Assessment 2

AI Project

You Only Look Once (YOLO)

INFT2060

Group Name

AI Rangers

Members

Harrison Woolley

Max Faulkner

Victoria Schiavo

Jake Whamond

Zarak Khan

Table of Contents

Executive Summary

YOLO is a pioneering deep learning model within the computer vision field of artificial intelligence. YOLO's distinctive one-stage detection approach enables real-time object detection, achieved by processing images and videos in a single pass through a convolutional neural network. It simplifies object detection by predicting attributes within a grid-based structure, undertaking a multi-stage process. This enables the model to achieve computational efficiency. YOLO's core components of bounding box prediction, class probabilities, and object scores allow for precise object identification.

Due to its real-time capabilities, if properly utilised within major disciplines in the future, YOLO has the potential to completely transform industries. This report will explore how these changes could affect the traffic management and agricultural fields in detail, displaying examples of potential objection detection tools that may one day be deployed and visualising their success with metrics and graphs to display their effectiveness and make conclusions.

Furthermore, this report will also explore the YOLO model's strengths and limitations, drawing conclusions from our own tests and research as well as discussing ethical concerns around the model. Lastly, we will explore the future avenues that the YOLO model may explore in the near future by observing its current development and its place in the computer vision realm as a whole.

Introduction

YOLO *(You Only Look Once)* is a computer vision model that is known for its efficient and accurate real-time object detection capabilities. It is unique in that it can identify objects in images in a single pass, unlike other models which require multiple passes to process an image successfully. This efficiency enables the model to be well-suited for usage in multiple applications and industries such as self-driving cars, surveillance, and additional real-time image and video applications. Following YOLO's initial conception, several versions of the model have been developed and released, bettering its speed and accuracy with each iteration. With these successes and improvements as it has evolved over time, YOLO has cemented itself as a pivotal model in the field of computer vision.

Background and Description

The YOLO model marks a substantial breakthrough in the field of object detection technology. It belongs to a class of one-stage object detection models capable of processing an entire image in a singular forward pass of a convolutional neural network. To understand the significance of YOLO, we can investigate the evolution of object detection models and exploring alternative models.

The evolution of detection models

Traditional object detection models typically utilised a two-stage approach in which the first stage identified regions of interest and the second stage classified objects within those identified regions. While this strategy was functionable, it was intensive on computer systems and not properly optimised for real-time applications.

YOLO model vs two-stage

Two-stage models require multiple passes through a neural network and post processing. YOLO condenses the detection process within a single pass, enabling for faster results. This speed is invaluable in real-time applications like surveillance, live video analysis, etc.

Alternative models

Single Shot Multibox Detector (SSD): This is a one-stage object detection model that is similar to YOLO. SSD combines multiple convolutional layers with varying scales to identify objects of different sizes and aspect ratios within the single forward pass.

RetinaNet: This is another one-stage object detection model that specialises in dense and small-scale objects and has gained notoriety for its balance of simplicity and accuracy. It is a popular object detection model in the field of aerial and satellite imagery.

YOLO Nano: This lightweight version of YOLO sacrifices accuracy for faster inference time and reduced model sizes, serving as an alternative for resource-constrained environments. It was created for deployment for edge devices and embedded systems.

Description of YOLO's components

Input processing: YOLO utilises an input image and divides it into a grid. Each cell in the grid is responsible for predicting bounding boxes and class probabilities for potential objects in that cell. This grid-based approach enables parallel processing which, in turn, speeds up the detection process.

Bounding box prediction: Within each grid cell, YOLO will predict multiple bounding boxes. These boxes have attributes of width, height, and centre locations. YOLO will also use anchor boxes to assist in the prediction allowing the AI to handle objects of various sizes and aspect ratios.

Non-maximum suppression (NMS): After the model generates predictions bounding boxes and class probabilities, a post-processing step known as NMS is applied to eliminate redundant and low-confidence detections. This process ensures only the most reliable and relevant predictions are retained.

Backbone convolutional network (CNN): YOLO utilises a deep convolutional neural network as its backbone architecture. YOLO utilises CNN to extract features from the input image and perform object detection tasks effectively.

Application and Impact

Traffic Management

The YOLO model's remarkable adaptability in functioning effectively on a variety across a variety of different hardware platforms, as well as its capacity to customise models for different specific tasks makes it versatile for deployment in various contents, wielding the power to transform various distinct industries.

One area where the YOLO model has the potential to see large usage is within the traffic management industry. This could include for enhanced safety, traffic monitoring or management, automated toll collection, parking, traffic enforcement, or within public transport.

One of the primary areas that YOLO models may see success in this industry is through number plate recognition. YOLO can be integrated into some of the current infrastructure already existent, such as within CCTV hardware. The model can find utility within smart cities by being integrated with CCTV system for the purpose of detecting number plates on motor vehicles involved in criminal or suspicious activities. It can also serve as an extension of the pre-existing automatic number plate recognition system used for traffic enforcement.

Therefore, the integration of YOLO into the traffic management industry has the potential to massively enhance the effectiveness of applications across multiple different domains, with a particular emphasis on its possible impact on number plate recognition.

Agriculture

In the future, the YOLO model holds considerable promise in addressing a wide range of tasks within the agriculture and farming fields in a variety of different areas. These potential applications may include monitoring the health of crops and providing information regarding their spacing, along with other relevant factors. YOLO's object detection capabilities can also extend to weed detection, aiding in identifying unwanted growths, as well as pest and disease detection. Livestock monitoring could be enhanced by YOLO's expertise, offering animal classification and health assessment. This ability to identify potential issues immediately would greatly impact the agricultural industry, allowing for adequate time to rectify these detected issues.

As technology continues to expand, the prospects of deploying YOLO into this sector becomes increasingly more plausible with the model's versatility allowing it to provide insights, enhance efficiency and a boost to overall productivity in this industry.

Sports

Another area YOLO models have the potential to find success is in professional sports, where they can enhance the fan experience, assist in sports management and aid in game analysis. Areas YOLO models could be employed include ball tracking, where the trajectory and the movement of balls in multiple sports can be observed; and player tracking, allowing for valuable insights into player performances and movement patterns. Graphic depictions of this information such as heatmaps can even be generated, which could assist coaches in making decisions best suited for the team.

They could also play a critical role in decision-making processes for other parties like referees. For example, YOLO could assist in determining rulings on offside, an area of refereeing prone to human error.

The incorporation of YOLO models in the professional sports industry is liable to grow in the future as they become more prominent in other sectors, having the potential to provide valuable insights and enhance the overall experiences for all participants alike.

Product Manufacturing

The manufacturing industry is another sector that stands to benefit from the addition of YOLO models. Areas within this industry the YOLO model could be integrated include quality and defect detection, where flaws in products and packaging can be detected and mitigated through quality control; or packaging and labelling to verify the accuracy of labels and packaging to the product, ensuring items are correctly labelled, packaged and sorted.

YOLO's introduction in the product manufacturing industry could revolutionise the industry with its ability to improve quality control throughout the manufacturing process, allowing for increased productivity and a reduction in errors produced.

Experimental Evaluation

Industry 1. Traffic Management

In our exploration of how the usage of YOLO models may positively impact the traffic management industry, we have selected a **vehicle number plate detector** model for experimentation purposes. This model is designed to detect and identify all number plates within its field of view, assigning them a label and a confidence score. During our testing of this model, we experimented using a diverse range of datasets and images, recording the results and created visuals to illustrate the model's effectiveness.

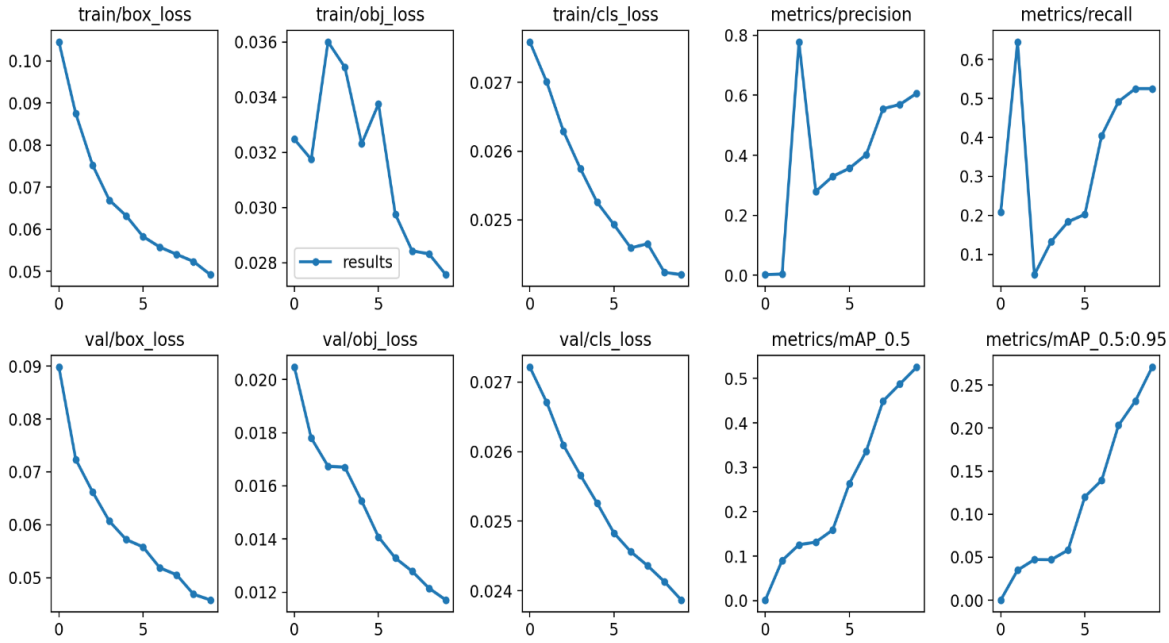
Our process began with compiling a comprehensive dataset we could test the model on, including various photos of vehicles taken in different settings, distances and angles to assess the model's performance by. After this dataset was created and its capabilities tested, we proceeded to go through the process of employing different metrics to draw conclusions about the model's performance and accuracy.

As our evidence shows, the model checks for key elements of interest, such as the license plate itself, including its number, and the background (or non-relevant parts of the image).

Metrics

epoch	train/box_loss	train/obj_loss	train/cls_loss	metrics/precision	metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.95	val/box_loss	val/obj_loss	val/cls_loss	x/lr0	x/lr1	x/lr2	metrics/license-plate_mAP	metrics/license-plate-number_mAP	metrics/license-plate_mAP_50	metrics/license-plate-number_mAP_50
0	0.10456	0.03247	0.027592	0.002387	0.20886	0.001611	0.000352	0.089795	0.02047	0.027215	0.082	0.002	0.002	0.000361	0.000343	0.0017097	0.0015121
1	0.087494	0.031743	0.027009	0.004446	0.64557	0.090405	0.03495	0.072362	0.017804	0.02671	0.062694	0.003694	0.003694	0.013868	0.056031	0.048237	0.13257
2	0.075266	0.036002	0.026289	0.77804	0.047687	0.12581	0.047261	0.06622	0.01674	0.026091	0.042972	0.004972	0.004972	0.024891	0.069632	0.085088	0.16653
3	0.06692	0.035081	0.025747	0.27945	0.13291	0.13176	0.047083	0.060678	0.0167	0.025664	0.022835	0.005835	0.005835	0.05306	0.041106	0.12484	0.13868
4	0.0632	0.03231	0.025254	0.32955	0.18354	0.15893	0.05828	0.057209	0.015429	0.025259	0.00604	0.00604	0.00604	0.073455	0.043104	0.17787	0.14
5	0.058278	0.033751	0.02493	0.35671	0.20253	0.2637	0.12011	0.055818	0.014075	0.02483	0.00604	0.00604	0.00604	0.1764	0.063821	0.3235	0.2039
6	0.055761	0.029754	0.024591	0.40242	0.40506	0.33664	0.1394	0.05187	0.013283	0.024561	0.00505	0.00505	0.00505	0.17228	0.10651	0.38726	0.28603
7	0.054026	0.028424	0.024651	0.55517	0.49214	0.44908	0.20311	0.050544	0.012784	0.02436	0.00406	0.00406	0.00406	0.2372	0.16901	0.49764	0.40053
8	0.052341	0.028317	0.024235	0.56924	0.52532	0.48735	0.23133	0.046884	0.012147	0.02413	0.00307	0.00307	0.00307	0.33222	0.13044	0.59122	0.38348
9	0.049208	0.027562	0.024201	0.60667	0.52532	0.52538	0.27072	0.045798	0.011703	0.023867	0.00208	0.00208	0.00208	0.3772	0.16424	0.63103	0.41973

Above is a table of results of various tests we ran on the YOLO model. The next image will display the visualisations of the data, as well as our observed evaluations.



Train/box loss – This graph demonstrates the model's training process as it is exposed to the dataset. During this process, after each epoch, the model's parameters are updated to minimise loss with the goal of enhancing its ability to detect license plates. The 'box loss' in this graph, which pertains to bounding box predictions, provides insight into how well the model is learning to predict the coordinates of the license plates in the images. Over the course of our 10-epoch test, there is a clear trend in the data, showing a consistent decrease in the loss value. This is a positive indicator during the training as it signifies the model's improve proficiency in locating license plates in images.

Train/obj loss – In this graph, we can visualise the object loss during the training process of the model. This visual representation allows us to determine the model's ability to discern he presence of objects within specific grid cells. We can determine, based on the graph, that there is a trend in the data, indicating a consistent reduction in object loss throughout the training process. This tells us that the YOLO model is improving within its ability's to successfully identify the presence or absence of objects within grid cells.

Train/cls loss – Within this graph, we can see the classification loss of the model during its training. The results of this graph can help inform us of the model's capabilities to determine if an object is a license plate or if it is something else. In the results of this graph, there is a clear decrease in the loss each epoch. This is a positive result as it displays that the YOLO model is improving in its ability to accurately classify and correctly labelling the objects within the provided images.

Metrics/precision – In this graph, we can observe how the precision metric evolves across different thresholds to evaluate the model's performance. When the threshold decreases the recall rate tends to increase, but with a potential decrease in precision as a trade-off. This graph allows us to visualise this trade-off, allowing us to draw conclusions about which threshold aligns with the model's objectives the best. The results depicted in this graph showcase that when using Threshold 2, the model achieves the highest precision of all thresholds observed at 0.77840. At this confidence level, we can say that the model makes more accurate predictions.

Metrics/recall – In this graph, we can observe how the recall metric may vary depending on the threshold. We can see that Threshold 1 had the highest recall at 0.64557. Therefore, that threshold is the most optimal choice to maximise recall.

Val/box loss – During the validation stage, we can visualise the bounding box loss from the model. In comparison to the training graph, there is a clear improvement in results decreasing each epoch which communicates to us that the mode is localising objects with high accuracy during the validation process.

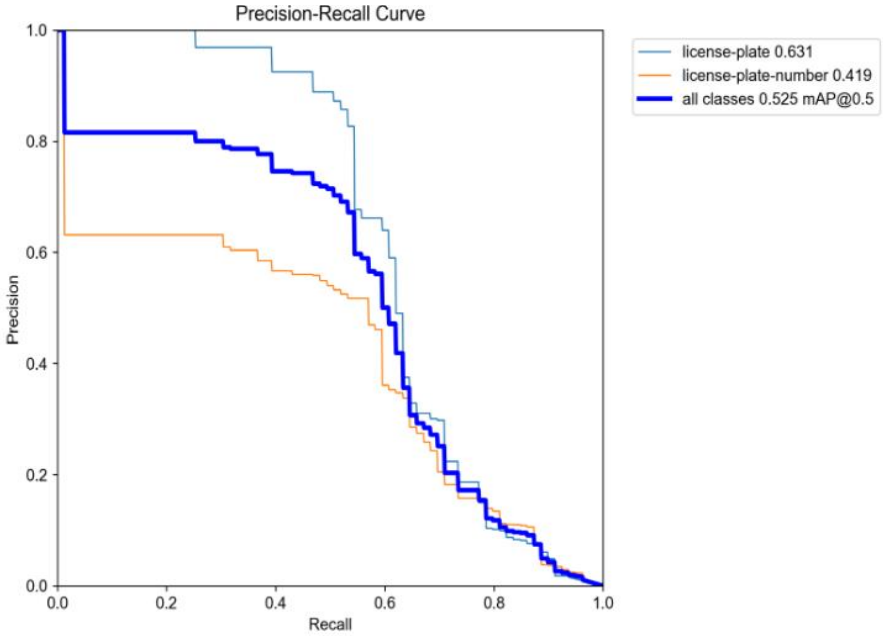
Val/box obj – This graph provides insights into how the model performs object identification during the validation phase. We can compare it to the training graph and perceive a trend of loss, declaring that it is getting better at predicting the presence or absence of objects within grid cells.

Val/cls loss – This graph details the classification loss during the validation phase. We can compare this graph to the corresponding training graph and see a slight improvement. Furthermore, within this graph we can see positive results as the graph continually decreases. This informs to us that the model is improving at correctly identifying and classifying the objects it detects.

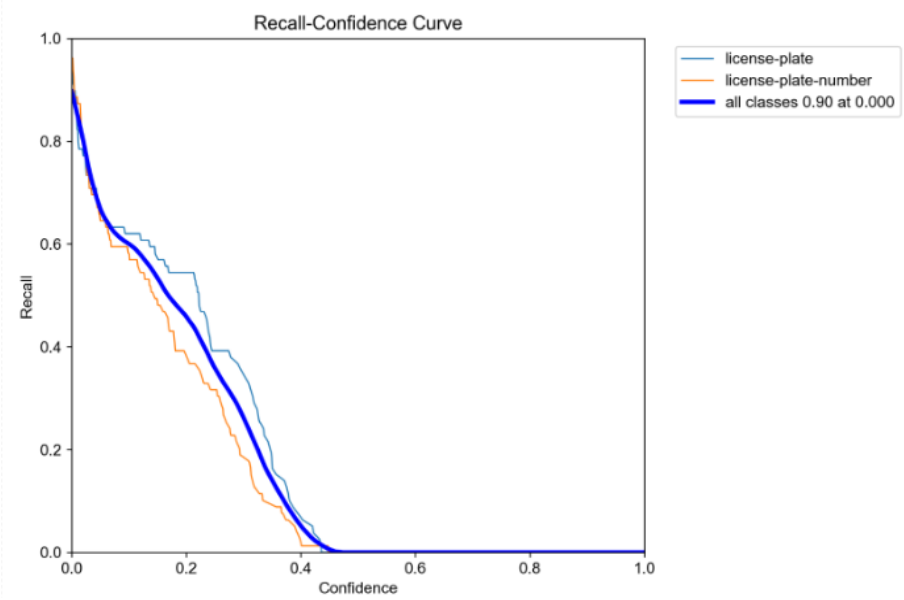
Metrics/mAP 0.5 – In this graph, we can determine the average precision *(mAP)* at the intersection over union *(IoU)* threshold of 0.5. During this test the highest mAP at IoU 0.5 was 0.52538 on threshold 9. We can surmise that the model can accurately detect and classify objects within images at a decent level of accuracy.

Metrics/mAP 0.5 : 0.95 – This graph is similar to the last with the main difference being its mAP score of 0.95. For our results, the highest score observed was Threshold 9 with a mAP at IoU 0.5 of 0.27272. This is a positive score as it indicates that the model is performing well at detecting and classifying objects within images at a high level of precision and recall.

The following graphs will demonstrate the many metric curves of the models. These graphs provide a visual representation of how our model trade-offs change as the classification thresholds are adjusted and gives us information on what threshold works best for the model's functionality:

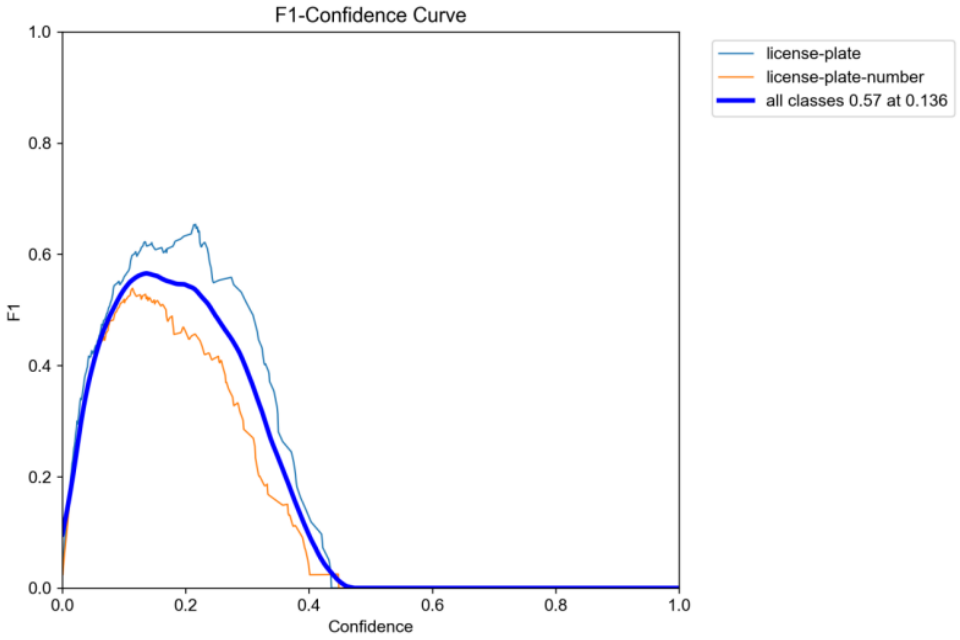


In this graph, we can see the results of a precision/recall curve. While the model makes high precision predictions, it is missing a significant number of positive instances due to a low recall. The graph then drops to 0.8 precision with a recall around 0.4-0.5, suggesting that the threshold is more inclusive. It is able to capture true positives, but inadvertently allows a higher number of false positives, which leads to a visible decrease in precision. As the recall increases, the precision slowly drops. This reinforces the fact that the model will catch almost all positive instances, but at the cost of generating a fair number of false positives.

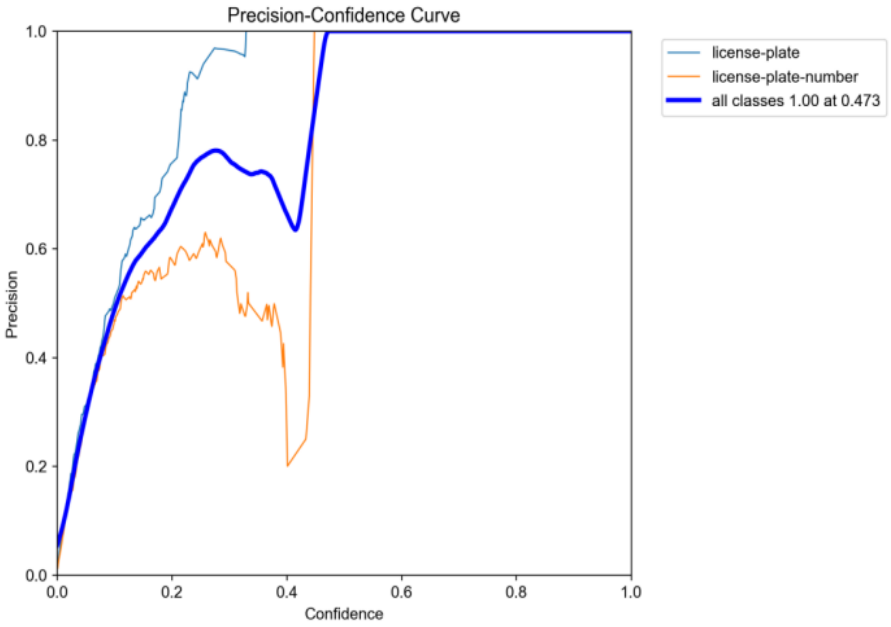


This graph illustrates the relationship between recall and confidence and their correlation. The model initially exhibits high recall, indicating that the model at this threshold effectively

identifies nearly all positive instances in the dataset, but at a low confidence. As recall drops to 0.06 and confidence from 0.2 to 0.3, it suggests the model becomes more confident in its predictions at the cost of missing some positive instances, resulting in a lower recall rate. As the threshold for higher confidence is reached, the recall drops further. The model runs into the same problem as mentioned before, but with an increasing number of positive instances being missed.



At the beginning of the graph the threshold allows for a low f1 score of 0.2 but quickly grows to a 0.6 with a confidence of 0.2, this indicates that the model at this threshold will make predictions that will become more reliable and gains an ok balance between precision and recall. As the confidence increases however the f1 score slowly drops suggesting that the model is confident in its predictions and loses the balance between precision and recall.



This graph displays the relationship between precision and confidence. The threshold begins a low precision of 0.2 and equally low confidence. It then quickly rises to a precision of 0.8 and a confidence of 0.3 to 0.4, indicating the model is making more precise and confident predictions. As confidence levels continue to increase to around 0.5, the precision reaches 1.0, indicating that the model is making decently confidence predictions with a high level of precision, meaning most of its predictions are correct.

Traffic Model: Inference

After successfully training the model, we ran inference on the YOLO model by feeding the model with data that the model was not trained with. Examples of the inference's results can be found in the images below:

As these images showcase, the model has done a reasonable job at locating the number plates. It yields high confidence predictions ranging from 0.80 to 0.85 out of a maximum score of 1.

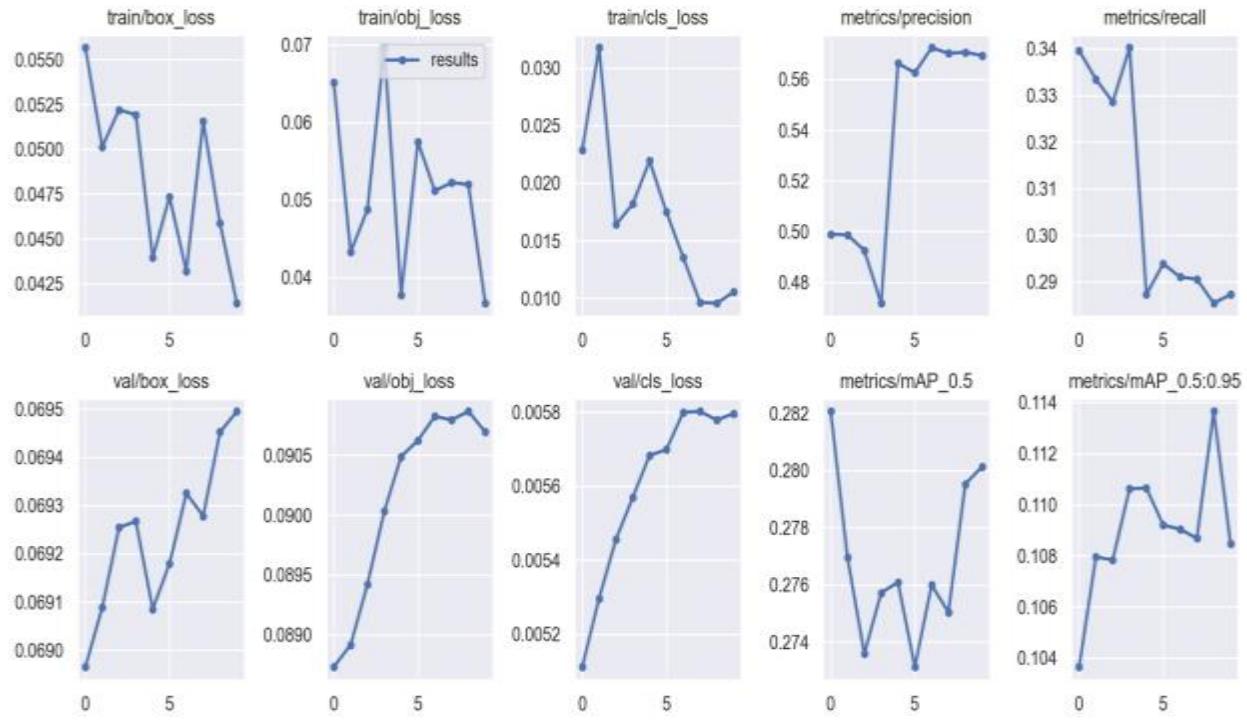
From this, it becomes evident that the model has effectively learnt from its training and is able to successfully apply that knowledge to previously unseen images. As we will dive further into the next section, this result validates our hypothesis that a comparable model could be implemented in a real-world scenario and integrated with existing ALPR infrastructure.

Industry 2. Agriculture and Farming

The YOLO model we have selected for experimentation purposes that could potentially positively impact the farming and agriculture industry is a **cashew disease identification detector** model. This model is designed to detect multiple objects such as insects or disease within its field of view, identifying them with the correct label. Through our testing with this model, we experimented with a varied range of datasets and images, recording the results and created visual to help illustrate the model's effectiveness.

Our process during this phase began with compiling an appropriate and comprehensive set of images we could use as a dataset for testing purposes which contained different pictures of plants, diseases, insects and abiotic in numerous settings, distances and angles. After this dataset was created and its capabilities tested, we proceeded to go through the process of employing different metrics to draw conclusions about the model's ability to correctly assign multiple objects with the labels *abiotic*, *insect* and *disease*.

Metrics



Train/box loss – This graph demonstrates the model's training process as it is exposed to the dataset. During this process, after each epoch, the model's parameters are updated to minimise loss with the goal of enhancing its ability of identifying labels for the objects in the images. The 'box loss' in this graph, which pertains to bounding box predictions, provides insight into how well the model is learning to find the coordinates of the objects within the provided images. Over the course of our 10-epoch test, we can decipher a trend in the data that shows a staggered decrease from 0.0550 to 0.0425. This is a positive indicator during the training as it signifies the model's improve proficiency in locating and labelling objects in the images.

Train/obj loss – In this graph, we can visualise the object loss during the training process of the model. Through analysis of this graph, we can determine the model's ability to discern he presence of objects within specific grid cells. There is a staggered line that generally

decreases overtime during the 10 epochs. While the results do have spikes, the graph does show a decrease compared to the first test and showcasing that model has made improvement in its ability to successfully identify the presence or absence of objects within grid cells.

Train/cls loss – Within this graph, we can see the classification loss of the model during its training. The results of this graph can help inform us of the model's capabilities to determine the correct classification for an identified object. In the graph, there is a clear decrease over the epochs giving a positive result as the model is improving in its ability to accurately classify and correctly labelling the objects within the provided images.

Metrics/precision – In this graph, we can observe the progression of how the precision metric evolves across different thresholds to evaluate the model's performance. Generally, if the precision is high in a model, the recall is not. The model showcases a steady increase over time with 0.6 being the highest precision reached, indicating the model is about 60% precise in its predictions.

Metrics/recall – In this graph, we can observe there is a slight decline of the recall rate over the testing. This indicates that the model's ability to correctly identify true positive instances has decreased. There could be multiple factors that could have resulted in this decrease include precision, quality of the dataset used or the complexity of the model.

Val/box loss – During the validation stage, we can visualise the bounding box loss from the model. It remains stable in this graph and does not improve or worsen significantly over the training process. While 0.6 is not a desired result for bounding box loss, this result is higher than the training set. This could mean that overfitting or a dataset mismatch could be occurring.

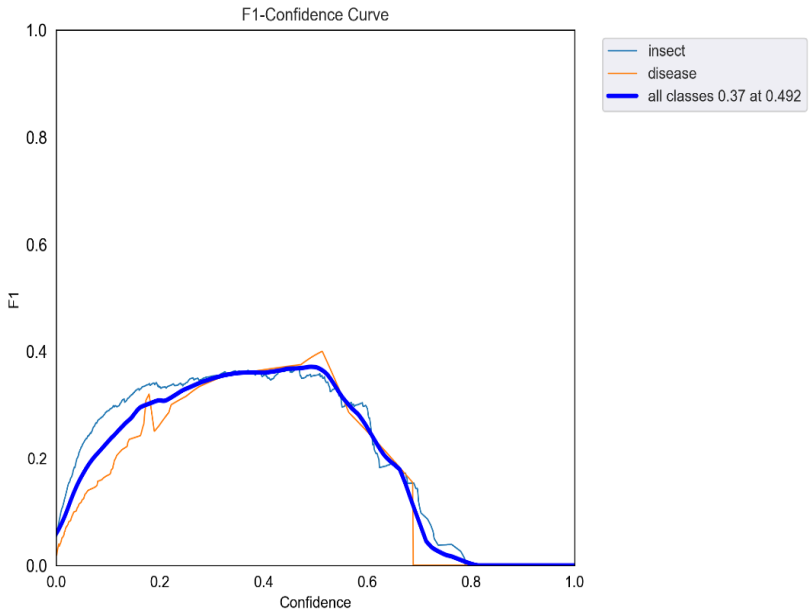
Val/box obj – This graph provides insights into how the model performs object identification during the validation phase. We can see a negative result as the object loss is extremely high at 0.8/0.9, displaying that the model objection detection is in a bad state. This, again, could be the result of either overfitting or bad data quality between training and validation data.

Val/cls loss – This graph details the classification loss during the validation phase. The results show that the classification loss sits at around 0.5/0.6, indicating that the model is struggling to correctly classify objects in the dataset, likely due to the aforementioned factors.

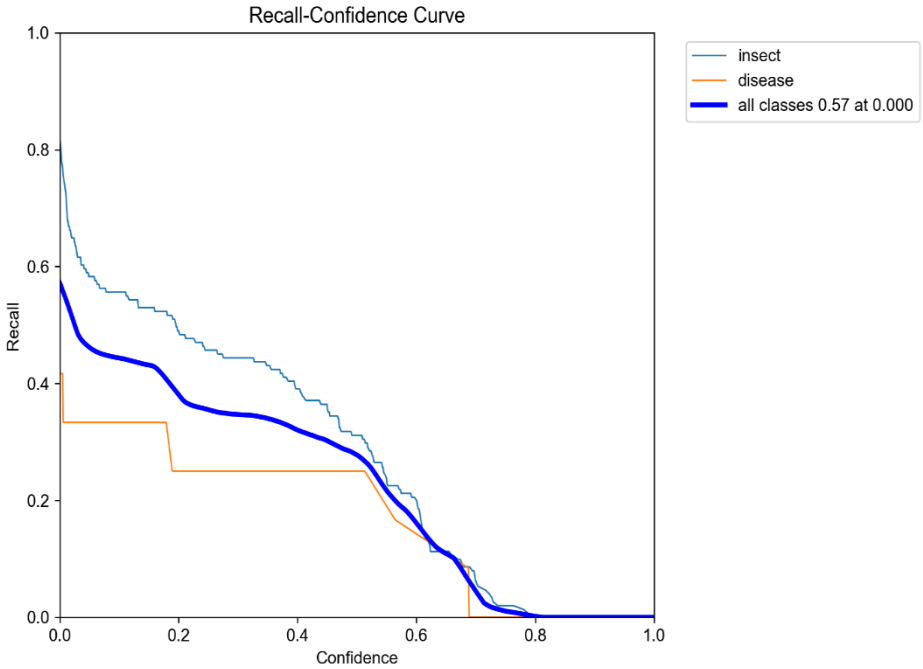
Metrics/mAP 0.5 – In this graph, we can determine the highest mAP at the IoU score achieved was 0.282, which is relatively low. This ties back into the issues identified above.

Metrics/mAP 0.5 : 0.95 – With the high score of 0.114, this graph presents the same issue as the last as the model is having issues identifying and classifying images within the images provided.

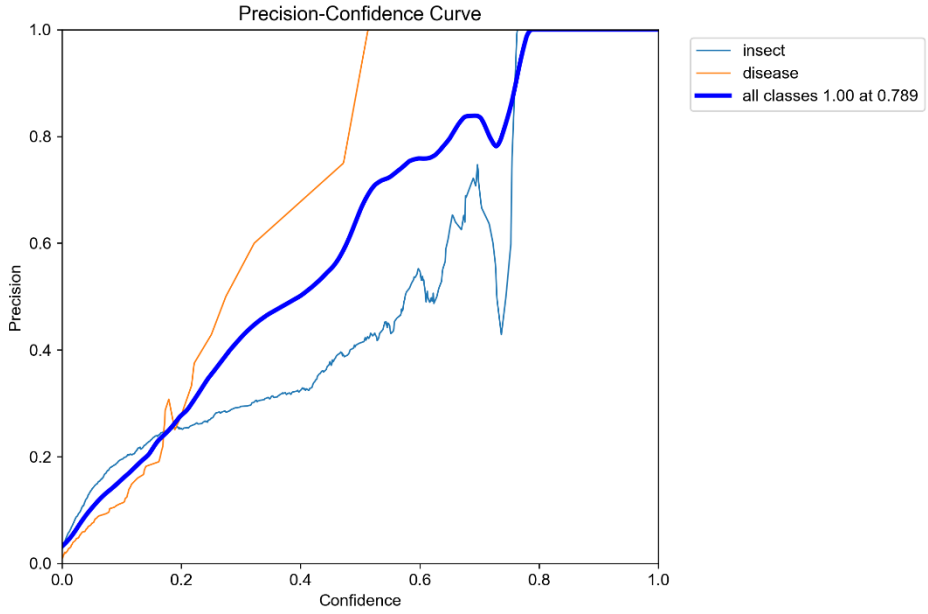
The following graphs will demonstrate the many metric curves of the models. These graphs provide a visual representation of how our model trade-offs change as the classification thresholds are adjusted and gives us information on what threshold works best for the model's functionality:



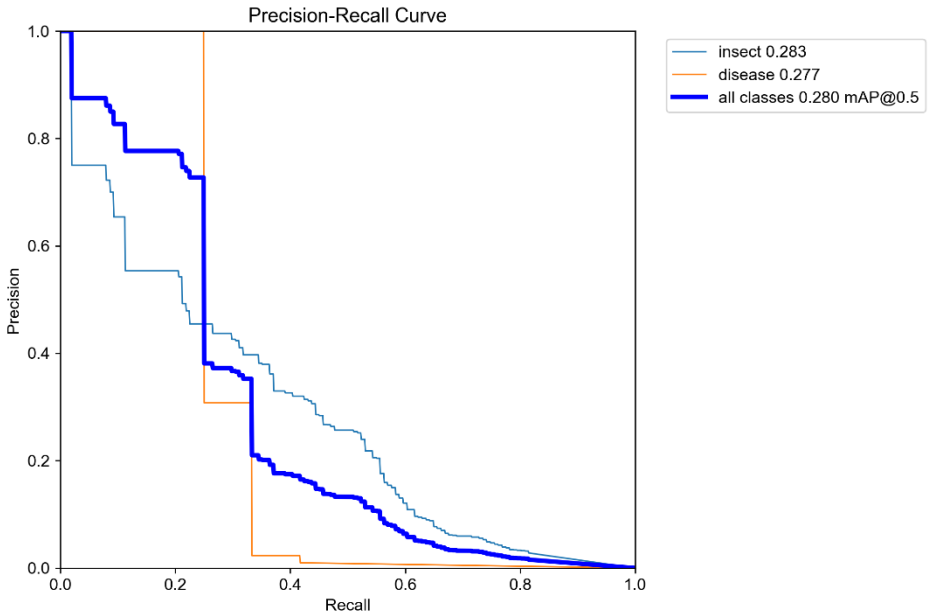
This graph provides a visual presentation of the relationship between the F1 score and the model's confidence. Within the graph, the F1 starts at 0.2 and quickly rises to 0.4. With the confidence rising from 0.4 to 0.6, the two parameters are decently balanced with one another. However, as the confidence rises furthermore, the F1 score begins to decline, ending in a low score. This trend results in an undesirable recall and precision scores as the model becomes overly confident in its predictions.



Using this graph, we can visualise the relationship between recall and confidence. Initially, the model begins with a balance between a recall of 0.6 and a confidence 0.4 to 0.6. Like the last graph, as the confidence rises to 0.8, the recall rate drops to a low score. The model is confident in making predictions but is still missing many positive instances.



This graph illustrates the correlation between the model's precision and confidence. A steady increase in both areas can be seen until the precision reaches 1.0 and confidence reaches 0.8. The model makes high precision with a high amount of confidence, but suffers from a trade-off with a low recall rate. Therefore, despite their heights, the model may still miss several relevant instances.



This graph showcases the relationship between precision and recall, conveying that the model only produces positive predictions when it has a high level of confidence. While it rarely makes mistakes in assigning labels to objects, it is liable to miss many positive instances due to a low recall rate. In the middle portion of the graph, it becomes evident that as the classification threshold is lowered, the model starts to make positive predictions which results in an increase of recall. At the end of the curve, the model operates high recall and low precision. This means the model classifies nearly all instances as positive, inadvertently leading to a higher number of false positives.

Cashew Plant Model: Inference

After successfully training the model, we ran inference on the YOLO model by feeding the model with data that the model was not trained with. Examples of the inference's results can be found in the images below:

As we can determine from these images, the model's performance in correctly identifying and labelling objects it is designed to look for has been inconsistent. Furthermore, there are notable variations in the probability scores it assigns, raising several important observations regarding the model's behaviour.

In images where the model demonstrates better performance, a common attribute is the presence of larger and more easily distinguishable object boxes. In such cases, the model tends to identify and label the objects correctly with high probability scores, showcasing precision and confidence in its detections. This suggests a resemblance between these images and the data in which this model was trained on.

When examining the remaining images, however, the model struggles to correctly identify objects, resulting in a greater number of false positives. The difficulties faced in these scenarios can be attributed to not only the complexity of the task at hand in identifying disease in cashew farming, which appear subtle and can vary widely, but also a lack of variance in images present in the dataset in which the model was trained on.

Based on these insights, this model is better suited for deployment in areas with well-organised cashew crops featuring larger and more distinguishable object boxes where it can leverage its strengths more effectively.

Conclusion

Our experiment aimed to assess the performance of two industry-specific YOLO models, one designed for traffic number plate recognition, and the other for cashew tree disease detection. During our testing, we have been able to gain insight into the practicality and capabilities of both these detection models.

Regarding the number plate detection model, it showed potential for future real-world implementations as it delivered an acceptable level of accuracy in identifying license plates in various images within the dataset. The high performance of this model suggests that a model of a similar specification could be implemented in the traffic management industry to evolve current systems. However, further refinement and fine tuning may be necessary to achieve an even higher level of performance so the model is able to adjust to the challenging conditions it may face in action when identifying number plates in real time.

In contrast, the cashew disease tree detection model encountered significant challenges in correctly identifying objects, often misclassifying them. The frequency of these errors have a great effect on the model's potential in real world application, which raises the question of whether this task is truly well-suited for a YOLO model to handle as the results for this particular model need to be precise in its classification of objects.

These errors in testing indicate a crucial need for a more varied and diverse dataset in which the model can draw its knowledge from. From observing the provided dataset's images, many of them were taken in conditions where objects that could be classified as either *abiotic*, *insect* or *disease* were not distinct from one another. If further testing were to be conducted, images that contain cashew trees in full view and better quality that clearly showcase the classifiable objects in detail would be worth adding to the training dataset to help ensure the model can perceive the varied types of disease symptoms on the trees.

To conclude, our experiment highlights the potential of YOLO-based models in these specific industries. These findings could help guide future investigation into the introduction of YOLO models in these industries and assist in answering numerous questions that may be raised.

Advantages and Limitations

Strengths:

The YOLO model excels in environments where objects have distinct and easily recognizable characteristics, particularly when trained on datasets that closely resemble the objects it aims to identify. This is evident in applications such as license plate detection, where the YOLO model performs well.

The areas where the YOLO model displays strengths that we have seen during our testing include:

- **Real-time object detection –** YOLO models can excel in real-time object detection, making them suitable for applications where low latency is important like the traffic management industry.
- **Object localisation –** This results in accurate bounding box coordinates, as seen in our number plate model.
- **Multi-class detecting –** The model can detect multiple instances of the same object, plus different objects, within datasets and label them accordingly based on the model's familiarity with them.
- **Efficiency –** Its single-pass architecture allows for resource-efficient in both time and processing of images.
- **Adaptability and versatility** – The greatest strength of YOLO models is that they can be trained for a wide variety of object detection tasks in multiple different domains.

During testing, the strong qualities of YOLO models was best showcased in the number plate detection model, which demonstrated the architectures skills to correctly identify objects within a wide range of images and conditions making it a good example to display the ability YOLO models have in terms of object detection tasks.

Limitations:

The YOLO model exhibits notable limitations, particularly in scenarios where objects possess similar or overlapping characteristics. This is best evident when considering the cashew plant model, where the YOLO model frequently produced false positives, particularly when assigning *disease* or *abiotic* classes to objects.

The areas where the YOLO model displays limitations that we saw in our testing include:

- **Limited accuracy in complex scenarios** – In cluttered scenes, as present in our evaluation of the cashew plant model, the model encountered significant difficulties in object detection and location determination. This heavily impacted the accuracy of its deductions, leading to numerous false positives. Overall, this impacts its usability and necessities more manual intervention from that of users, defeating the purpose of streamlining the classification process.
- **Small object detection –** For smaller objects, such as insects, the model struggled to locate them. Overlapping objects also posed challenges for bounding box predictions for some images.

The cashew disease detection model prominently showcased the majority of YOLO's limitations during our evaluation process, making it clear that model suffers considerably from the issues identified above. While rectification is possible, the scope of the task may mean that it is outside of YOLO's scope as a computer vision model.

Ethics:

The ethical implications surrounding artificial intelligence *(AI)* represent a complex and relatively recent issue in moral discussions. The YOLO model, which possesses a remarkable ability to identify objects across various applications, brings forth a plethora of ethical dilemmas due to its wide-ranging potential. From supporting governments in the logistics of mass surveillance to identifying illicit opium poppy crops and license plates on vehicles, the YOLO model stands at a complicated intersection of technological capability versus ethical consideration through potential misuse.

In the context of mass surveillance, where the model can be employed to monitor public spaces, questions arise about the extent to which such surveillance infringes upon individual privacy rights. YOLO's ability of identification raises profound concerns about privacy and surveillance practices.

Employing YOLO in identifying specific crops or objects, like opium poppy plants, presents a moral quandary. While it can aid in combating illegal activities, it presents issues related to environmental and societal impacts. Identifying illegal crops may lead to punitive actions impacting livelihoods and local communities. Balancing between law enforcement needs, broader societal welfare and economic stability becomes a central ethical challenge in these scenarios.

The accuracy and reliability of the YOLO model in differentiating between similar objects or situations add another layer of ethical complexity. False positives or misidentifications can have severe consequences, particularly in legal contexts. Inaccurate identifications may lead to wrongful arrests, legal actions, or damaged reputations, highlighting the imperative to ensure the model's precision and fairness in its applications. For the YOLO model to overcome this ethical issue, attempts should be made to disclose these identified problems, ensuring the model has been sufficiently tested to avoid issues such as misidentifications in more private areas the model may be utilised.

Resolving ethical challenges posed by the YOLO model requires collaborative effort involving policymakers, technologists, and ethicists. Establishing robust frameworks in developing AI applications is crucial. These frameworks should encompass the technical aspects of AI, privacy, and wider societal impact. The design and deployment of AI systems should ensure fairness, transparency, and accountability in decision-making processes. Interdisciplinary research that involves experts from fields such as ethics, law, sociology, and technology is vital to anticipate and mitigate the potential risks and challenges associated with AI advancements. Continued research and development efforts should focus on enhancing the accuracy, fairness, and interpretability of AI models like YOLO.

The rise of YOLO models has indeed raised some significant ethical concerns that must be addressed to maintain ethically sound and trustworthy applications across multiple industries.

Future Directions and Conclusion

The field of object detection has seen significant advancements with the development of YOLO. As we look forward towards the future, several trends and challenges become readily apparent.

Emerging from the realm of YOLO, we can observe several trends. A crucial one is that of real-time object detection, which is prominent and evident in recent YOLO models. Areas demanding low-latency object detection include robotics and self-driving cars, which continue to push the boundaries of what YOLO could offer in a positive and constructive manner. Efficiency and optimisation represent another trend as there is a growing emphasis in ensuring YOLO models are optimised for maximum efficiency, involving endeavours such as model pruning or architectural modifications to create lightweight models and models available for edge devices. Domain or industry-modified models, which were subject to exploration in this report, are another example of a notable trend. This involves the adaption of training data and specific model architecture. The final area of interest is multi-model objection detection, which integrates YOLO object detection technology with other technologies like radar sensors.

Despite their achievements as an important cornerstone in computer vision and objection detection technology, the YOLO model still faces several challenges. Small object detection, as already highlighted, is a major issue best exemplified during our testing of the cashew model where it struggled to identify *insect* objects. Diverse lighting and weather conditions may also pose issues, especially in outdoor or uncontrolled environments, particularly when a model has no frame of reference of the object it aims to identify would appear in those conditions. Scalability and resource constraints emerge lastly as an issue, prominent when YOLO models are deployed on devices suffering from resource constraints, highlighting the need to accommodate for different hardware and deployment scenarios.

As YOLO models continue to grow in popularity, several directions it can take in the future can be discerned. Semi-supervised or self-supervised learning could help reduce YOLO's reliability on an extensive and annotated dataset, holding promise for object detection to become more accessible for applications with limited training data. Continual learning would enable models to adapt incrementally to changing environments. Privacy and ethical concerns, particularly concerning the responsible use of object detection technology in areas such as facial recognition, will remain a focal point of concern in future developments, most notably when it also becomes an issue of legality.

In conclusion, as object detection continues to evolve, the path ahead is balanced between addressing existing challenges and exploring emerging trends. The real-time capabilities, efficiency and flexibility of YOLO models positions them as formidable contenders across a wide range of application and industries. Our research experience with YOLO models has demonstrated their object detection abilities, best demonstrated with the number plate detection model. However, it also underscores the need for ongoing refinement in more complex scenarios, as demonstrated by the cashew disease model. YOLO models are at the crossroads in innovation and are poised to change the future, provided they address their persistent challenges and stay on top of emerging trends, sticking to the path of enhancing their utility across multiple industries and applications.

References

Diwan, T., Anirudh, G. and Tembhurne, J.V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*. Doi: https://doi.org/10.1007/s11042-022-13644-y.

ieeexplore.ieee.org. (n.d.). *Object Detection and Tracking Using Yolo*. [online] Available at: https://ieeexplore.ieee.org/abstract/document/9544598.

ieeexplore.ieee.org. (n.d.). *Surveillance System using Moving Vehicle Number Plate Recognition*. [online] Available at: https://ieeexplore.ieee.org/document/10212426

Park, S.-S., Tran, V.-T. and Lee, D.-E. (2021). Application of Various YOLO Models for Computer Vision-Based Real-Time Pothole Detection. *Applied Sciences*, 11(23), p.11229. doi: https://doi.org/10.3390/app112311229.

Sahota, H. (2023). *The History of YOLO Object Detection Models from YOLOv1 to YOLOv8*. [online] Deci. Available at: https://deci.ai/blog/history-yolo-object-detection-models-from-yolov1-yolov8/.

Sahota, N. (2023). *Object Detection 101: Applications, Challenges, and Future Directions*. [online] Neil Sahota. Available at: https://www.neilsahota.com/object-detection-101-applications-challenges-and-future-directions/.

Models and datasets used:

- Number plate dataset - https://universe.roboflow.com/jodaryle-factor-gjkmr/license-plate-object-detection/dataset/6
- Cashew disease dataset:- https://huggingface.co/datasets/KaraAgroAI/CADI-AI/tree/main
- Number plate model - https://huggingface.co/keremberke/yolov5n-license-plate
- Cashew plant model - https://huggingface.co/KaraAgroAI/CADI-AI

Appendix A – Python Code

Traffic - Inference

```
import yolov5

# load model

model = yolov5.load('keremberke/yolov5s-license-plate')

# set model parameters

model.conf = 0.25  # NMS confidence threshold

model.iou = 0.45  # NMS IoU threshold

model.agnostic = False  # NMS class-agnostic

model.multi_label = False  # NMS multiple labels per box

model.max_det = 1000  # maximum number of detections per image

# apply dataset

img = '…./../...'

# perform inference

results = model(img, size=640)

# inference with test time augmentation

results = model(img, augment=True)

# parse results

predictions = results.pred[0]

boxes = predictions[:, :4] # x1, y1, x2, y2

scores = predictions[:, 4]

categories = predictions[:, 5]

# show detection bounding boxes on image

results.show()

 save results into "results/" folder

results.save(save_dir='./results/')
```

Traffic - Training

```
!yolov5 train --data data.yaml --img 640 --batch 16 --weights
keremberke/yolov5s-license-plate --epochs 10
```

Cashew Plant Model - Inference

```
import yolov5

model = yolov5.load("KaraAgroAI/CADI-AI")

# Images

img = ['../../..']# dataset

# set model parameters

# set Non-Maximum-Suppression(NMS) threshold to define

# minimum confidence score that a bounding box must have in order to
be kept.

model.conf = 0.20  # NMS confidence threshold

# perform inference

results = model(img, size=640)

# Results

results.print()

results.xyxy[0]  # img1 predictions (tensor)

results.pandas().xyxy[0]  # img1 predictions (pandas

#parse results

predictions = results.pred[0]

boxes = predictions[:, :4] # x1, y1, x2, y2

scores = predictions[:, 4]

categories = predictions[:, 5]

# show detection bounding boxes on image

results.show()
```

Cashew Plant Model - Training

```
!yolov5 train --data data.yaml --img 640 --batch 16 --weights
KaraAgroAI/CADI-AI --epochs 10
```