_negative



Run ↑	Smoothed	Value	Step	Relative
.	0	0	499	5.231 min

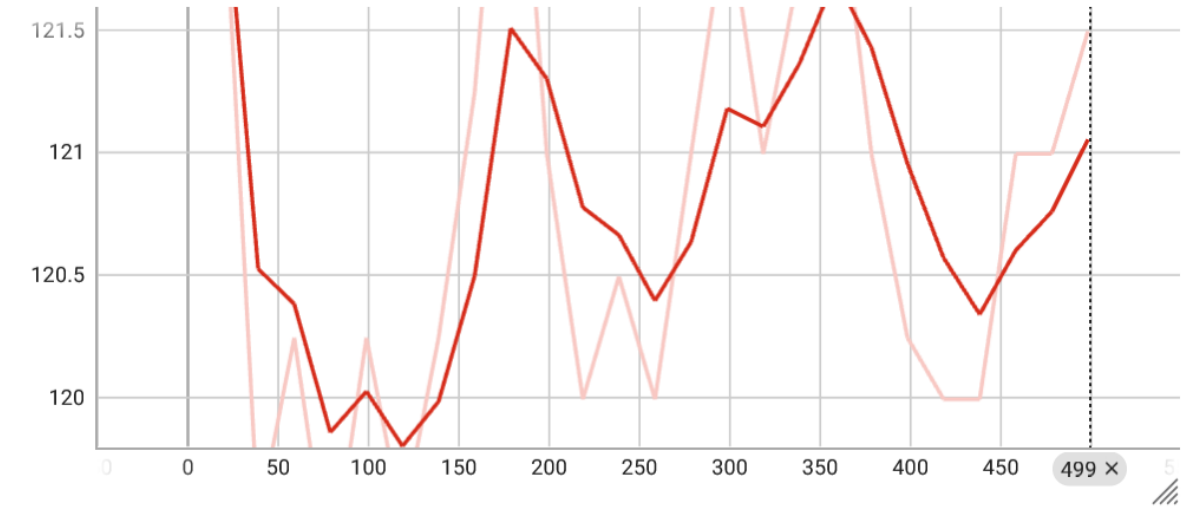
mask_rcnn/false_positive



Run ↑	Smoothed	Value	Step	Relative
.	0	0	499	5.231 min

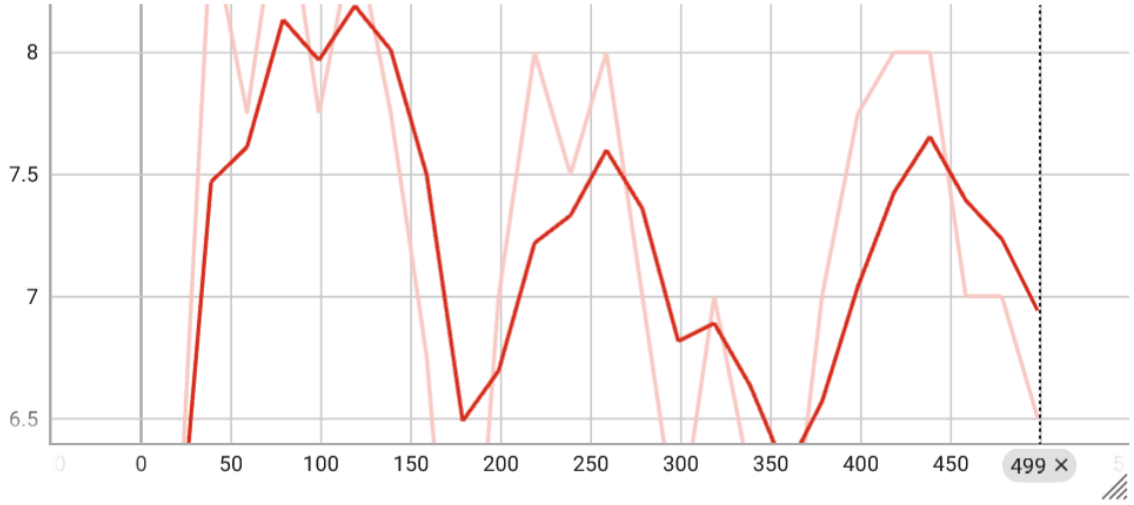
Name: PRANAY GORANTLA
REG.NO: 21MIS0123

roi_head/num_bg_samples



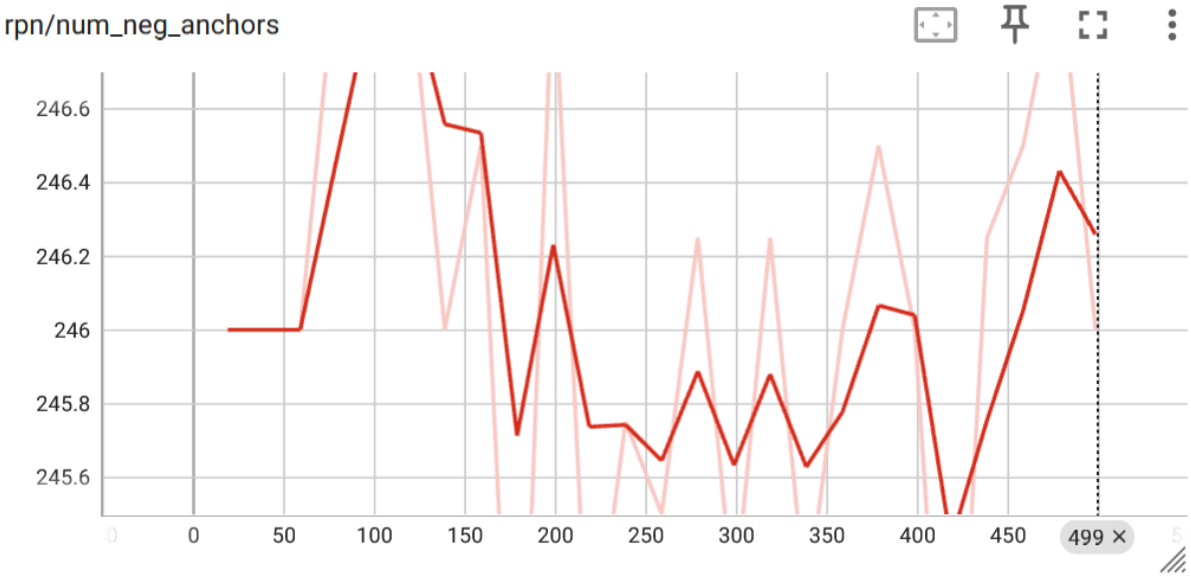
Run ↑	Smoothed	Value	Step	Relative
.	121.0586	121.5	499	5.231 min

roi_head/num_fg_samples

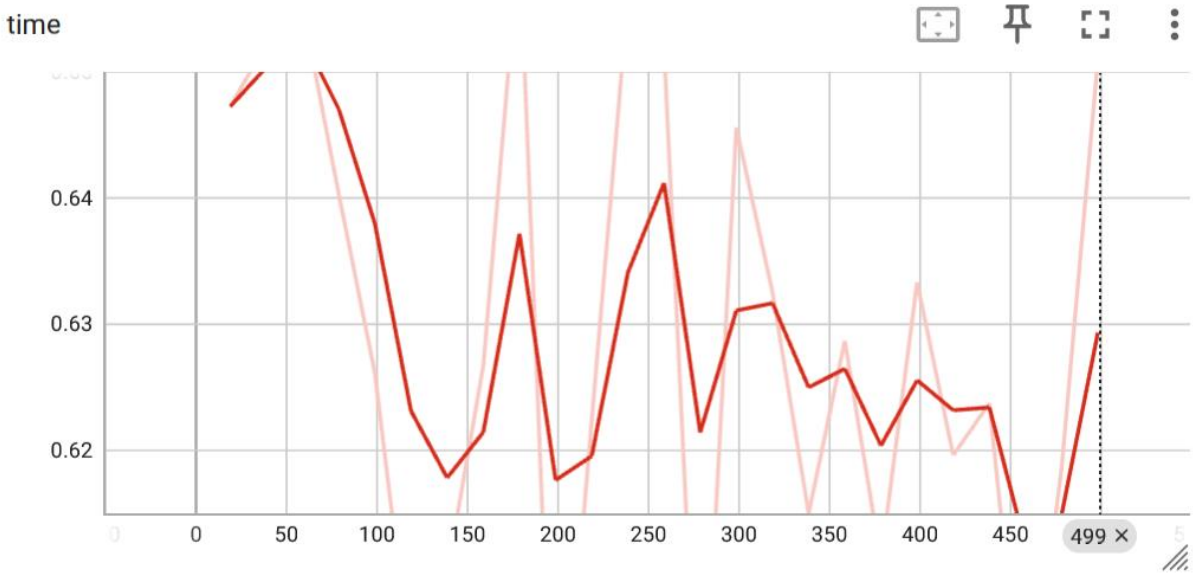


Run ↑	Smoothed	Value	Step	Relative
.	6.9414	6.5	499	5.231 min

Name: PRANAY GORANTLA
REG.NO: 21MIS0123



Run ↑	Smoothed	Value	Step	Relative
.	246.2589	246	499	5.231 min



Run ↑	Smoothed	Value	Step	Relative
.	0.6293	0.6509	499	5.231 min

```
Category ids in annotations are not in [1, #categories]! We'll apply a mapping for you.

[02/13 16:38:12 d2.data.datasets.coco]: Loaded 493 images in COCO format from /content/drive/MyDrive/newAugmentedMut
[02/13 16:38:12 d2.data.build]: Distribution of instances among all 6 categories:
|  category  | #instances   |  category  | #instances   |  category  | #instances   |
|:----------:|:-------------|:----------:|:-------------|:----------:|:-------------|
|  objects   | 0            | 0hrMutton  | 99           | 12hrMutton | 100          |
| 24hrMutton | 103          | 36hrMutton | 92           | 48hrMutton | 99           |
|            |              |            |              |            |              |
|   total    | 493          |            |              |            |              |
[02/13 16:38:12 d2.data.dataset_mapper]: [DatasetMapper] Augmentations used in inference: [ResizeShortestEdge(short_
[02/13 16:38:12 d2.data.common]: Serializing the dataset using: <class 'detectron2.data.common._TorchSerializedList'
[02/13 16:38:12 d2.data.common]: Serializing 493 elements to byte tensors and concatenating them all ...
[02/13 16:38:12 d2.data.common]: Serialized dataset takes 0.19 MiB
```

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

```
[02/13 16:38:12 d2.evaluation.evaluator]: Start inference on 493 batches
[02/13 16:38:17 d2.evaluation.evaluator]: Inference done 11/493. Dataloading: 0.3286 s/iter. Inference: 0.0817 s/iter. Eval: 0.0006 s/iter. Total: 0.4109 s/iter. ETA=0:03:18
[02/13 16:38:22 d2.evaluation.evaluator]: Inference done 27/493. Dataloading: 0.2666 s/iter. Inference: 0.0807 s/iter. Eval: 0.0006 s/iter. Total: 0.3481 s/iter. ETA=0:02:42
[02/13 16:38:28 d2.evaluation.evaluator]: Inference done 44/493. Dataloading: 0.2473 s/iter. Inference: 0.0833 s/iter. Eval: 0.0007 s/iter. Total: 0.3315 s/iter. ETA=0:02:28
[02/13 16:38:33 d2.evaluation.evaluator]: Inference done 60/493. Dataloading: 0.2424 s/iter. Inference: 0.0845 s/iter. Eval: 0.0007 s/iter. Total: 0.3278 s/iter. ETA=0:02:21
[02/13 16:38:38 d2.evaluation.evaluator]: Inference done 71/493. Dataloading: 0.2699 s/iter. Inference: 0.0844 s/iter. Eval: 0.0007 s/iter. Total: 0.3552 s/iter. ETA=0:02:29
[02/13 16:38:43 d2.evaluation.evaluator]: Inference done 88/493. Dataloading: 0.2583 s/iter. Inference: 0.0845 s/iter. Eval: 0.0006 s/iter. Total: 0.3436 s/iter. ETA=0:02:19
[02/13 16:38:48 d2.evaluation.evaluator]: Inference done 103/493. Dataloading: 0.2579 s/iter. Inference: 0.0839 s/iter. Eval: 0.0006 s/iter. Total: 0.3426 s/iter. ETA=0:02:13
[02/13 16:38:53 d2.evaluation.evaluator]: Inference done 118/493. Dataloading: 0.2592 s/iter. Inference: 0.0837 s/iter. Eval: 0.0006 s/iter. Total: 0.3436 s/iter. ETA=0:02:08
[02/13 16:38:59 d2.evaluation.evaluator]: Inference done 136/493. Dataloading: 0.2535 s/iter. Inference: 0.0838 s/iter. Eval: 0.0006 s/iter. Total: 0.3380 s/iter. ETA=0:02:00
[02/13 16:39:04 d2.evaluation.evaluator]: Inference done 154/493. Dataloading: 0.2494 s/iter. Inference: 0.0837 s/iter. Eval: 0.0006 s/iter. Total: 0.3339 s/iter. ETA=0:01:53
[02/13 16:39:10 d2.evaluation.evaluator]: Inference done 168/493. Dataloading: 0.2523 s/iter. Inference: 0.0835 s/iter. Eval: 0.0006 s/iter. Total: 0.3366 s/iter. ETA=0:01:49
[02/13 16:39:15 d2.evaluation.evaluator]: Inference done 186/493. Dataloading: 0.2491 s/iter. Inference: 0.0834 s/iter. Eval: 0.0006 s/iter. Total: 0.3333 s/iter. ETA=0:01:42
[02/13 16:39:20 d2.evaluation.evaluator]: Inference done 200/493. Dataloading: 0.2498 s/iter. Inference: 0.0846 s/iter. Eval: 0.0007 s/iter. Total: 0.3352 s/iter. ETA=0:01:38
[02/13 16:39:25 d2.evaluation.evaluator]: Inference done 216/493. Dataloading: 0.2499 s/iter. Inference: 0.0843 s/iter. Eval: 0.0007 s/iter. Total: 0.3350 s/iter. ETA=0:01:32
[02/13 16:39:31 d2.evaluation.evaluator]: Inference done 232/493. Dataloading: 0.2497 s/iter. Inference: 0.0844 s/iter. Eval: 0.0007 s/iter. Total: 0.3349 s/iter. ETA=0:01:27
[02/13 16:39:36 d2.evaluation.evaluator]: Inference done 248/493. Dataloading: 0.2489 s/iter. Inference: 0.0842 s/iter. Eval: 0.0007 s/iter. Total: 0.3339 s/iter. ETA=0:01:21
[02/13 16:39:41 d2.evaluation.evaluator]: Inference done 266/493. Dataloading: 0.2473 s/iter. Inference: 0.0840 s/iter. Eval: 0.0007 s/iter. Total: 0.3322 s/iter. ETA=0:01:15
[02/13 16:39:46 d2.evaluation.evaluator]: Inference done 282/493. Dataloading: 0.2468 s/iter. Inference: 0.0839 s/iter. Eval: 0.0006 s/iter. Total: 0.3315 s/iter. ETA=0:01:09
[02/13 16:39:52 d2.evaluation.evaluator]: Inference done 300/493. Dataloading: 0.2447 s/iter. Inference: 0.0838 s/iter. Eval: 0.0006 s/iter. Total: 0.3293 s/iter. ETA=0:01:03
[02/13 16:39:57 d2.evaluation.evaluator]: Inference done 316/493. Dataloading: 0.2445 s/iter. Inference: 0.0837 s/iter. Eval: 0.0006 s/iter. Total: 0.3290 s/iter. ETA=0:00:58
[02/13 16:40:02 d2.evaluation.evaluator]: Inference done 332/493. Dataloading: 0.2440 s/iter. Inference: 0.0835 s/iter. Eval: 0.0006 s/iter. Total: 0.3283 s/iter. ETA=0:00:52
[02/13 16:40:08 d2.evaluation.evaluator]: Inference done 348/493. Dataloading: 0.2457 s/iter. Inference: 0.0836 s/iter. Eval: 0.0007 s/iter. Total: 0.3300 s/iter. ETA=0:00:47
[02/13 16:40:13 d2.evaluation.evaluator]: Inference done 364/493. Dataloading: 0.2451 s/iter. Inference: 0.0837 s/iter. Eval: 0.0007 s/iter. Total: 0.3296 s/iter. ETA=0:00:42
[02/13 16:40:18 d2.evaluation.evaluator]: Inference done 380/493. Dataloading: 0.2457 s/iter. Inference: 0.0836 s/iter. Eval: 0.0007 s/iter. Total: 0.3301 s/iter. ETA=0:00:37
[02/13 16:40:24 d2.evaluation.evaluator]: Inference done 396/493. Dataloading: 0.2453 s/iter. Inference: 0.0836 s/iter. Eval: 0.0007 s/iter. Total: 0.3297 s/iter. ETA=0:00:31
[02/13 16:40:29 d2.evaluation.evaluator]: Inference done 413/493. Dataloading: 0.2446 s/iter. Inference: 0.0836 s/iter. Eval: 0.0007 s/iter. Total: 0.3291 s/iter. ETA=0:00:26
[02/13 16:40:34 d2.evaluation.evaluator]: Inference done 429/493. Dataloading: 0.2450 s/iter. Inference: 0.0837 s/iter. Eval: 0.0007 s/iter. Total: 0.3296 s/iter. ETA=0:00:21
[02/13 16:40:42 d2.evaluation.evaluator]: Inference done 444/493. Dataloading: 0.2510 s/iter. Inference: 0.0838 s/iter. Eval: 0.0007 s/iter. Total: 0.3356 s/iter. ETA=0:00:16
[02/13 16:40:47 d2.evaluation.evaluator]: Inference done 461/493. Dataloading: 0.2496 s/iter. Inference: 0.0838 s/iter. Eval: 0.0007 s/iter. Total: 0.3342 s/iter. ETA=0:00:10
[02/13 16:40:52 d2.evaluation.evaluator]: Inference done 476/493. Dataloading: 0.2503 s/iter. Inference: 0.0838 s/iter. Eval: 0.0007 s/iter. Total: 0.3349 s/iter. ETA=0:00:05
[02/13 16:40:57 d2.evaluation.evaluator]: Total inference time: 0:02:42.633254 (0.333265 s / iter per device, on 1 devices)
[02/13 16:40:57 d2.evaluation.evaluator]: Total inference pure compute time: 0:00:40 (0.083722 s / iter per device, on 1 devices)
[02/13 16:40:57 d2.evaluation.coco_evaluation]: Preparing results for COCO format ...
[02/13 16:40:57 d2.evaluation.coco_evaluation]: Saving results to ./output/coco_instances_results.json
[02/13 16:40:57 d2.evaluation.coco_evaluation]: Evaluating predictions with unofficial COCO API...
Loading and preparing results...
DONE (t=0.00s)
creating index...
index created!
```

```
[02/13 16:40:57 d2.evaluation.fast_eval_api]: Evaluate annotation type *bbox*
[02/13 16:40:58 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.41 seconds.
[02/13 16:40:58 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/13 16:40:58 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.02 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.205
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.207
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.207
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.205
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.242
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Evaluation results for bbox:
|   AP   |  AP50  |  AP75  |  APs  |  APm  |  APl   |
|:------:|:------:|:------:|:-----:|:-----:|:------:|
| 20.458 | 20.699 | 20.699 |  nan  |  nan  | 20.458 |
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category   | AP    | category   | AP     | category   | AP    |
|:-----------|:------|:-----------|:-------|:-----------|:------|
| objects    | nan   | 0hrMutton  | 93.971 | 12hrMutton | 0.000 |
| 24hrMutton | 0.000 | 36hrMutton | 8.321  | 48hrMutton | 0.000 |
Loading and preparing results...
DONE (t=0.01s)
creating index...
index created!
```

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

```
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Per-category bbox AP:
| category   | AP    | category   | AP     | category   | AP    |
|:-----------|:------|:-----------|:-------|:-----------|:------|
| objects    | nan   | 0hrMutton  | 93.971 | 12hrMutton | 0.000 |
| 24hrMutton | 0.000 | 36hrMutton | 8.321  | 48hrMutton | 0.000 |
Loading and preparing results...
DONE (t=0.01s)
creating index...
index created!
[02/13 16:40:58 d2.evaluation.fast_eval_api]: Evaluate annotation type *segm*
[02/13 16:40:58 d2.evaluation.fast_eval_api]: COCOeval_opt.evaluate() finished in 0.15 seconds.
[02/13 16:40:58 d2.evaluation.fast_eval_api]: Accumulating evaluation results...
[02/13 16:40:58 d2.evaluation.fast_eval_api]: COCOeval_opt.accumulate() finished in 0.03 seconds.
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.205
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.207
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.207
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.205
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.242
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.242
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Evaluation results for segm:
|   AP   |  AP50  |  AP75  |  APs  |  APm  |  APl   |
|:-------|:-------|:-------|:------|:------|:-------|
| 20.458 | 20.699 | 20.699 |  nan  |  nan  | 20.458 |
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Some metrics cannot be computed and is shown as NaN.
[02/13 16:40:58 d2.evaluation.coco_evaluation]: Per-category segm AP:
| category   | AP    | category   | AP     | category   | AP    |
|:-----------|:------|:-----------|:-------|:-----------|:------|
| objects    | nan   | 0hrMutton  | 93.971 | 12hrMutton | 0.000 |
| 24hrMutton | 0.000 | 36hrMutton | 8.321  | 48hrMutton | 0.000 |
OrderedDict([('bbox', {'AP': 20.458357356134574, 'AP50': 20.699429966946912, 'AP75': 20.699429966946911
```

REFERENCES:

[1] Soo-Kyoung Lee, Jung-Whan Chon, Young-Kwon Yun, Jae-Chung Lee, Cheorun Jo, Kwang-Young Song, Dong-Hyeon Kim, Dongryeoul Bae, Hyunsook Kim, Jin-San Moon, Kun-Ho Seo, Properties of broiler breast meat with pale color and a new approach for evaluating meat freshness in poultry processing plants, Poultry Science, Volume 101, Issue 3, 2022, 101627, ISSN 0032-5791, https://doi.org/10.1016/j.psj.2021.101627.

[2] Qingying Luo, Xingyu Rong, Zhenkun Xiao, Xulin Duan, Yuan Zhou, Jie Zhang, Xiao Wang, Zhaoyuan Peng, Jianwu Dai, Yaowen Liu, Zhengfeng Fang, Effect of chitosan films containing clove essential oil-loaded microemulsions combined with deep learning on pork preservation and freshness monitoring, Food Control, Volume 168, 2025, 110914, ISSN 0956-7135, https://doi.org/10.1016/j.foodcont.2024.110914

[3] Zheng-Xu Fu, Yun-Cheng Li, Cai-Ying Zhang, Wei-Jun Chen, Fan-Bing Meng, Da-Yu Liu, Nontargeted metabolomics reveals dynamic changes in the quality of fresh yak meat during ice-temperature preservation, LWT, Volume 206, 2024, 116579, ISSN 0023-6438, https://doi.org/10.1016/j.lwt.2024.116579.

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

[4] D. -E. Kim, N. -D. Mai and W. -Y. Chung, "A IoT-Based Meat Quality Monitoring Using Camera and Gas Sensor With Wireless Charging," in IEEE Sensors Journal, vol. 24, no. 6, pp. 7317-7324, 15 March15, 2024, doi: 10.1109/JSEN.2023.3328915

[5] M. J. Oates, J. D. González-Teruel, M. C. Ruiz-Abellon, A. Guillamon-Frutos, J. A. Ramos and R. Torres-Sánchez, "Using a Low-Cost Components e-Nose for Basic Detection of Different Foodstuffs," in IEEE Sensors Journal, vol. 22, no. 14, pp. 13872-13881, 15 July15, 2022, doi: 10.1109/JSEN.2022.3181513.

[6] Saman Abdanan Mehdizadeh, Mohammad Noshad, Mahsa Chaharlangi, Yiannis Ampatzidis, AI-Driven Non-Destructive Detection of Meat Freshness Using a Multi-Indicator Sensor Array and Smartphone Technology, Smart Agricultural Technology, 2025, 100822, ISSN 2772-3755, https://doi-org.egateway.vit.ac.in/10.1016/j.atech.2025.100822.

[7] Michela Albano-Gaglio, Puneet Mishra, Sara W. Erasmus, Juan Florencio Tejeda, Albert Brun, Begonya Marcos, Cristina Zomeño, Maria Font-i-Furnols, Visible and near-infrared spectral imaging combined with robust regression for predicting firmness, fatness, and compositional properties of fresh pork bellies, Meat Science, Volume 219, 2025, 109645, ISSN 0309-1740, https://doi.org/10.1016/j.meatsci.2024.109645.

[8] Eko Prasetyo, Nanik Suciati, Chastine Fatichah, Aminin, Eric Pardede, Standardizing the fish freshness class during ice storage using clustering approach, Ecological Informatics, Volume 80, 2024, 102533, ISSN 1574-9541, https://doi.org/10.1016/j.ecoinf.2024.102533.

[9] Taoping Liu, Wentian Zhang, Mitchell Yuwono, Miao Zhang, Maiken Ueland, Shari L. Forbes, Steven W. Su, A data-driven meat freshness monitoring and evaluation method using rapid centroid estimation and hidden Markov models, Sensors and Actuators B: Chemical, Volume 311, 2020, 127868, ISSN 0925-4005, https://doi.org/10.1016/j.snb.2020.127868

[10] Hui Lu, Aiying Song, Ming Li, Xianqi Yao, Yuling Cai, Longlong Dong, Dacheng Kang, Yunguo Liu, Evaluation of the freshness (TVB-N) of pork patty during storage based on PLS-DA, SVM and BP-ANN models, Food Control, Volume 171, 2025, 111121, ISSN 0956-7135, https://doi.org/10.1016/j.foodcont.2024.111121.

[11] Jie Wang, Linlin Xia, Han Liu, Chong Zhao, Siyu Ming, Jingyi Wu, Colorimetric microneedle sensor using deep learning algorithm for meat freshness monitoring, Chemical Engineering Journal, Volume 481, 2024, 148474, ISSN 1385-8947, https://doi.org/10.1016/j.cej.2023.148474.

Name: PRANAY GORANTLA
REG.NO: 21MIS0123

[12] Wei Gong, Hong-Bin Yao, Tao Chen, Yu Xu, Yuan Fang, Hong-Yu Zhang, Bo-Wen Li, Jiang-Ning Hu, Smartphone platform based on gelatin methacryloyl(GelMA)combined with deep learning models for real-time monitoring of food freshness, Talanta, Volume 253, 2023, 124057, ISSN 0039-9140, https://doi.org/10.1016/j.talanta.2022.124057.

[13] Yao Zheng, Quantong Zhang, Xin Wang, Quanyou Guo, Classifying the freshness of large yellow croaker (Larimichthys crocea) at 12- and 24-hour intervals using computer vision technique and convolutional neural network, Smart Agricultural Technology, Volume 10, 2025, 100767, ISSN 2772-3755, https://doi.org/10.1016/j.atech.2025.100767.

[14] Xia Xu, Xinyu Wang, Yicheng Ding, Xuxia Zhou, Yuting Ding, Integration of lanthanide MOFs/methylcellulose-based fluorescent sensor arrays and deep learning for fish freshness monitoring, International Journal of Biological Macromolecules, Volume 265, Part 2, 2024, 131011, ISSN 0141-8130, https://doi.org/10.1016/j.ijbiomac.2024.131011.

[15] Mahamudul Hasan, Nishat Vasker, M. Saddam Hossain Khan, Real-time Sorting of Broiler Chicken Meat with Robotic arm: XAI-enhanced Deep Learning and LIME Framework for Freshness Detection, Journal of Agriculture and Food Research, Volume 18, 2024, 101372, ISSN 2666-1543, https://doi.org/10.1016/j.jafr.2024.101372.

[16] Dayuan Wang, Min Zhang, Qibing Zhu, Benu Adhikari, Intelligent vegetable freshness monitoring system developed by integrating eco-friendly fluorescent sensor arrays with deep convolutional neural networks, Chemical Engineering Journal, Volume 488, 2024, 150739, ISSN 1385-8947, https://doi.org/10.1016/j.cej.2024.150739.

[17] Yuandong Lin, Ji Ma, Jun-Hu Cheng, Da-Wen Sun, Visible detection of chilled beef freshness using a paper-based colourimetric sensor array combining with deep learning algorithms, Food Chemistry, Volume 441, 2024, 138344, ISSN 0308-8146, https://doi.org/10.1016/j.foodchem.2023.138344.

[18] Xinyi Liu, Ziling Li, Ziyu Wang, Xipeng Cui, Wenhao Li, Manman Wang, Yu He, BCNO quantum dots-based ratiometric fluorescence platform integrated with portable device: Hypoxanthine sensing for on-site assessment of meat freshness with deep learning, Chemical Engineering Journal, Volume 480, 2024, 147917, ISSN 1385-8947, https://doi.org/10.1016/j.cej.2023.147917.

[19] Mingxin Hou, Xiaowen Zhong, Ouyang Zheng, Qinxiu Sun, Shucheng Liu, Mingxin Liu, Innovations in seafood freshness quality: Non-destructive detection of freshness in Litopenaeus vannamei using the YOLO-shrimp model, Food

Chemistry, Volume 463, Part 1, 2025, 141192, ISSN 0308-8146, https://doi.org/10.1016/j.foodchem.2024.141192.

[20] Trinidad Perez-Palacios, Mar Ávila, Teresa Antequera, Juan Pedro Torres, Alberto González-Mohino, Andrés Caro, MRI-computer vision on fresh and frozen-thawed beef: Optimization of methodology for classification and quality prediction, Meat Science, Volume 197, 2023, 109054, ISSN 0309-1740, https://doi.org/10.1016/j.meatsci.2022.109054

[21] Z. W. Bhuiyan, S. A. R. Haider, A. Haque, M. R. Uddin and M. Hasan, "IoT Based Meat Freshness Classification Using Deep Learning," in IEEE Access, vol. 12, pp. 196047-196069, 2024, doi: 10.1109/ACCESS.2024.3520029.

[22] Min Li, Jianguo Xu, Chifang Peng, Zhouping Wang, Deep learning-assisted flavonoid-based fluorescent sensor array for the nondestructive detection of meat freshness, Food Chemistry, Volume 447, 2024, 138931, ISSN 0308-8146, https://doi.org/10.1016/j.foodchem.2024.138931.

[23] Yuandong Lin, Ji Ma, Da-Wen Sun, Jun-Hu Cheng, Chenyue Zhou, Fast real-time monitoring of meat freshness based on fluorescent sensing array and deep learning: From development to deployment, Food Chemistry, Volume 448, 2024, 139078, ISSN 0308-8146, https://doi.org/10.1016/j.foodchem.2024.139078.

[24] Guangzhi Wang, Yuchen Guo, Yang Yu, Yan Shi, Yuxiang Ying, Hong Men, ColorNet: An AI-based framework for pork freshness detection using a colorimetric sensor array, Food Chemistry, Volume 471, 2025, 142794, ISSN 0308-8146, https://doi.org/10.1016/j.foodchem.2025.142794.

[25] Muhammad Waqas, Zhengjie Chen, Yawar Abbas, Ambar Farooq, Xiaoxue Han, Hong Zhong, Xianwen Ke, Houbin Li, Xinghai Liu, Highly sensitive zinc oxide nanoparticle composite film with deep learning-assisted mobile technology for enhanced food freshness monitoring, Food Bioscience, Volume 62, 2024, 105541, ISSN 2212-4292, https://doi.org/10.1016/j.fbio.2024.105541

Name: PRANAY GORANTLA
REG.NO: 21MIS0123