PJEST

PJEST

Article

Vegetation and Non-Vegetation Classification Using Object Detection Techniques and Deep Learning from Low/Mixed Resolution Satellite Images

Faisal Ahmed [1], Waheed Noor [1], Mohammad Atif Nasim[2], Ihsan Ullah [1], Abdul Basit[1]

[1] Department of Computer Science & Information Technology, University of Balochistan; faisalahmedmengal@gmail.com, waheed.noor@um.uob.edu.pk, ihsanullah@um.uob.edu.pk, drabasit@um.uob.edu.pk

[2] Food & Agriculture Organization of the United Nations; mohammad.atif@fao.org

* Correspondence: Waheed Noor waheed.noor@um.uob.edu.pk;

ABSTRACT: Vegetation cover classification using mixed or low-resolution scalar images is challenging. Fortunately, recently deep learning object detection methods have emerged as a replacement to the conventional machine learning methods for the detection and classification of land use and land cover. This paper presents a deep learning object detection approach for land use and land cover detection using low/mixed resolution satellite images acquired from Google Earth satellite images. Google Earth images are accessible freely using the Google Earth Pro desktop application. Our dataset consists of two (02) classes (vegetation and non-vegetation) with a total of 450 labeled images captured from different parts of Pakistan. We present a comparison of the recent anchor-free object detection model YOLOX with the anchor-based object detection model YOLOR for solving real-time problems. The end-to-end differentiability, efficient GPU utilization, and absence of hand-crafted parameters make anchor-free models a compelling choice in object detection, and yet not been explored on Land cover classification using satellite images. Our experimental study shows that YOLOX delivers an overall accuracy of 83.50% on Vegetation and 86% on Non-Vegetation classes, which outperformed YOLOR by 30% on Vegetation classes and 34% on non-Vegetation classes for our dataset. We also show how an object detection system can be used for Vegetation and Non-Vegetation classification tasks, which can then be used for change monitoring and assisting in developing geographical maps using low/mixed resolution freely available satellite images

Keywords: *Deep learning; Earth Observation; Land Use Detection, Land Cover Detection, Object Detection, Remote Sensing (RS).*

Citation: Noor, W., Faisal Ahmed, Mohammad Atif Naseem, Ihsanullah, & Abdul Basit. Vegetation and Non-Vegetation Classification using Object Detection Techniques and Deep Learning from Low/Mixed Resolution Satellite Images : Vegetation and Non-Vegetation Detection Techniques and Deep Learning. *Pakistan Journal of Emerging Science and Technologies (PJEST), 4(4).* https://doi.org/10.58619/pjest.v4i4.152

Academic Editor: Firstname Lastname

Received date: 8-Oct-2023
Revised date: 26-Dec-2023
Accepted date: 27-Dec-2023
Published date: 30-Dec-2023

Introduction:

Remote sensing is being used widely to make measurements of the earth using sensors on aircrafts, satellite images, and nowadays drones. Often, the collected data from these sensors are in the form of images, which can be manipulated, analyzed, visualized, and applied to agriculture, disaster recovery, forestry, regional planning and for other applications like analyzing natural resources and development activities. Significant efforts have been taken by Governmental programs such as ESA's Copernicus and NASA's Landsat to make data available freely with the intention to boost entrepreneurship and innovation for commercial and non-commercial purposes. However, satellite data needs pre-processing and transformation into structured semantics before it can be fully utilized [1]. One of those fundamental semantics types are

land use and land cover classification [2], [3]. The land use classification deals with the automatic labeling of a particular land area according to its usage, e.g., recreational, residential, settlement, industrial, etc., while the land cover classification is aimed at automatically classifying a particular land area according to the land surface cover, e.g., vegetation, water bodies, rangeland, etc.

Fig. 1: Land use and land cover object detection based on Google Earth images. Images are labeled to identify the vegetation and non-vegetation classes.

As compared to general classification tasks, remote sensing classification is considered to be more challenging, and, the progress of classification in the remote sensing area is slow due to the unavailability of reliably labeled datasets. The performance of classification systems in supervised machine learning depends strongly on the availability of quality dataset that is properly labeled with pre-defined set of classes [4].

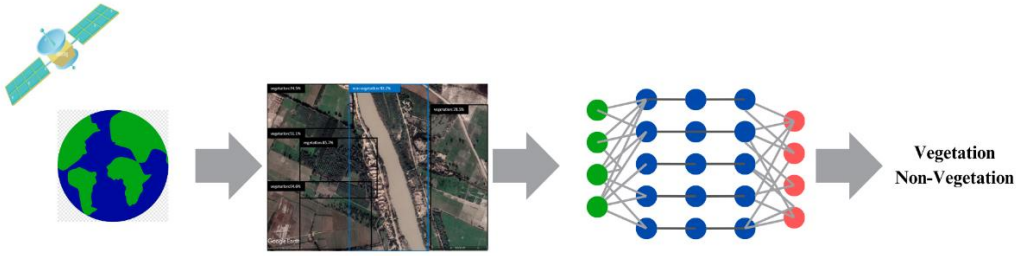

Fig. 2: Object detection and classification process using satellite images for land cover classification.

Penatti et al [5] used low-level color descriptors to evaluate deep CNNs on the UCM and BCS datasets. while using the same data sets, provided an intensive comparison of various machine learning methods including deep CNNs, Bag-of-Visual-Words (BoVW) and

spatial pyramid match kernels for land cover classification. Table 1 provides details of available public datasets for the purpose of land cover and land use classification.

Most of the models, to the best of our knowledge, are based on single class per image data sets and the availability of high-resolution satellite images. However, to utilize remotely sensed satellite images for land use and regional planning, the transformation of desired images into one class per image in a real implementation environment poses challenges to such models.

Machine learning (ML) approaches traditionally extract features such as corners, edges, and color histogram from every image. Then the extracted features are given to the learning algorithm as an input, which then performs object classification, as shown in [6] and [7]. Still, the widely used ML techniques are Haar-feature based classification [8] and support-vector-machines (SVM) together with histograms of oriented gradient features [9]. On the other side, classification and localization are performed jointly in deep learning (DL) methods [10] [11] .

In order to determine which approach is better among the ML and DL for a particular problem, hardware resources, and the training data are critically important to be considered. In general, DL approaches perform well subject to the availability of large training data with high computational power to train the model in an acceptable time.

Remote Sensing (RS) retrieves images from resources such as satellites, aircraft, and nowadays from drone sensor technologies, which makes it harder to collect data for training of classification and object detection tasks. Some typical sensor technologies like light detection, ranging, Red, Green, and Blue (RGB) cameras, and infrared (IR) are used for data collection in RS. Such remotely captured images poses some challenges for classification, 1) scale: objects with respect to image size can be very small which offers little information about features, 2) orientation: images of the same objects are present with different rotation angles. Therefore, algorithms need to extract features from images that are rotation-invariant [12]. Furthermore, the object detection algorithms are required to handle occlusions, compression artifacts, and noisy signals effectively during model building.

In this work, we aim to explore and use the YOLOX, an anchor-free object detection model and compare its performance with the YOLOR, an anchor-based object detection model on land cover classification using low-resolution satellite images. Anchor-free models in object detection offer several compelling advantages. One key strength is their end-to-end differentiability, which allows the network to be trained seamlessly without the need for additional processing steps like Non-Maximum Suppression (NMS). This not only simplifies the training process but also enhances computational efficiency, enabling the entire network to run exclusively on GPUs without relying on CPU operations. Another notable advantage is the absence of hand-crafted hyperparameters and design choices. Unlike anchor-based models, anchor-free models do not require manually defined scales and ratios for anchor boxes. Furthermore, anchor-free models exhibit versatility by leveraging contextual information more effectively. Unlike anchor-based approaches that crop boxes from images, anchor-free models consider the entire image

during the prediction process. This holistic approach to context can result in improved accuracy, especially in scenarios where a comprehensive understanding of the surroundings is crucial. We also aim to explore the performance of the anchor-free model and compare it with the anchor-based object detection model on satellite images which has not been explored so far on the land cover datasets.

Generally, a satellite image contains multiple classes rather than cropped objects that have one class per image. land cover images have been collected for our own dataset from different places in Pakistan. Our aim is to detect and highlight different land covers as vegetation and non-vegetation in a single image using a rectangular bounding box. The significance of this research is particularly pronounced at the local level, given that agriculture and livestock serve as the primary drivers of our economy. The potential implementation of this work holds substantial benefits for these sectors, especially in Balochistan, where 70% of the area comprises rangeland. It's noteworthy that, in our current study, we are confined to addressing only two classes. However, this limitation serves as the initial groundwork for a more extensive multi-class problem, which we have identified as a direction for future research. Expanding our study to include a broader range of classes will enable us to provide a more comprehensive solution that aligns with the diverse needs of our local agricultural and livestock contexts. In Table 1 we present available benchmark datasets for similar tasks but all these datasets provide already cropped images for training and testing of models, which normally is not the case when it comes to the implementation of these models. We have compared two models YOLOX [13] and YOLOR [14] and given a valid and fair comparison among them. We applied transfer learning for training samples using the pre-trained weights. Both models are designed specifically for real-time detection.

Table 1: Different dataset specification

Name & Reference	Resolution	No. of Images	Images per class	Classes
UCM [3]	256 x 256	2100	100	21
AID [15]	600 x 600	10000	200 - 400	30
SAT-4 [16]	28 x 28	500000	100000	4
SAT-6 [16]	28 x 28	405000	54000	6
BCS [5]	64 x 64	2876	1423	2
EuroSAT [17]	64 x 64	27000	2000 - 3000	10

II. DIFFERENT TYPES OF ALGORITHMS

Two-stage, single-shot, and anchor-free DL architectures are best suited to this task. Generally, a two-stage architecture generates a huge amount of candidate regions (e.g., selective search algorithm [18]) and then classification is performed for each region. Region-based convolutional neural network (R-CNN) [19] is primarily considered to be

one of the first successful methods in two-stage architecture. R-CNN extracts class features using CNN that get candidate regions (bounding boxes) as input, which are then forwarded to the SVM for classification. Arguably, the R-CNN computational efficiency is limited since the extraction of candidate regions heavily depends on the search algorithm. This problem is being addressed by Fast-R-CNN, which extracts features directly and uses a softmax layer that extends the predictions of CNN. Faster-R-CNN [20] is further improved by employing a region proposal network using CNN in place of a search algorithm. Faster-R-CNN exhibits efficiency in the inference speed which is often needed in real-time applications. However, they are still computationally demanding but exhibit high detection performance.

In contrast, detection and classification steps are combined in single-shot approaches to jointly predict the bounding box and the class of the object. You only look once (YOLO) [21], YOLOv2 [22], YOLOv3 [23], and single-shot multi-box detector (SSD) [24] with the capabilities of real-time processing are successful single-shot architectures. In this way, YOLO increases processing time significantly by training a single CNN that is capable to predict bounding box and object class labels jointly. YOLOv2 is tuned prior on the bounding box and relies on fully convolution layers instead of predicting widths and heights. While, YOLOv3 is capable of multi-scale (three) prediction making it more suitable to predict multiple spatial resolution objects. YOLOv4 [25] introduced data augmentation that improved the detection and computational performance of YOLOv3 with improvement in loss metrics of IoU, and evolved activation functions. Similarly, SSD that use multiscale feature maps for independent detection are not much different than YOLO architectures. YOLOv5 [26] in the series of YOLO versions is attributed as having moderate-to-high detection performance but with higher efficiency in detection of 48.2 AP on COCO at 13.7 ms.

Both approaches, two-stage and single-shot, rely on horizontally aligned bounding boxes that may result in limiting their performance when tight bounding-boxes are under consideration. Object shapes are approximated more tightly in oriented bounding boxes compared to horizontally aligned ones. [27] introduced a text detector that is single-shot and can predict bounding boxes with arbitrary rotations, different aspect ratios, and varying sizes. Similarly, to extract features that are rotation sensitive, [28] used a regression branch by rotating the conventional filters actively. [29] did modifications in YOLOv3 for the prediction of oriented bounding boxes in remote sensing applications.

While looking at the two-stage and single-shot architectures which rely on anchors and introduce a lot of hyperparameters in the model for the localization of objects, the anchor-free architectures do not involve sliding window rather adopt a pixel-based approach similar to the semantic segmentation. The early examples of anchor-free architectures are CornerNet [30] and fully convolutional one-stage object detection [31]. The aim of these architectures is to reduce hyperparameters numbers and also extra training time. Furthermore, object detection can be simplified by anchor-free architectures and lead to enhancements in training speed. While object detection is performed in general, both approaches ML and DL have been proposed for aerial imagery object detection [32] [33].

III. OBJECT DETECTION WITH YOLOR

The YOLOR (stands for "You Only Learn One Representation") is an anchor-based unified network proposed to encode implicit and explicit knowledge together, similar to the human brain, which can learn knowledge from normal and as well as subconsciousness learning. To serve various tasks simultaneously, the unified network can generate a unified representation. With the very small amount of additional cost (the amount of parameters and calculations is less than one ten thousand) the network improves the performance effectively.

Prediction refinement, multi-task, and kernel alignment have been introduced into the implicit knowledge learning process, and their effectiveness has been verified. The use of neural network matrix factorization, or vector has been discussed as a tool to model implicit knowledge, and its effectiveness has been verified at the same time. It has been confirmed that implicit representation learned can correspond to physical characteristics accurately and also be presented in a visual way. And the implicit and explicit knowledge can be integrated if the operators confirm the physical meaning of an objective and it will have an effect of multiplier. The unified network showed great capability of catching the physical meaning of unlike tasks.

$$y = f_\theta(X) + \epsilon + g_\phi \epsilon_{ex}(X), \epsilon_{im}(Z))$$
$$minimize\ \epsilon + g_\phi \epsilon_{ex}(X), \epsilon_{im}(Z))$$

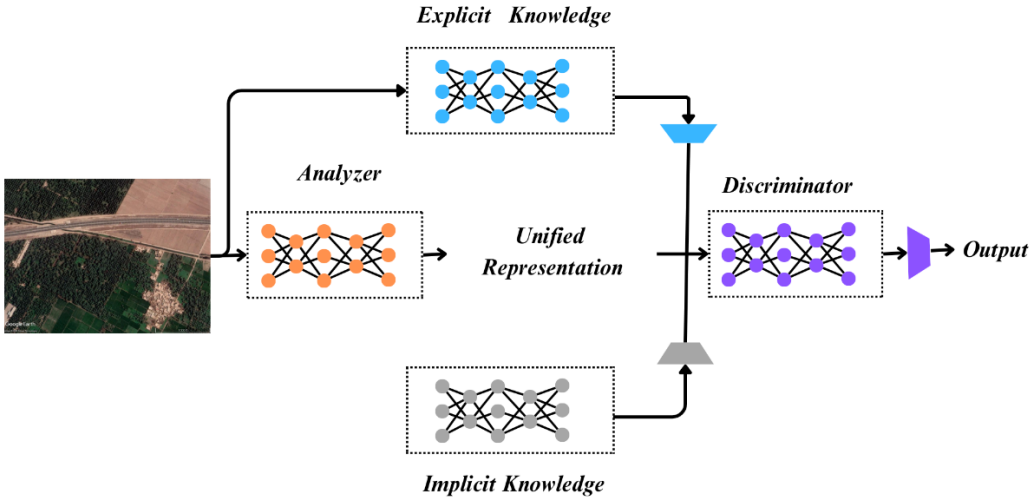

Fig. 3: YOLOR Multipurpose single unified network model [14]

IV. Object detection with YOLOX

As compared to anchor-based YOLO detectors, the YOLOX detector works in an anchor-free manner. YOLOX chose YOLOv3 [23] with DarkNet53 as a baseline. As shown in Figure 4, there have been some changes in training strategies compared to the original implementation [23], such as EMA added weights updating, IoU loss, *lr* schedule, and IoU-aware branch. BCE loss is used for training *cls* and *obj* branch, and for training reg

branch IoU loss has been used. These general tricks for the training are orthogonal to the YOLOX improvement, that's why it has been put to the baseline.

Moreover, for data augmentation, it has only conducted ColorJitter, multi-scale and RandomHorizontalFlip and discarded the RandomResizedCrop strategy because RandomResizedCrop was found overlapped with the planned mosaic augmentation. Furthermore, the YOLO detect head has been replaced with a lite decoupled head, which contains a 1 x 1 Conv layer for reducing channel dimension, followed by two 3 x 3 Conv layers with two parallel branches respectively. To boost YOLOX's performance it has added Mosaic and MixUp into its augmentation strategies.

To switch YOLO to an anchor-free manner, the predictions have been reduced from 3 to 1 for each location and make them predict four values directly, i.e., two offsets in terms of the grid of the left-top corner, and the width and height of the predicted box. While the anchor-free version selects only one positive sample for every object and ignores other high-quality predictions. To tackle this problem, a 3 x 3 center area has been assigned positives. The application of anchor-free models in object detection lies in their end-to-end differentiability, optimized GPU utilization, lack of hand-crafted parameters, and improved utilization of contextual information. These advantages, especially in terms of speed, efficiency, and generalization, establish anchor-free models as valuable and promising alternatives in the field of object detection

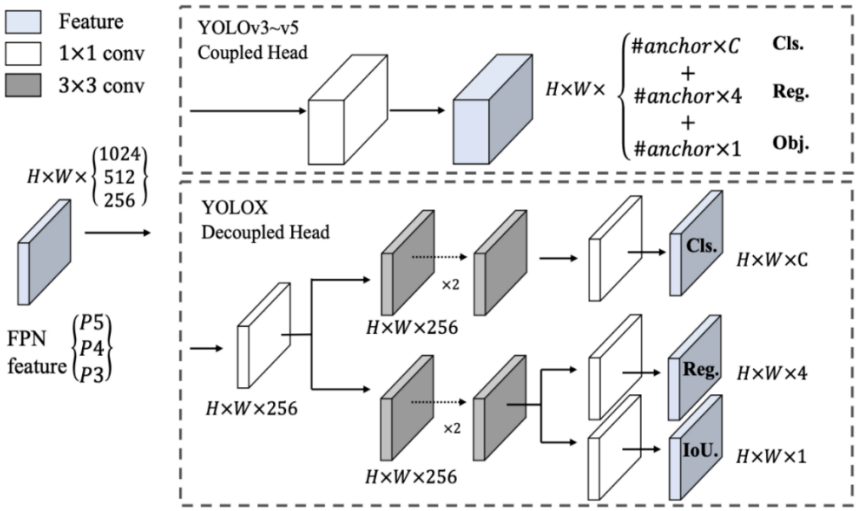

Fig. 4: The difference between the head of YOLOv3 and the proposed decoupled head has been illustrated here [13].

V. DATASET FOR LAND COVER DETECTION

A. Creation of Dataset

To train any deep learning model one of the important criteria is to create a dataset for training and testing as the dataset plays a vital role in training the convolution neural network. To work with land cover detection, land cover images were captured at the

altitude of 1000 feet from Google Earth Pro. These images were taken from different cities in all provinces of Pakistan. A total of 450 images were taken from Google Earth Pro. All captured images were uploaded to Roboflow and each image was annotated one by one. There are a total of two (02) classes named as vegetation and non-vegetation. Each land use or land cover is mapped to vegetation if it is Cropland, Orchard, Forest, Rangeland, etc., similarly River, Highway, Residential, Industry, etc. are mapped to non-vegetation. Images were resized to 640 x 640 pixels for model training and testing. The dataset was split into training, validation, and testing sets, as these splits are frequently used in the field of object detection which allows straightforward division of the dataset and it is suitable for scenarios where you have a substantial amount of data and to assess the model's generalization performance on unseen data. From the total of 450 images, 315 (70 percent) images were selected for training, 90 (20 percent) images were selected for validation, and 45 (10 percent) images for testing. Furthermore, different types of augmentation were applied to the training set to increase the number of training dataset. The dataset was transformed in XML and TXT format such that it can be used easily for model building purposes.

Augmentation of Data

We apply augmentation techniques to our dataset to increase the robustness of our classification models. We used flipping (vertical, horizontal), geometric augmentations, rotation (clockwise and counter-clockwise), photometric augmentations by saturation and brightness techniques. The dataset size got increased to 1366 images after the augmentation and all the satellite images are in Red, Green, and Blue (RGB) color space.

Finally, our dataset consist of 3,724 land cover objects with class distribution as provided in Table 2.

Table 2: Class Distribution of Overall Annotated Objects in our dataset

Classes	Objects (Percentages)
Vegetation	1694 (45%)
Non-Vegetation	2030 (55%)

VII. EXPERIMENTS AND RESULTS

A. Software and Hardware Requirements

To train any machine learning or deep learning models swiftly, we need a good Graphical Processing Unit (GPU) because training machine learning and deep learning models involve mathematical computations, which take longer durations on a conventional Central Processing Unit (CPU) during the training phase. We used Google Colab Pro for our experiments with specifications: NVIDIA Tesla P100 GPU, RAM 24 GB, CUDA version 11.1, and Pytorch version 1.7.1.

B. Training the Models

To train the models we created three random splits of the dataset, namely the training set, validation set, and test set with a 70:20:10 ratio respectively. We also ensured that each split has a representative sample of each class. Therefore, we have 1149 images for training, 110 images for validation, and 53 images for testing. Before we train our models on our datasets, we benchmarked models with pre-trained models such that the two best models can be used for further training.

We trained both of the models YOLOX-S [13] and YOLOR [14] on the training set. We adopt a transfer learning approach, as both of the models have been pre-trained on the COCO dataset. While training some of the hyperparameters were selected on experimental trials whereas the initial values were taken as suggested in the model. Batch sizes were tested between 8 and 16 due to hardware constraints, but it could be increased further in multiples of 2, if the training size is increased.

C. Performance Metric

To evaluate the performance of any object detection model, Mean Average Precision (mAP) serves as a crucial metric for evaluating the performance of object detection models, particularly in contexts involving diverse object classes. Its significance lies in providing a comprehensive measure that considers both precision and recall across all classes, offering a balanced assessment of the model's capabilities. In scenarios where individual class performance may vary, mAP acts as a unifying metric by averaging the precision values for each class, facilitating a holistic understanding of the model's ability to precisely identify objects while ensuring a high recall of relevant instances. Furthermore, mAP addresses challenges posed by class imbalance, offering a single scalar metric that simplifies model comparison and selection by summarizing performance across different classes.

The adoption of mAP is underpinned by its threshold independence, meaning it evaluates performance over a range of confidence thresholds for positive predictions. This characteristic is particularly relevant in object detection, where the acceptance threshold significantly impacts the trade-off between precision and recall. It considers both precision and recall for each class, generating precision-recall curves and calculating the Average Precision (AP) for each class. The mAP is the average of these AP values, providing a

single, interpretable measure of the model's overall effectiveness. A higher mAP indicates better performance, with the metric ranging from 0 to 1, where 1 represents perfect detection across all classes. Additionally, mAP provides insights into the model's robustness at different IoU thresholds through metrics like mAP@0.5 and mAP@0.5:0.95, enhancing the understanding of its bounding box prediction capabilities. In essence, mAP serves as a versatile and comprehensive evaluation metric, addressing various complexities associated with object detection tasks and contributing to informed model assessment.

$$\frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

n= number of classes

AP_K = the average of class K

D. Results

We compared many deep learning-based object detection models such as R-CNN, Faster R-CNN, Single Shot Detector (SSD) and variants of YOLO. We chose the models that were performing best after the training with the highest accuracy with high processing performance. The test was performed on 53 images which were different from the training set and the models were trained on Google Colab Pro. In the case of the YOLOX model, since YOLOX is an Anchor-free model, the learning speed and as well as accuracy turned out to be impressively high. Although the YOLOR model is also faster than other models, the accuracy is so low that it fails to detect some of the objects in an image. The results are shown in Table 3 for both models, and we concluded that the most suitable model for object detection and classification is YOLOX among the other tested models. Figure 5 shows object detection and classification results from both models.

The YOLOX model exhibited outstanding performance on our dataset, surpassing other object detection models and achieving remarkable mAP scores. Specifically, it attained a high mAP of 0.83 for the vegetation classes and an impressive 0.86 for the non-vegetation classes (Table-3).

Table 3: YOLOX performance on Vegetation and Non-vegetation classes

Class	**mAP@0.5**
Vegetation	0.83
Non-Vegetation	0.86

The YOLOR model's mAP score is 0.54 on vegetation classes and 0.53 on Non-vegetation classes, as detailed in Table 4, signifies a moderate level of performance across both

classes. The evaluation indicates that the YOLOR model excels in identifying and classifying objects within both natural and man-made environments. However, there is a notable challenge in detecting objects within land cover classes.

Table 4**:** YOLOR performance on Vegetation and Non-vegetation classes

Class	**mAP@0.5**
Vegetation	0.54
Non-Vegetation	0.53

VII. CONCLUSION

Our experiments suggested that the deep learning approach in object detection tasks delivers an excellent result. We were able to achieve higher accuracy through a deep-learning approach. In this work, we compared two models and demonstrated their performances. Furthermore, our object detection framework was able to identify the location of different land cover areas with a moderate number of samples. From the testing done with the two models, it can be concluded that the YOLOX model works better than the YOLOR model in LULC classification task even with moderate training size of low-mixed resolution sattelite images.

This research opens venues for interdisciplinary researchers and developers from agriculture, food, environment, and livestock to exploit these models on low/mixed resolution satellite images for effective planning, range land management, vegetation degradation, and different such activities.

In the future, with our framework, building accurate and stable land use/ land cover classifiers using low/mix resolution satellite images, new models can be built with a wide variety of classes such as settlements, industrial areas, forests, form lands, and rangelands.

Fig. 5-a Detected areas of vegetation/ non-vegetation by the YOLOX model on the test set.

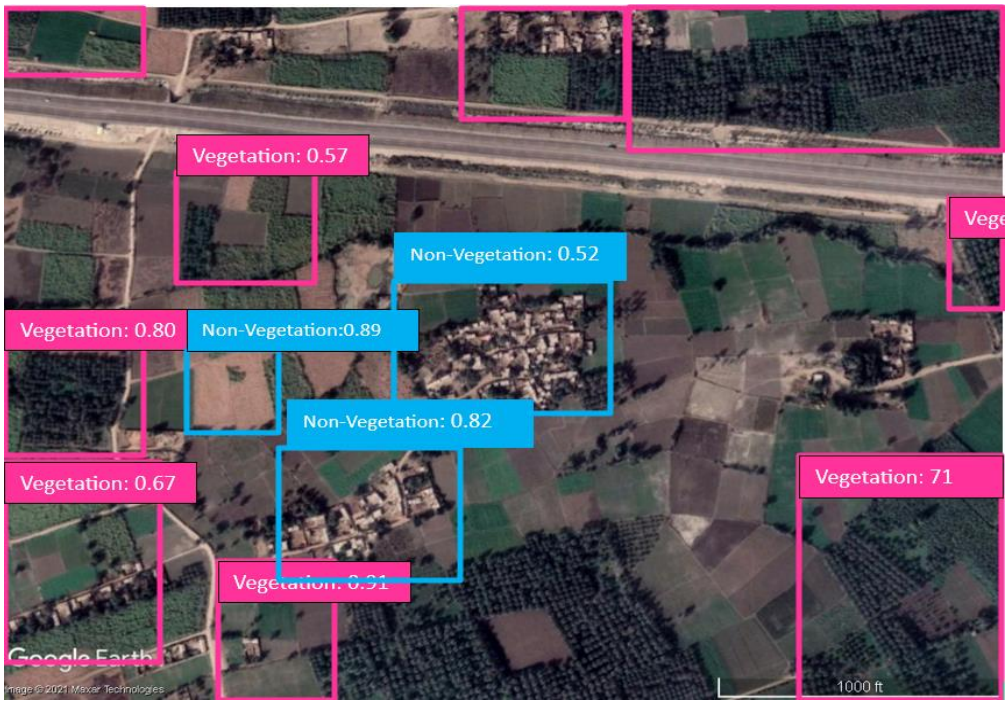

Fig. 5-b Detected areas of vegetation/ non-vegetation by the YOLOR model on the test set.

Fig. 5-c Detected areas of vegetation/ non-vegetation by the YOLOX model on the test set.

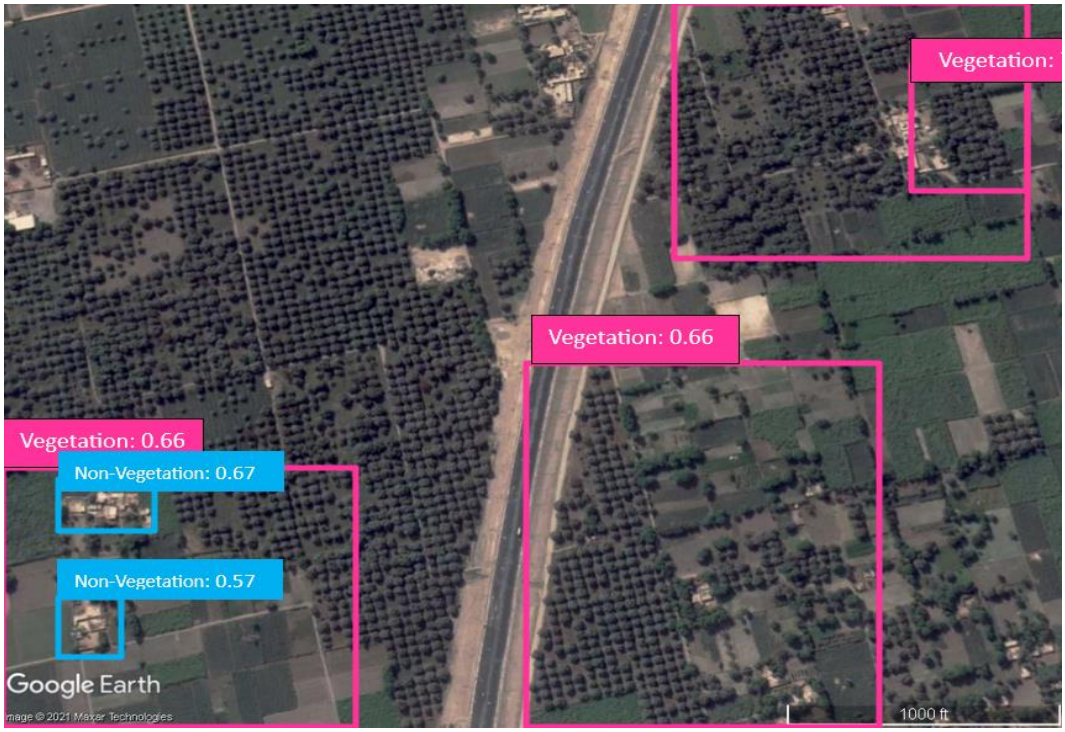

Fig. 5-d Detected areas of vegetation/ non-vegetation by the YOLOR model on test the set.

Fig. 5-e Detected areas of vegetation/ non-vegetation by the YOLOX model on the test set.

Fig. 5-f Detected areas of vegetation/ non-vegetation by the YOLOR model on the test set.

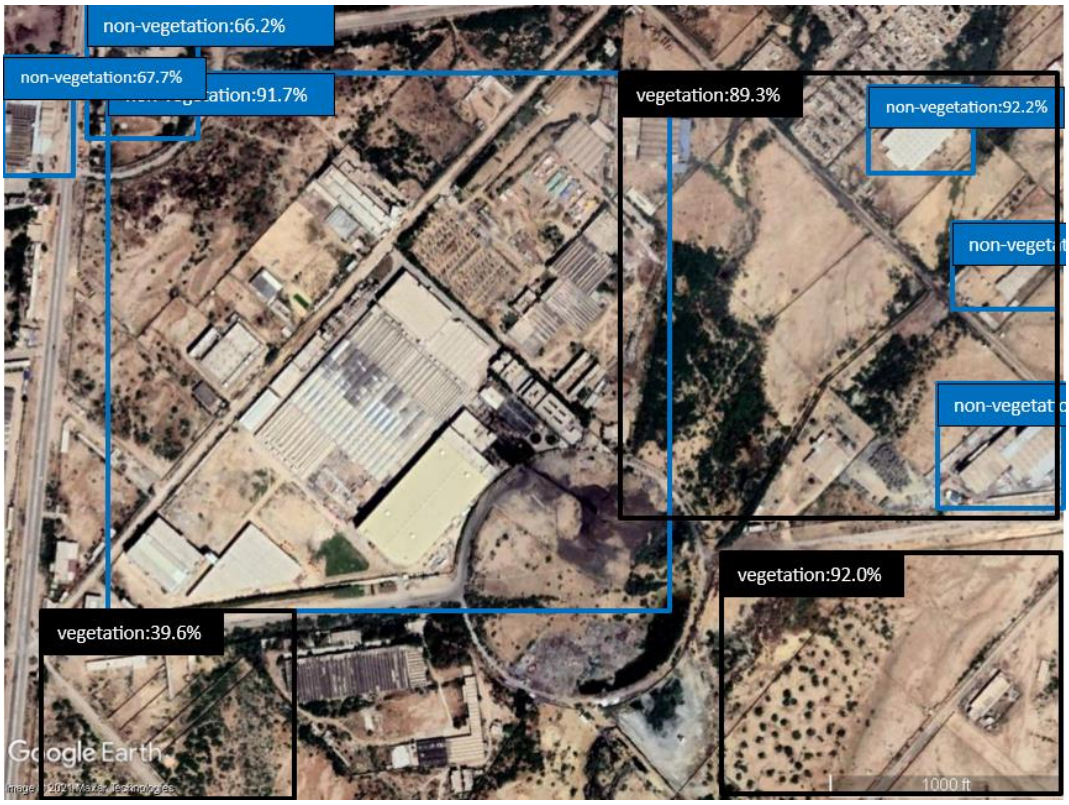

Fig. 5-g Detected areas of vegetation/ non-vegetation by the YOLOX model on the test set.

Fig. 5-h Detected areas of vegetation/ non-vegetation by the YOLOR model on the test set.

Fig. 5-i Detected areas of vegetation/ non-vegetation by the YOLOX model on the test set.

Author's Contribution: F.A., Conceived the idea; W.N., Designed the simulated work and M.A.N., did the acquisition of data; I.U., Executed simulated work, data analysis or analysis and interpretation of data and wrote the basic draft; A.B., Did the language and grammatical edits or Critical revision.

Funding: Work presented in this article was funded by HEC, Pakistan.

Conflicts of Interest: The authors declare no conflict of interest.

Acknowledgement: We are thankful to HEC for their financial support under Aghaz-e-Haqooq Balochistan scholarship and Government Innovation Lab (GIL) for computational resources.

References

[1] Huang, L., et al., OpenSARShip: A Dataset Dedicated to Sentinel-1 Ship Interpretation. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2018. **11**(1): p. 195-208. https://doi.org/10.1109/jstars.2017.2755672

[2] Basu, S., et al., DeepSat: a learning framework for satellite imagery, in Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems. 2015, Association for Computing Machinery: Seattle, Washington. p. Article 37. https://doi.org/10.1145/2820783.2820816

[3] Yang, Y. and S. Newsam, Bag-of-visual-words and spatial extensions for land-use classification, in Proceedings of the 18th SIGSPATIAL International

Conference on Advances in Geographic Information Systems. 2010, Association for Computing Machinery: San Jose, California. p. 270–279. https://doi.org/10.1145/1869790.1869829

[4]. Russakovsky, O., et al., ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 2015. **115**(3): p. 211-252. https://doi.org/10.1007/s11263-015-0816-y

[5] Penatti, O.A.B., K. Nogueira, and J.A.d. Santos. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? in 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). 2015. doi: 10.1109/CVPRW.2015.7301382.

[6] Prasad, D., Survey of The Problem of Object Detection In Real Images. International Journal of Image Processing (IJIP), 2012. **6**: p. 441. https://www.researchgate.net/publication/235216716_Survey_of_The_Problem_of_Object_Detection_In_Real_Images

[7] Lowe, D.G., Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 2004. **60**(2): p. 91-110. https://doi.org/10.1023/b:visi.0000029664.99615.94

[8] Viola, P. and M. Jones, Robust Real-Time Object Detection. Vol. 57. 2001. https://www.researchgate.net/publication/215721846_Robust_Real-Time_Object_Detection

[9] Dalal, N. and B. Triggs. Histograms of oriented gradients for human detection. in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. doi: 10.1109/CVPR.2005.177

[10] Liu, L., et al., Deep Learning for Generic Object Detection: A Survey. International Journal of Computer Vision, 2020. **128**(2): p. 261-318. https://doi.org/10.1007/s11263-019-01247-4

[11] Taufique, A.M., B. Minnehan, and A. Savakis Benchmarking Deep Trackers on Aerial Videos. Sensors, 2020. **20**, DOI: 10.3390/s20020547.

[12] Li, K., et al., Rotation-Insensitive and Context-Augmented Object Detection in Remote Sensing Images. IEEE Transactions on Geoscience and Remote Sensing, 2018. **56**(4): p. 2337-2348. doi: 10.1109/TGRS.2017.2778300.

[13] Ge, Z., et al., YOLOX: Exceeding YOLO Series in 2021. 2021. https://arxiv.org/abs/2107.08430

[14] Wang, C.-Y., I.H. Yeh, and H.-y. Liao, You Only Learn One Representation: Unified Network for Multiple Tasks. 2021. https://arxiv.org/abs/2105.04206

[15] Xia, G.S., et al., AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification. IEEE Transactions on Geoscience and Remote Sensing, 2017. **55**(7): p. 3965-3981. doi:10.1109/TGRS.2017.2685945.

[16] Basu, S., et al., DeepSat: a learning framework for satellite imagery. 2015. 1-10. doi:10.1145/2820783.2820816.

[17] Helber, P., et al. Introducing Eurosat: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. in IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium. 2018. doi:10.1109/IGARSS.2018.8519248.

[18] Uijlings, J.R.R., et al., Selective Search for Object Recognition. International Journal of Computer Vision, 2013. **104**(2): p. 154-171.

https://ivi.fnwi.uva.nl/isis/publications/bibtexbrowser.php?key=UijlingsIJCV2013&bib=all.bib

[19] Girshick, R., et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. in 2014 IEEE Conference on Computer Vision and Pattern Recognition. 2014. doi:10.1109/CVPR.2014.81.

[20] Ren, S., et al., Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017. **39**(6): p. 1137-1149. doi:10.1109/TPAMI.2016.2577031.

[21] Redmon, J., et al. You Only Look Once: Unified, Real-Time Object Detection. in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. doi:10.1109/CVPR.2016.91.

[22] Redmon, J. and A. Farhadi. YOLO9000: Better, Faster, Stronger. in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017. doi:10.1109/CVPR.2017.690.

[23.] Redmon, J. and A. Farhadi, YOLOv3: An Incremental Improvement. 2018. https://www.researchgate.net/publication/324387691_YOLOv3_An_Incremental_Improvement

[24] Liu, W., et al., SSD: Single Shot MultiBox Detector. Vol. 9905. 2016. 21-37. doi:10.1007/978-3-319-46448-0_2.

[25] Bochkovskiy, A., C.-Y. Wang, and H.-y. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020. https://www.researchgate.net/publication/340883401_YOLOv4_Optimal_Speed_and_Accuracy_of_Object_Detection

[26] al, g.j.e., yolov5. 2021. https://docs.ultralytics.com/yolov5/

[27] Liao, M., B. Shi, and X. Bai, TextBoxes++: A Single-Shot Oriented Scene Text Detector. IEEE Transactions on Image Processing, 2018. **27**(8): p. 3676-3690. doi:10.1109/TIP.2018.2825107.

[28] Nosaka, R., H. Ujiie, and T. Kurokawa. Orientation-Aware Regression for Oriented Bounding Box Estimation. in 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). 2018. doi:10.1109/AVSS.2018.8639332.

[29] Lei, J., et al., Orientation Adaptive YOLOv3 for Object Detection in Remote Sensing Images. 2019. p. 586-597. doi:10.1007/978-3-030-31654-9_50.

[30] Law, H. and J. Deng, CornerNet: Detecting Objects as Paired Keypoints. International Journal of Computer Vision, 2020. **128**. doi:10.1007/s11263-019-01204-1.

[31] He, Z.T.C.S.H.C.T., FCOS: Fully Convolutional One-Stage Object Detection. Computer Vision and Pattern Recognition, 2019. https://arxiv.org/abs/1904.01355

[32] Carrio, A., et al., A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles. Journal of Sensors, 2017. **2017**: p. 3296874. https://doi.org/10.1155/2017/3296874

[33] Zhu, X.X., et al., Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources. IEEE Geoscience and Remote Sensing Magazine, 2017. **5**(4): p. 8-36. https://doi.org/10.1109/mgrs.2017.2762307