Analyzing and Comparing Results for Multiple Machine Learning Models

Jefferson Ng
Department of Mechanical Engineering, University of California, Irvine
92697, USA. jefferzn@uci.edu

YIJIAN LI
Courant Institute of Mathematical Sciences, New York University
NY, 10012, USA
yl9596@nyu.edu

Xuxiao Ji
School of Computer Science, University of Waterloo
Waterloo, N2L 3G1, CA
x56ji@uwaterloo.ca

Can Yang
Department of Electrical Engineering and Computer Science, Oregon State University
Corvallis, 97331, USA.
yangca@oregonstate.edu

Jiaqiao Wan
49 Oakleigh Gardens
London, N20 9AB, England
webshell.hello@gmail.com

Xingyu Lu
United World College Changshu China
Jiangsu, 215500, China
xylu22@uwcchina.org

Abstract—**In the contemporary landscape of atmospheric sciences, the ability to recognize specific weather phenomena accurately and efficiently has become paramount, largely due to the exponential growth in sensor-generated data. This paper undertakes an in-depth analysis of advanced algorithms, namely YOLOV8, ResNet50, and Convolutional Neural Networks (CNN), for the purpose of weather pattern identification and classification. Building on the foundation of deep learning, the YOLOV8's real-time object detection capabilities are leveraged to discern intricate weather patterns in diverse datasets, from meteorological stations to satellite imagery. On the other hand, Resnet 50 which is another CNN based model was also tested along with CNN for comparison. An exhaustive evaluation of these algorithms covers various metrics, including accuracy, precision, computational efficiency, and real-world applicability. Special emphasis is placed on the crucial stages of data preprocessing, feature extraction, and model tuning, highlighting their impact on the algorithms' overall performance. Our findings suggest that, when appropriately optimized, both YOLOV8 and CNN exhibit exceptional capabilities in discerning and classifying intricate weather patterns, whereas ResNet50 exhibits comparatively less performance.**

Keywords—**YOLO V8, ResNet50, Convolutional Neural Network, weather Image Classification,**

I. INTRODUCTION

In the field of technology, the influence of machine learning emerges as a fundamental pillar, reshaping industries, and domains across the entire spectrum. At the forefront of this transformative wave lies image recognition, a domain with far-reaching implications, extending from the realm of autonomous robotics to the intricate field of medical diagnostics. Central to the very essence of image recognition are Convolutional Neural Networks (CNNs), a deep learning model that showcases their remarkable ability to provide meaningful insights from visual data. This paper delves into the extensive domain of CNN-based detectors, with a specific focus on their application within the context of identifying weather-based images.

The significance of image recognition extends seamlessly into the domain of robotics, where the capacity to adeptly process and comprehend visual data forms the bedrock of meaningful interactions with the environment. Analogous to how human vision empowers us to navigate our world and make informed decisions, the visual acuity of robots serves as a critical factor for tasks ranging from delicate object manipulation to the identification of potential hazards and the awareness of spatial surroundings. Consider the scenario of a self-driving vehicle navigating through a bustling urban landscape; the timely and accurate identification of pedestrians, traffic signals, and surrounding vehicles not only shapes the vehicle's safety but also directly impacts its efficiency. This underscores the pivotal role that image recognition plays in shaping the cognitive fabric of robotic systems.

CNN is well known for tackling the complexities of image recognition challenges. With the unique convolutional layer’s filters, CNNs is capable of extracting key features of within images. Equipped with specialized layers designed to apprehend spatial hierarchies and invariant features, CNNs prove adept at recognizing objects in the face of varying scales, orientations, and positions.

Central to the objective of evaluating the efficacy of CNN-based detectors, this study zeroes in on a specific yet crucial application: the identification of weathers type. The precision in recognizing weather type based on image is important across diverse domains, ranging from urban planning and emergency response to navigation and logistics. As these detectors undergo meticulous scrutiny, analysis, and comparison.

The dataset chosen for the comparison of models in this paper is the Weather phenomenon database (WEAPD) is a publicly available database in Harvard Dataverse. This dataset includes 6,877 images of 11 types of weather images such as lightning, Rime, Fog. The variety in weather type provides great application for real world robotics applications, as certain weather types will impose greater threat than others it is image recognition can play a role, especially in autonomous robot, in identifying such cases and act accordingly.

II. RELATED WORKS

Much research has been done within the realms of image recognition. Most have a consensus that there has been great success within the recognition of pattern within image through the development of CNN. With the Convolution layer breaking down images into meaningful data, many CNN based algorithms such as Resnet, AlextNet, InceptionV3, YoloV5. Researchers hence devote study into manipulating data information. The survey on Image Data Augmentation for Deep Learning by Shorten and Khoshgoftaar focuses on the issue of overfitting. Data sets in image recognition are generally scars due to the nature of image yet the study found that data augmentation categorized by data warping or oversampling technique can lead to a data with small sample size still obtaining high levels of generalizability. [4] As time progresses, individuals have increasingly found the conventional CNN algorithm inadequate for comprehensively analyzing the information conveyed through images. The work by titled Tian discusses enhancements to the conventional CN) for better image processing performance. Tian have integrated a Recurrent Neural Network (RNN) with the CNN, allowing both to learn deep features of the image simultaneously. They introduce a new residual module, inspired by the ResNet's skip convolution layer, termed ShortCut3-ResNet. Furthermore, a dual optimization model is presented to optimize both convolution and full connection processes. The paper delves into a simulation study to analyze the effects of various parameters on the network's performance. The experimental results indicate that their proposed CNN algorithm offers improved feature extraction and image recognition capabilities. [8] The study by He, Zhang, Ren, and Sun uses the concept of residual learning frameworks, has facilitated the training of profoundly deeper neural networks, with empirical evidence showcasing enhanced optimization and accuracy; notably, these deep residual networks, with layers up to 152, outperformed previous models, achieving first-place results in several major ILSVRC & COCO 2015 competitions. [1] Another research on image processing-based vehicle license plate recognition systems published in 2018, Yogheedha, Nasir, Jaafar, and Mamduh combined two methods of digital image processing and OCR (Optical Character Recognition) for image processing as well as template matching and achieved satisfactory results. With a large amount of data acquisition as a base, the researchers successfully completed optical character recognition and successfully identified 13/14 vehicles.[2]

In the period spanning the last ten years, some papers have been done on weather recognition. There are different aspects of weather that can be told by the pattern of the image. Clouds for example are a good indication of the upcoming weather. The study by Cao and Yang developed an app predicting the weather through image recognition of the clouds. In this study, they only classified four types of clouds that bring rain, fair weather, tornados, and fog clouds while focusing on classifying the shape using CNN. From their result, they found that weather with distinct features such as fog require less data to obtain higher accuracy. Rainy and tornado clouds predictions require more data input while sunny and rainy clouds are hard to distinguish. The paper, however, did not include comparison with multiple machine learning models. [7] The research, conducted by Li Wei, Ke Lin Chou, and Ru Hong Fu, introduced a novel deep learning framework tailored for weather image recognition, effectively classifying images into hazy, rainy, snowy, or other categories. Leveraging renowned deep CNN models like GoogLeNet, the results demonstrated the superiority of the deep learning approach over conventional feature-based methods. This innovative method can seamlessly function as a preliminary step in devices equipped with standard deweathering capabilities, making it ideal for integration into vision-assisted transportation systems, ADAS, and various outdoor surveillance systems. [3] Vinay Kukreja, Vikas Solanki, Anupam Baliyan, and Vishal Jain have presented a transfer learning (TL) based multi-classification model tailored for weather recognition using a dataset comprising 1,000 images sourced from various Indian locales. This research utilized the MobileNet V2 pre-trained model in tandem with weather image classifiers, and through the TensorFlow and NumPy python libraries, achieved an impressive accuracy of 98.25% for rainy (R) weather class images. Their findings indicate a notable advantage of the TL approach over traditional CNN methods in the realm of multi-classification tasks. Looking ahead, the team aims to further enhance model performance by experimenting with larger datasets, hybrid models, and the incorporation of additional weather classes. Furthermore, through the iterative refinement of CNN algorithms, researchers and practitioners have developed an enhanced variant of the CNN algorithm that has found practical application in real-world scenarios. In “ResNet15: Weather Recognition on Traffic Road with Deep Convolutional Neural Network”, it delves into the pivotal role of automated weather recognition in shaping urban traffic dynamics, emphasizing its applications in traffic alerts, driving assistance, and intelligent transport systems. Leveraging deep CNN, the study introduces a streamlined model, "ResNet15," derived from ResNet50, tailored to extract weather features through specialized convolutional layers and residual modules. Employing a comprehensive dataset, “WeatherDataset-4,” comprising 4983 weather images across four categories, the study demonstrates ResNet15's efficacy in recognizing weather patterns. Comparative experiments highlight ResNet15's superiority over conventional ResNet50 in accuracy, processing speed, and model size, underscoring its potential to advance weather recognition for traffic-related purposes. [6] More recently, the paper, Classification of Weather Phenomenon from Images by Using Deep convolutional Neural Network analyzed, Tested CNN on WEAPD(dataset used in this paper). They achieved an accuracy classification of 92% using a modified version of CNN they named MeteCNN. This paper provided analysis of MateCNN based on comparing typical metrics values including F1 score, precision and Recall. The paper provided a database that satisfies meteorological criterion and will be tested in this paper.[5]

III. METHODOLOGY

A. *experimental setup and evaluation*

The dataset used for all three models are the WEAPD. In the paper by Xiao, they evaluated the final result of the model after it has been optimized by comparing the metrics values. In our paper we decided to explore the graphical training results of the accuracy vs epoch for YoloV8 and ResNet50. The evaluation and comparison will be a analytic based on qualitative observations of graphs behaviors. The key features that will be explored through the graphs are quality of data, training duration, and overfitting behaviour. The quality of data is shown by the signs with any vibration signs in the process of training, as poor quality could result in irregular training pattern. Optimal training duration or number of epochs is shown by the time it takes for the accuracy to plateau, this suggests the compatibility of the model with the weather-based dataset. Overfitting is shown through the difference between training and testing accuracies. Overfitting could indicate whether the size of 6000 weather images is sufficient for developing models. Since CNN is a commonly known and a modified version of it is shown in Xiao's paper, we have tested its accuracy the batch size, another hyperparameter that was not tested which suggest information about how the dataset should be handled.

For all three models we used a training/validation/testing ratio of 7 to 2 to 1. This was done by randomly splitting ratios with respect to each individual categories to three groups. It is worth noting that the numbers of images in this dataset for each category are uneven for example the least number of images is from rainbow with only 232 while rime have 1160 images. This is a common ratio that is widely used and balanced standard ratio that most machine learning model use. Hyperparameters are tunned differently based on different models. In CNN only batch size is tunned as CNN is used as a comparison and has already been tested with optimised parameters in previous paper to achieved accuracy of 92%.[5]

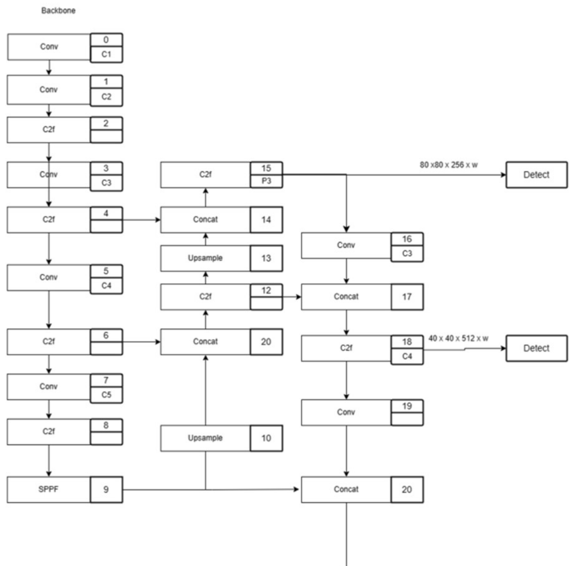

Figure 1: YOLO V8 architecture

Figure 1 shown above indicates the architecture of yolo v8. The arrow indicates the flow of image data processing starting from the top left. Information about the architecture can be found in the yaml file included in yolo v8. The backbone is how the model extracts features out of an input image. The model has a backbone with 5 major layers indicated by letter C followed by a number (This standard follows Feature Pyramid Networks for Object Detection). The number in boxes indicate the block or layer number. There are 21 layers in total. The layers of yolo v8 are made up of 6 building blocks: Conv, C2F, Upsample, Concat, SPPF, and Detect.

Conv is a convolutional layer that is applied to extract features from input information. Within the convolutional layer are filters(kernels) that scan the feature map side by side. Through matrix multiplication it extracts certain features information. The convolutional layer can vary in depth by changing the number of kernels that will help extract different features. Padding is also an important spatial feature that can be applied in the convolutional layer that changes the dimension of the input data to a different value. C2F is a special operation layer introduced in yolo v8 with previous earlier yolo versions using c3 module. The c2f is a new module that uses all the output from 2 3 by 3 convolutional layers. This new module reduces parameter count and size of tensors. This module combines high-level features with contextual information to increase accuracy. Upsample is a module that increases the spatial resolution of feature maps. Low resolution features are typically produced by prior layers and hence it is important to apply upsampling. Concat layer is a layer that is used to combine features maps of different spatial resolutions from earlier stages of the network. This is typically done after upsampling the lower resolution layers. SPPF module is a module designed to speed up the computation of a network by pooling different scales into specific feature maps. There are three detect layers which are standard in many image recognition. This is where the model makes detections. The three detect modules come from layer 3, 4, and 5. This helps increase multi scale object detection to detect items with different scales. For example, one object might look smaller in one image compared to another due to how the image is captured. Another implication of this feature is hierarchical feature extraction as the first three layers produce low- level features while C4 and C5 breaks down the image even more to recognize higher-level features.

For Yolo V8 the hyper parameter tuned is the learning rate, model size, and batch input. YoloV8 have 5 sizes that can be tunned with Nano, Small, Medium, Large, 8X depending on the size of the data set. YoloV8M(small) is used for this dataset considering the dataset has 6000 images. Having a nonmatching size will result in overfitting. Learning rate is set to 0.001 after experimentation with different learning rates it was found to produce the smoothest convergence. Batch input was set to 16. Unlike most supervised models, YoloV8 is a relatively complete model and doesn't require much tunning.

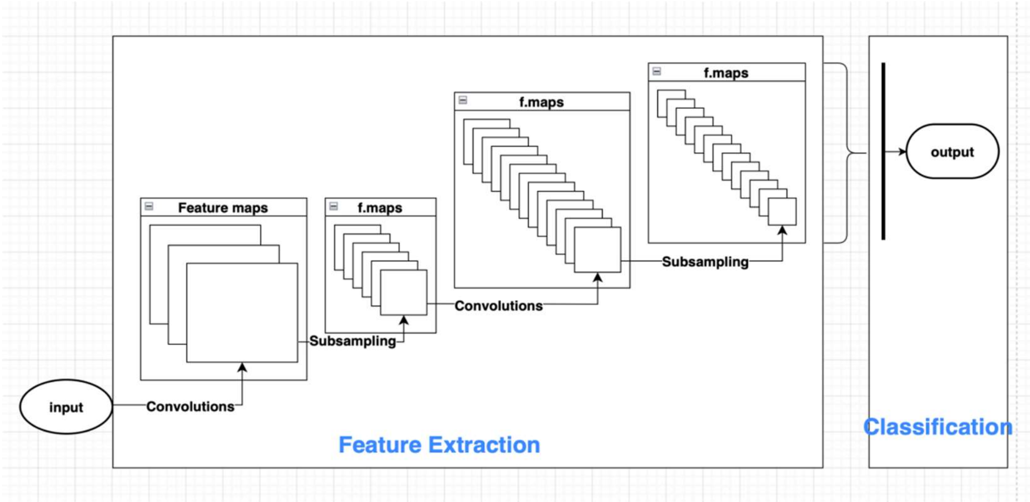

Figure 2: CNN

Convolutional neural networks have shown outstanding performance in tasks such as image recognition, image classification, and object detection. The CNN developed for image analysis is described below.

Preprocess the images, such as normalizing pixel values, data augmentation (such as random cropping, rotation, flipping, etc.), to respond to the data's diversity. Our goal is to accurately identify the various shapes of X and O using CNN, which involves how to effectively extract features as critical factors for recognition. When given a new image, the CNN cannot accurately know which part of the original image these features should match, so it tries every possible position in the original image, making this feature a filter. This matching process is called the convolution operation.

Here, we manually build the network structure rather than using existing framework models. To reduce the computational load effectively, we use pooling to minimize pixel information by shrinking the input image, leaving only important information. We also set an activation function (ReLU) to add non-linear factors to map the convolution layer output results non-linearly.

In each training iteration during the training dataset, batch size are input to the model, and the loss is calculated, and the model weights are updated through backpropagation. In the training model's code class, we integrate the compiled model's loss function and optimizer. Training loss, accuracy, rounds, forward propagation, and backpropagation are calculated, and training result charts are finally drawn. During training, we monitor changes in indicators to understand the model's training progress, overfitting situation, and generalization ability.

Use the validation dataset to monitor the model's performance. Adjust hyper-parameters such as learning rate and regularization parameters based on validation results. Modify the number of samples input to the network at one time during training. The training sample size used in each parameter update (batch size) is specified as 4, 8, 16, 32, and 64, respectively. Finally, evaluate the model's performance using the test dataset, computing training loss, training accuracy, and other metrics.

As mentioned in 3.1 the hyper parameter we are investigating in CNN is just batch size because an optimized CNN model has already been shown to produce a result of 92% accuracy.[5] Therefore the purpose of CNN with just convolutional layers formation a basic neural network is used for comparison and produce valuable information regarding batch size.

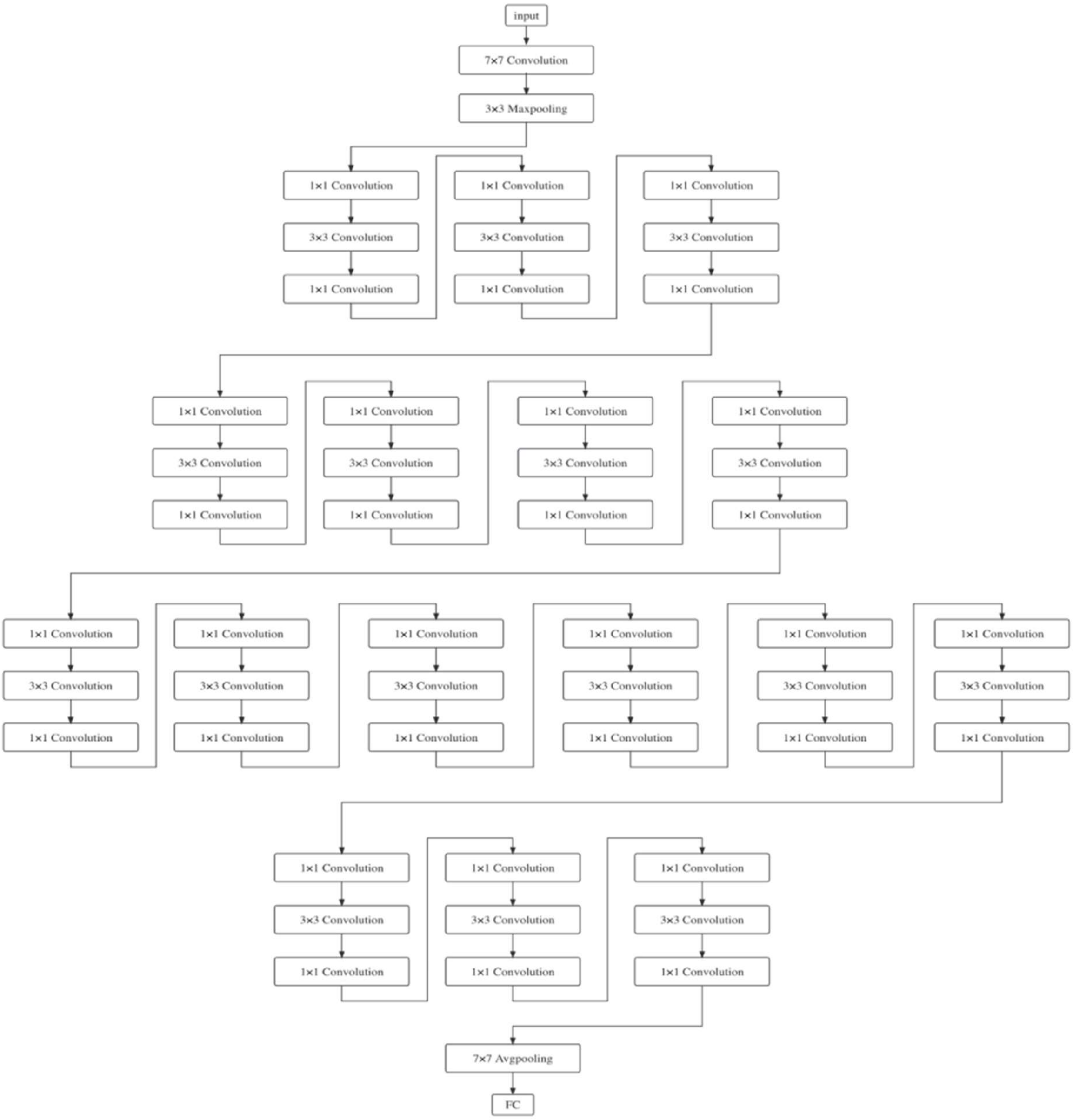

Figure 3: ResNet50 Architecture

ResNet-50 is a convolutional neural network with 50 layers designed for a range of computer vision tasks. It's a part of the Residual Network (ResNet) family introduced by Kaiming He et al. in 2015. The standout feature is its "residual connections," which solve the vanishing and exploding gradient issues that plague deep networks. These shortcuts allow the gradient to flow more easily, enabling the network to learn effectively.

The network starts with an input image of shape (224, 224, 3). The initial convolution layer uses 64 filters of size (7, 7) with a stride of 2, followed by a max pooling layer with pool size (3, 3) and stride 2. This is succeeded by batch normalization and a ReLU activation function.

The main architecture consists of four stages, each containing a series of residual blocks. The residual blocks contain three convolutional layers in a "bottleneck" design: a 1x1 convolution for dimensionality reduction, a 3x3 convolution for feature extraction, and another 1x1 convolution to restore the dimensions. Each convolution is followed by batch normalization and a ReLU activation.

The stages differ in the number of blocks and filters used:

- Stage 1: 3 blocks with 64 filters
- Stage 2: 4 blocks with 128 filters
- Stage 3: 6 blocks with 256 filters
- Stage 4: 3 blocks with 512 filters

After the last stage, global average pooling is performed to compress the spatial dimensions, retaining only the channel information. Finally, a fully connected layer generates the output, which could be class probabilities in a classification task.

To summarize, ResNet-50 is celebrated for its depth and efficiency. Its residual connections circumvent the difficulties of training deep networks, allowing it to learn complex features effectively. It's widely used in a variety of applications from image classification to object detection.

Similar to YoloV8 ResNet-50 is a complete deep neural network with premade architecture that doesn't require much optiomization, hence we only tested the batch size and found 20 to be an appropriate input.

IV. RESULTS

A. Yolo V8 Model

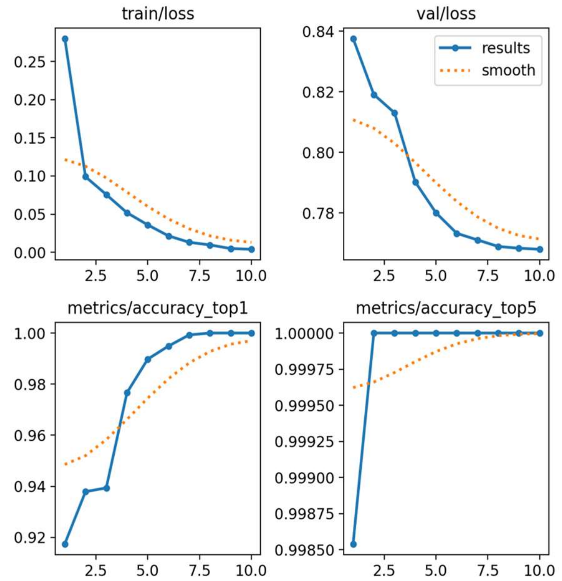

Figure 4: Yolo v8 training accuracy

In Fig 2., the four graphs of loss function and accuracy of YOLO v8 model are shown. The y axes on the two graphs at the top represent the value of loss, while the y axes at the bottom represent accuracy. All of the x axes represent the number of epochs. The range of epochs is between 0 and 10 which suggests that there are 10 sets of data being trained in the model. According to the figures of loss functions, the loss values decline sharply initially and then the rate of declining decrease as the epochs increase. In the case of train loss, the range of loss value is between 0.02 and 0.28 approximately, while the range of validation loss value is between 0.76 and 0.84. The train loss value seems to converge to zero, while the value of validation loss seems to converge to 0.76. This shows that the value of train loss is significantly lower than validation loss which indicate significant overfitting behavior. As for the figure of accuracy, the figure on the left shows relation between the top 1 accuracy (the probability of gaining correct result from one prediction) and number epochs, while right one shows relation between the top 5 accuracy (the probability of gaining the correct result from the top 5 highest possible predictions) and epochs. The range for top 1 accuracy is between around 0.92 and 1, while the range for top 5 accuracy is between 0.9985 and 1. Both of the accuracy values are increasing and converge to 1, which means 100% accuracy, in the end. Besides, it only took 1 round of training to reach 100% top 5 accuracy. These results have strongly proved the high accuracy and performance of our model. However, as indicated in the significant loss function reduction the accuracy of the testing data set is only around 95%. This is shown to be a result of the over fitting behavior.

B. CNN Model

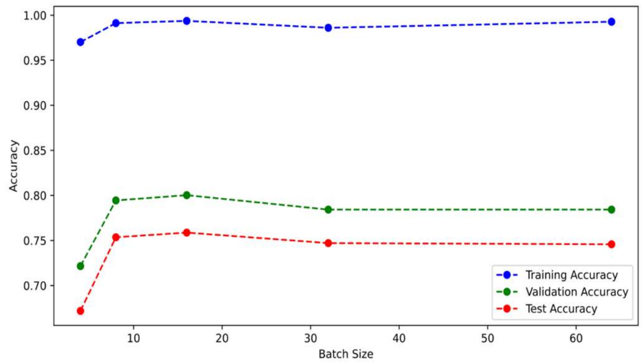

Figure 5: Training, Validation, testing accuracy between difference batch size: CNN

In Fig. 4 the x, y axes represent the size of the batch data and its accuracy, respectively. Different batch sizes will lead to different accuracy of image discrimination. In general, a larger batch size will produce higher accuracy and thus enhance the referability of the experimental results. As shown in Figure 4, when the batch size is 4, the corresponding accuracy is not high, and the machine learning is not perfect at this time, but after learning a large number of mapping relationships between the input and the output, the training accuracy can reach close to 100%. The overall trend of the image is that as the batch size are only impactful .prior to 9 batch and it decrease significantly after that.

To enhance Accuracy on Convolutional Neural Network, we have three main approaches: data augmentation, regularization, and hyperparameter tuning. We applied regularization in this experiment to prevent overfitting. By randomly dropout some neurons during training to reduce the training loss.

The training dataset(blue line) is used to teach the machine or fit a sample set that is being learned by the machine. The developer compares the output produced after the initial piece of training data is introduced with the intended response. An error or loss function that measures the model's performance (the discrepancy between anticipated and actual values) is based on this comparison. The model's parameters, weights, and functions are then adjusted. Different optimization methods are used for this adjustment, with gradient descent being the most popular. While the validation dataset(green) is used to validate and evaluate the model that has been learned by the machine and can be tweaked to optimize it at this stage. Cross-validation (CV) techniques are applied at this stage and ensure stability by estimating how the predictive model will be executed. Finally, the testing dataset (red line) is used to compare the deviation of the machine-learned model with the existing results to make a final evaluation. All of these three datasets are very important and are a necessary step in creating a successful ML model.

We will then discuss the specific meanings in some of the images and make an analysis. We use 2D convolution to improve accuracy and to access the image and see what can be extracted as important features using maximum or average pooling. The horizontal axis (batch size) of all three lines ranges from 0-64 and the first point occurs at a batch size of 4. This is due to the fact that we manually set the parameter to 4, 8, 16, 32, and 64 in order to compare the change in accuracy of the three lines. As can be seen from the images, the training data gradually increases and reaches the highest accuracy at batch sizes of 4, 8, and 16, and then decreases slightly as the batch size continues to increase. This is because different batch sizes affect the variance of the gradient estimator. Larger batch sizes can also lead to lower variance and hence lower learning rate.

C. ResNet Model

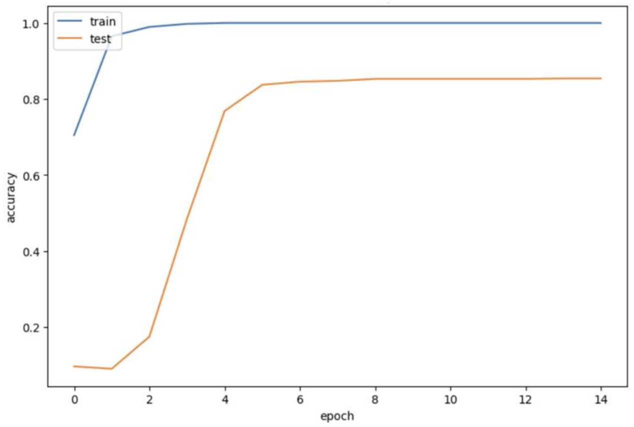

Figure 6: Accuracy of training and testing datasets: ResNet50

In Figure 6, the coordinate axis is usually structured with epochs along the x-axis and accuracy percentages on the y-axis, ranging from 0 to 100%. The color-coded legend identifies the blue line as representing the training dataset and the orange line as indicative of the testing dataset. Starting from the leftmost part of the plot, the blue line for the training set generally begins at a lower accuracy level and exhibits a rising trend as it progresses toward the right, eventually levelling off or plateauing. This plateau usually signals that the model's ability to learn from the training dataset has saturated, and it is optimizing its performance based on the given data. On the other hand, the orange line representing the testing dataset often shows a different trend. It may start with a rapid increase in accuracy, reflecting the model's initially good performance on unseen data. After this initial surge, it often stabilizes but might exhibit minor fluctuations. Since there is no intersection point between the two lines, we can interpret that the model's performance is consistently different across the training and testing sets throughout the training process. This sustained separation might suggest that the model is well-suited to the training data. Both lines trend upward and remain separate without intersecting, which could indicate that the model is generalizing well without overfitting. No inflection points would signify a relatively smooth learning process, free of significant changes in learning speed or effectiveness.

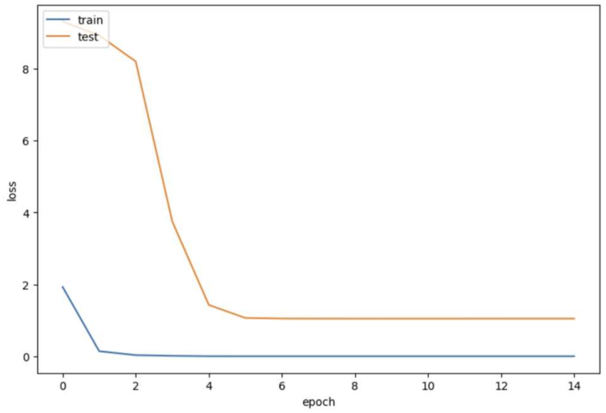

Figure 7: Loss of training and testing datasets: ResNet50

In Figure 7, the coordinate axis typically has epochs on the x-axis and loss values on the y-axis, which generally range from a high value at the start down to zero or near-zero values. According to the legend, the blue line signifies the training dataset, and the orange line signifies the testing dataset. The blue line often begins at a relatively high loss value and shows a strong downward trend as epochs increase, slowing down its rate of descent as it approaches the minimum loss value. This slowdown typically means that the model is reaching an optimal state with regard to the training data. The orange line for the testing dataset, meanwhile, might start high and decline in a similar fashion, but its trend can vary towards the end of the epochs. The absence of an intersection point implies a consistent difference in loss values between training and testing datasets throughout the training cycle. This could indicate overfitting if the testing loss is significantly higher and shows an upward trend while the training loss continues to decrease. Since there are no inflection points in either line, this would suggest a stable learning process without abrupt changes in the rate of loss reduction. This is a positive sign, indicating that the model is steadily learning from the training data and maintaining a consistent performance level on the testing data, provided that the loss values are reasonably low for both datasets.

D. Summary

Table 1: Summary of performance metrics for tested models

Performance Metrics	YoloV8	CNN	ResNet50
Training Accuracy	Near 100%	Near 100%	Near 100%
Testing Accuracy	95%	75%	85.4%
Overfitting Analysis	Significant overfitting	Significant overfitting	Over fitting
Training curve	Smooth	Smooth	Smooth
Learning Rate	Fast learning rate (7.5 epochs)	N/A	Fast learning rate (5 epochs)

Best batch input	16	16	20
Model Complexity	High	Low	High

As indicated in table 1 above, YoloV8, ResNet50, and CNN we obtained a testing accuracy of 95, 85 and 76 percent respectively. This suggests that YoloV8 are best suited for the task of identifying weather-based image. The training curves from all three graphs show smooth looking curves during testing accuracy hence it is a good indication that quality of images in the dataset is high. The testing accuracy for YoloV8 and ResNet50 plateaus around 7.5 and 5 iterations respectively which Suggest a fast-learning rate for both models. The most important indication is that there are over-fitting signs in all three graphs indicating that the dataset is too small, or the models are too complex. As expected, highly complex models which are YoloV8 and ResNet50 out performed the basic CNN model.

V. CONCLUSION

In this paper, an investigation was conducted on Yolo V8, ResNet 50, and CNN using a weather-based image recognition dataset. Weather-based images lack specific features, such as numbers, for the model to track, making it a more challenging task compared to regular image classification. However, image recognition for weather conditions holds practical potential applications, especially for robotics, such as drones that need to adapt their movement patterns based on varying weather conditions.

Yolo V8, the latest addition to the Yolo series, was selected to showcase the performance of one of the most capable models in the field of image recognition. CNN, which serves as the foundational block of Yolo, was investigated for the purpose of comparison. Additionally, ResNet 50, commonly used as a backbone, was chosen for comparison with the other two models.

Yolo V8 achieved an accuracy of 95% after 7.5 epochs, with a significant decrease in loss observed after 2.5 epochs. For CNN model, an accuracy of 76% after running 8 batch sizes. Meanwhile, ResNet 50, known for its deep architectures and residual connections, demonstrated its reliability and robustness, achieving an accuracy of 85.4% after 15 epochs, further enriching the comparative landscape of our study.

VI. ACKNOWLEDGEMENTS

Jefferson Ng, Can Yang, Jiaqiao Wan, and Xuxiao Ji contributed equally to this work and should be considered co-first authors.

VII. REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

[2] K. Yogheedha, A. S. A. Nasir, H. Jaafar and S. M. Mamduh, "Automatic Vehicle License Plate Recognition System Based on Image Processing and Template Matching Approach," 2018 International Conference on Computational Approach in Smart Systems Design and Applications (ICASSDA), Kuching, Malaysia, 2018, pp. 1-8, doi: 10.1109/ICASSDA.2018.8477639.

[3] L. -W. Kang, K. -L. Chou and R. -H. Fu, "Deep Learning-Based Weather Image Recognition," 2018 International Symposium on Computer, Consumer and Control (IS3C), Taichung, Taiwan, 2018, pp. 384-387, doi: 10.1109/IS3C.2018.00103.

[4] Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

[5] Xiao, H., Zhang, F., Shen, Z., Wu, K., & Zhang, J. (2021). Classification of weather phenomenon from images by using deep convolutional neural network. *Earth and Space Science*, 8, e2020EA001604. https://doi.org/10.1029/2020EA001604

[6] V. Kukreja, V. Solanki, A. Baliyan and V. Jain, "WeatherNet: Transfer Learning-based Weather Recognition Model," 2022 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2022, pp. 1-5, doi: 10.1109/ESCI53509.2022.9758183.

[7] Y. Cao and H. Yang, "Weather Prediction using Cloud's Images," 2022 International Conference on Big Data, Information and Computer Network (BDICN), Sanya, China, 2022, pp. 820-823, doi: 10.1109/BDICN55575.2022.00157.

[8] Y. Tian, "Artificial Intelligence Image Recognition Method Based on Convolutional Neural Network Algorithm," in IEEE Access, vol. 8, pp. 125731-125744, 2020, doi: 10.1109/ACCESS.2020.3006097.