Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 6 (2023) 1268 - 1275 ISSN Media Electronic: 2580-0760

Balinese Script Handwriting Recognition Using Faster R-CNN

Alif Adwitiya Pratama[1], Mahmud Dwi Sulistiyo[2*], Aditya Firman Ihsan[3]
[1,2,3]Department of Informatics, School of Computing, Telkom University, Bandung, Indonesia
[1]alifadwitiyap@student.telkomuniversity.ac.id, [2]mahmuddwis@telkomuniversity.ac.id, [3]adityaihsan@telkomuniversity.ac.id

Abstract

In Balinese culture, the ability to read Balinese script is one of the challenges young generations face. Advances in machine learning have proposed handwriting detection systems using both traditional and deep learning models. However, the traditional approach is usually impractical and is prone to inaccurate identification results. Convolutional Neural Network (CNN)-based models integrate feature extraction and classification into an end-to-end pipeline to increase performance. This research proposes that recognizing characters through an object detection approach makes an end-to-end process of localizing and classifying several characters simultaneously using the Faster R-CNN. Four CNN models, including ResNet-50, ResNet-101, ResNet-152, and Inception ResNet V2 were tested to detect 28 Balinese characters in a single form covering 18 consonants and 10 digits using Intersection over Union (IoU) thresholds: 0.5 and 0.75. ResNet-50 and Inception ResNet V2 achieve 0.991 mAP at IoU of 0.5, while Inception ResNet V2 excels at IoU of 0.75. Further analysis showed that class "nol" had the lowest Recall due to many undetected ground truths. Meanwhile, class "ba" had the lowest Precision due to its similarity with classes "ga" and "nga". This research contributes to experimenting with Faster R-CNN in detecting handwritten Balinese scripts.

Keywords: balinese script; faster R-CNN; handwriting

1. Introduction

Indonesia is home to diverse ethnic groups with unique cultures and traditions. However, this diversity is threatened by globalization because foreign cultures are considered more exciting or more unique and practical [1]. As in Balinese culture, while the Balinese language is still used for communication, using Balinese script in writing is becoming less common [2]. As a result, the ability to read Balinese script is one of the challenges young generations face [3]. The Balinese script is based on the Brahmi script and has two forms: a holy script used in religious rituals and a standard script used for daily writing and literary works [4], [5]. It contains over a hundred glyphs, including consonants, vowels, conjunct forms of consonants, digits, punctuation, and additional symbols for special consonants and musical notations [2]. An example of the consonant and numeric characters can be seen in Figure 1, typed using the Simbar Dwijendra font.

As a handwritten document, the Balinese script has various styles that writers use to write the different forms of glyphs. The length of the strokes, height, and width of a glyph can influence this difference. As a result, the same glyphs may appear vastly different from others. The problem of interclass similarity is also frequently encountered in Balinese script, as the shape of one glyph can be similar to another, with some parts being the same as those of another [4]. Recognizing handwritten characters is a subset of Optical Character Recognition (OCR) focus research.

Optical Character Recognition (OCR) is the field that deals with collecting and locating characters in paper files, photos, touch-display devices, and other media, then converting them into a device-encoded form [6].

Figure 1. Balinese Numeric and Consonant Characters

OCR is divided into two categories: Handwritten Character Recognition (HCR) focuses on recognizing handwritten text, and Printed Character Recognition

Accepted: 16-06-2023 | Received in revised: 16-09-2023 | Published: 26-11-2023

(PCR) focuses on recognizing printed text. HCR is considered more challenging than PCR due to the greater variety of handwriting styles compared to fonts [6]. HCR is subdivided into two categories: Online Recognition captures data for each line written by the user in real-time, and Offline Recognition only captures the result of the writing and thus is more difficult due to having fewer features [6]. This research is focused on offline handwriting recognition. Another research has explored online HCR and PCR discussed in [7] and [8].

Research [4] utilized Neighborhood Pixel Weight-Kirsch (NPW-Kirsch) and Histogram of Gradient (HoG) for extracting Balinese script handwriting features on lontar images. The classification using the K-Nearest Neighbor (KNN) and Neural Network (NN) models was employed. They experimented with a combination of feature extraction, different KNN neighbour distance calculations, and the number of hidden layers of the neural network. However, for detecting scripts, these traditional approaches are impractical due to the several stages to undergo and usually result in a low detection accuracy level; For instance, KNN using Euclidean calculation and HoG+NPW-kirsch combination feature extraction achieved an optimal accuracy of 44.88% [4].

Research [2] utilized a Gabor filter consisting of 64 filters and 7 zoning methods with a 10-pixel width to extract features from Balinese script handwriting on a lontar image resulting in 4,992 dimensions as the final feature. A single hidden neural network with 100, 200, and 300 neurons was employed to classify the features. The optimal result is 200 neurons with a training set accuracy of 97.20%, and the test set accuracy was 71.91%.

Research [3] utilized Image Centroid Zone (ICZ), Zone Centroid Zone (ZCZ), and Freeman Chain Code (CC) for extracting Balinese script handwriting features on paper images. The resulting features were classified using the Support Vector Machine (SVM). Experimenting with different kernel and feature extractor combinations. The optimal combination was ICZ, ZCZ, and CC feature extraction with linear kernels SVM, yielding an accuracy of 89.09%.

Research [9] utilized CNN with a residual network to classify Balinese script on lontar images. The authors experimented with different image sizes and transformed images, comparing their results to those of previous research. They achieved their best performance on a transformed black-and-white dataset with a size of 28×28, resulting in an accuracy of 83.47% and an F1 score of 86.80%. These results surpassed those of previous studies, leading the authors to conclude that using residual blocks positively impacts classification performance.

Traditional models require several stages: preprocessing, segmentation, feature extraction, and classification [3]. Convolutional Neural Network (CNN)-based models integrate feature extraction and classification stages into an end-to-end processing pipeline to increase performance as shown in [10], CNN outperforms traditional methods for numeric handwriting detection. Recognizing characters through an object detection approach with deep learning architecture makes the processing stages end-to-end in simultaneously localizing and classifying several characters. This ability was not possessed by other previous studies that apply traditional models. The Balinese script detection model in previous studies was considered less effective because the process of recognizing characters was carried out individually, requiring high computational costs, but with a high risk of low accuracy. Faster Region Convolutional Neural Network (Faster R-CNN) is an object detection method that has been successfully applied to the recognition of various types of handwriting, including kanji characters [11], digits [12], signatures [13], and text [14].

Previous research in [11]–[14] has shown that the Faster R-CNN model performs well in detecting handwriting. However, its performance in detecting Balinese script handwriting remains unknown until this research is conducted. This research proposes a method for detecting Balinese script handwriting on paper using a Faster R-CNN model. Four backbone CNN architectures using residual block were tested, namely: ResNet-50, ResNet-101, ResNet-152, and Inception ResNet V2, to detect 28 Balinese handwriting characters, including 18 consonants and 10 digits in a single form. Additionally, experiments were conducted using two different Intersection over Union (IoU) thresholds of 0.5 and 0.75 to assess the model's performance in the restricted IoU threshold.

This research contributes to experimenting with Faster R-CNN in detecting handwritten Balinese scripts. The proposed model integrates segmentation, feature extraction, and classification stages into an end-to-end pipeline to localize and classify multiple characters simultaneously.

2. Research Methods

As shown in Figure 2, the study followed a systematic approach that included several steps: manually gathering data and then preprocessing, including annotating, resizing, and splitting. After preprocessing the data, we augmented training data by generating new samples using data augmentation techniques, including rotating, changing image saturation, blurring, and noising. Finally, augmented data was used to train the model and will be evaluated using a test dataset with a different writer from the training dataset.

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

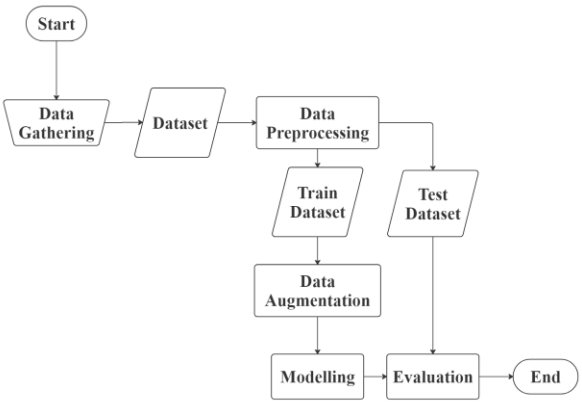


Figure 2. Flowchart of Research Methodology

2.1 Dataset

The research team collected the dataset used in this research, comprising 28 classes covering 18 consonants and 10 digits with a total of 253 data gathered from 25 native Balinese who have learned Balinese script writing through formal education. The respondents captured images using their mobile phone cameras, which the research team processed and annotated. However, the dataset has not yet been made public. Figure 3 presents a sample image from the dataset.

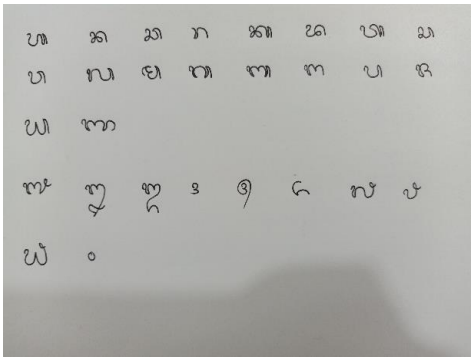


Figure 3 An Example of the Dataset

2.2 Data Preprocessing

Data preprocessing is a crucial step in any machine learning project because, in this step, the data will be processed so that it can become input for the model. In this study, we performed several steps as part of the data preprocessing. First, we annotated the images to label the consonants and numbers in each image. This annotation is necessary to ensure that the model can accurately recognize and classify the characters in the images. Next, all images were resized to 640×640 to ensure consistent input data. Finally, the dataset is split into training and test sets, with 70% of the data used for training and 30% for testing. The test set respondents are distinct from those in the training set to ensure the model can generalize well to unseen writing styles. This division resulted in a training set of 177 data and a test set of 76 data.

2.3 Data Augmentation

To increase the diversity and quantity of the training dataset, we performed data augmentation techniques on the processed training dataset. The techniques used in this study include rotating the image by -5 to 5 degrees, changing the image saturation by -15% to 15%, blurring the pixels up to 1 pixel, and adding 1% noise. These techniques were randomly applied to the images to produce new samples for the training dataset. As a result, we increased the size of the training dataset to 531 data. Figure 4 shows an example of augmented data from the input data in Figure 3.

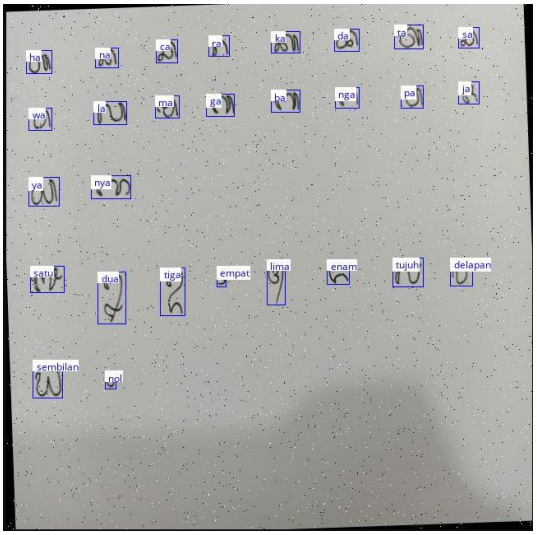


Figure 4 An Example of the Augmented Input Data

2.4 Convolutional Neural Network

CNN works with finding local area features in the image by utilizing various filters in the convolution layer to create a feature map, making it more efficient than using Multi-Layer Perception (MLP) only, as shown in research [15]. CNN architecture is generally followed by a pooling layer, where the results of the feature map will be sampled so that only dominant features in the image remain. The resulting feature map can be used as an input to MLP by flattening the shape of the feature map into the shape of the MLP input shown in Figure 5, or it can also be further developed by modifying the architecture so that performance can be improved.

Some examples of modified CNN architectures are in the Inception V1, ResNet, and Inception ResNet architectures. In the Inception V1 architecture, the modification made is by adding an inception block whose function is to divide the convolution and then recombine it to increase learning ability.

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

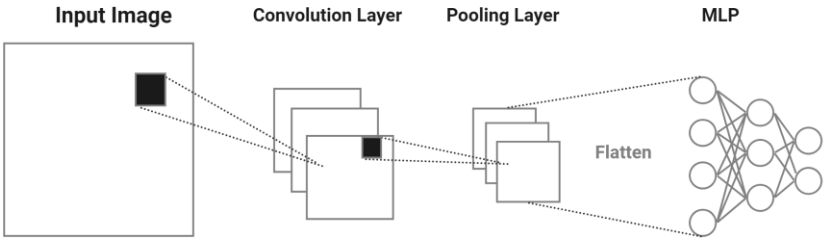


Figure 5 Overview of the General CNN Architecture [16]

In the inception block, regulation is carried out by adding a 1×1 convolution before the large convolution layer [16]. In the ResNet architecture, the modification made is by adding a residual connection to overcome the vanishing gradient problem where the weights will decrease with the depth of the network so that the ResNet architecture can have a very deep architecture [16]. Then Inception ResNet adds a residual connection similar to ResNet in the inception block to shorten the training [16].

2.5 Faster R-CNN

Faster R-CNN is an object detection method that combines the Regional Proposal Network (RPN) and Fast R-CNN. RPN is used to propose Regions of Interest (RoI) instead of Selective Search, used in previous methods. Fast R-CNN is an extension of R-CNN for RoI detection. The Faster R-CNN architecture divides the feature map generated by the CNN architecture for RPN and Fast R-CNN to reduce computation time [17], based on research [18] test time per image of Faster R-CNN 10 times faster than Fast R-CNN and 235 times faster than R-CNN with higher Mean Average Precision (mAP). Figure 6 provides an overview of the Faster R-CNN architecture.

Initially, the image becomes Faster R-CNN's input, and it will extract the main features using CNN to produce an RPN and Fast R-CNN feature map. The RPN layer outputs the location and objectness score of the RoI, which tells us whether the RoI contains objects or just backgrounds. It works by using anchors on a shared feature map and generating ROIs based on their ratio and size. However, many irrelevant RoI's are generated so this method will remove RoI that crosses image boundaries and apply non-maximum suppression with a threshold of 0.7 to remove overlapping anchors. After RPN proposes RoI's, Fast R-CNN detects proposed RoI classes by converting H×W RoI into H×W through max-pooling using h/H×w/W windows. The pooling result will then become the input of the fully connected layer with two outputs: class and RoI.

2.6 Evaluation

Evaluation is a necessary task to determine whether the model has good performance or not. Object detection often uses mAP as an evaluation metric. This metric is computed by averaging all Average Precision (AP) classes [19] which are calculated as Formula 1.

$$mAP = \frac{1}{c}\sum_{i-1}^{C} AP_i, \quad (1)$$

C is the total class, and APi_i is the precision for i-th class. AP is an area under curve Precision × Recall. Compute AP required finding detection larger than Intersection Over Union (IoU) [19] which is calculated as Formula 2.

$$IoU = \frac{area(RoI_p \cap RoI_{gt})}{area(RoI_p \cup RoI_{gt})}, \quad (2)$$

RoI_p is the predicted RoI and RoI_{gt} is the ground-truth RoI.

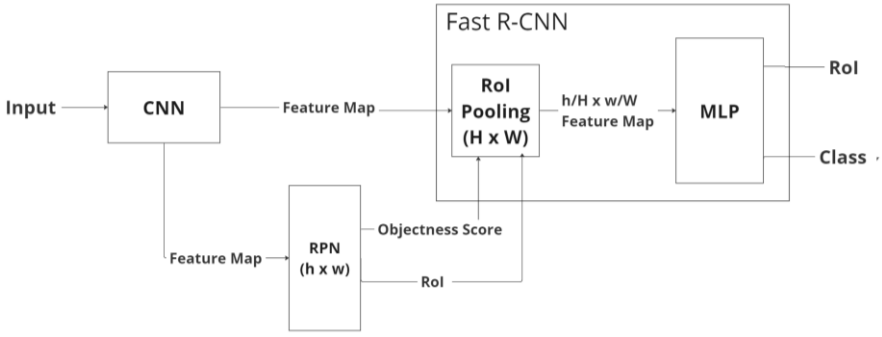


Figure 6. Overview of the Faster R-CNN Architecture [17]

Precision and Recall [19] are calculated as Formula 3 and Formula 4.

$$Pr(\tau) = \frac{\sum_{n=1}^{S} TP_n(\tau)}{\sum_{n=1}^{S} TP_n(\tau) + \sum_{n=1}^{N-S} FP_n(\tau)}, \quad (3)$$

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

$$Re(\tau) = \frac{\sum_{n=1}^{S} TP_n(\tau)}{\sum_{n=1}^{S} TP_n(\tau) + \sum_{n=1}^{G-S} FN_n(\tau)}, \quad (4)$$

Pr is precision, Re is recall, TP is a correct detection, FP is a non-existing object or misplaced detection of an existing object, and FN is an undetected ground truth. Detection confidence is supposed to be larger than the confidence threshold.

The next step is to sort detections in a descending way based on confidence rates and compute the AP [19] as Formula 5.

$$AP = \sum_{k=0}^{K} (Rr_k - Rr_{k+1}) Pr_{interp}(R_r(k)) \quad (5)$$

K is the total of prediction points, Pr_{interp} is the Interpolated Precision obtained by maximizing Precision corresponding to a higher Recall value than the current one, Rr_k is k-th sorted Recall value, and Rr_{k+1} is the next sorted Recall value.

2.7 Model Implementation

Development of the Faster R-CNN model in this research will be implemented using a pre-trained model available in Tensorflow 2 Detection Model Zoo and use Tensorflow Object Detection API for model development. This framework has been applied in various research to detect kanji characters [11], sign language [20], and traffic lights [21].

This research uses four different CNN architecture pre-trained models trained using the COCO dataset, including ResNet-50, ResNet-101, ResNet-152, and Inception ResNet V2. Every model will be fine-tuned using the default configuration with changes in num steps to 5,000 and batch size to 6. To check model performance, every model will be evaluated using mAP and the IoU thresholds of 0.5 and 0.75 while respecting the confidence threshold of 0.5.

3. Results and Discussions

The models were trained and evaluated using the Google Collaboratory platform with a V100 GPU and 24 GB of RAM. All models were trained on 531 augmented data and then tested on 76 data with different respondents from the training data. Each model was trained for 5000 steps with 6 batches using the default configuration. The resulting training loss between some models is shown in Figure 7.

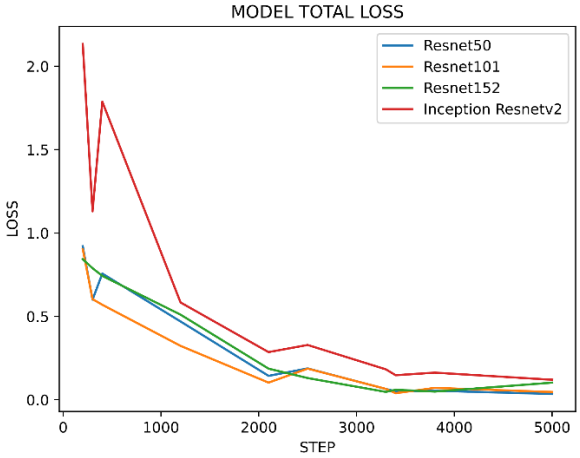


Figure 7. Total Losses from Some Different Models

Total loss training is extracted at steps 200; 300; 400; 1,200; 2,100; 2,500; 3,300; 3,400; 3,800; and 5,000. In the last step, the total loss reaches 0.035 on ResNet-50 architecture, 0.046 on ResNet-101 architecture, 0.102 on ResNet-152 architecture, and 0.120 on Inception ResNet v2 architecture. These results indicate that the model can learn features in Balinese handwriting feature. However, a training total loss value that is too low can indicate an overfitting of the model. Therefore, it is necessary to pay attention to the mAP value in the test data to observe better the model's performance, which can be seen in Figure 8.

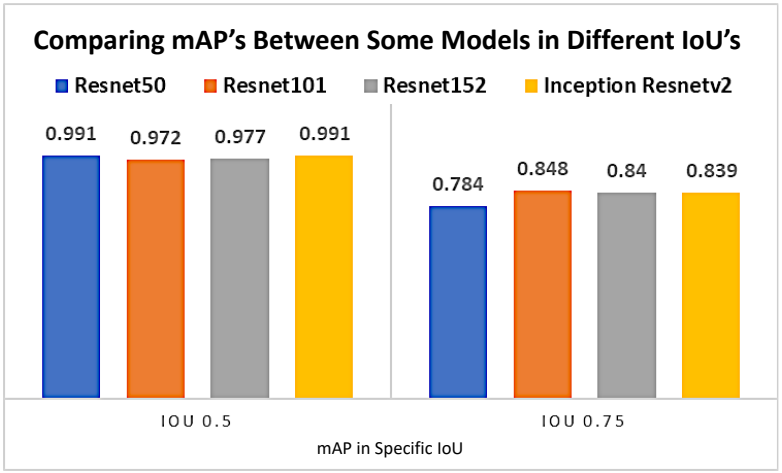


Figure 8. mAP Results from Different Models

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

According to Figure 8, there is a decrease in mAP when using an IoU threshold of 0.75. [11] suggests that setting the IoU threshold closer to 1 will require near-perfect RoI detection, which may result in the model not detecting more flexible RoI's. The significant drop in performance between the IoU threshold of 0.5 and 0.75 indicates that the model requires higher flexibility, as shown in Figure 9.

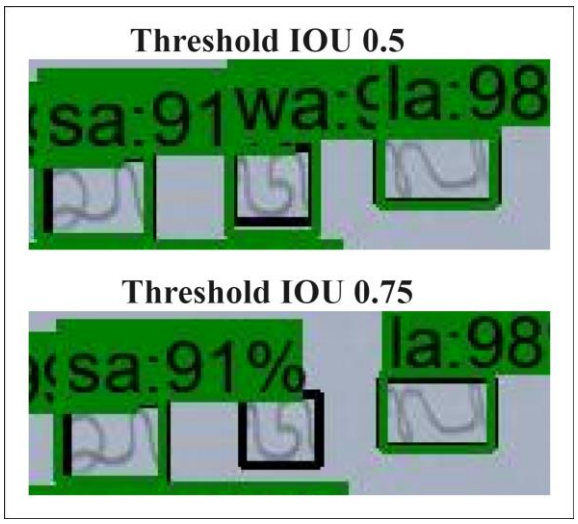


Figure 9. Low IoU Threshold Example

Figure 9 shows the detection results using the Inception ResNet v2 model with IOU thresholds of 0.5 and 0.75. The black box represents the ground truth, while the green box represents the correct detection result. At a threshold of 0.5, the model can detect the "sa", "wa", and "la" classes with confidence above 90%. However, with an IOU threshold of 0.75, the model does not detect the "wa" script because the detected RoI does not overlap perfectly with the ground truth RoI, as is the case with the "sa" and "la" classes.

Based on Figure 8, the mAP value was obtained using the evaluation function provided by the TensorFlow Object Detection API with the COCO detection metrics set. It was shown that the ResNet-50 and Inception ResNet V2 models achieved the highest mAP scores among the other models, with a score of 0.991 at an IoU threshold of 0.5. However, Inception ResNet V2 excels at a higher IoU threshold, i.e. 0.75.

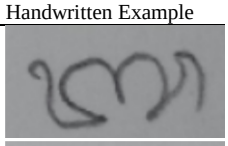
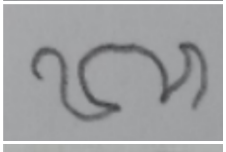
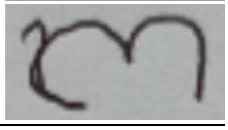
To evaluate the performance of each class, we use precision and recall metrics and generate a confusion matrix for the Inception ResNet V2 model with an IoU threshold of 0.5. The results are presented in Table 1, which shows the precision and recall for each class, and the confusion matrix, shown in Figure 10, provides more detailed information about the model's errors using a confusion matrix where columns show detection and rows show ground truth. While FN in the last columns indicates a false negative or undetected ground truth, and FP in the last row indicates a false positive or detection non-existing object.

Table 1. Each Class Precision and Recall

Class	Precision	Recall
satu	1	1
dua	1	1
tiga	1	1
empat	0.99	0.97
lima	1	1
enam	1	1
tujuh	0.97	1
delapan	0.99	0.99
sembilan	1	1
nol	1	**0.67**
ha	1	1
na	1	1
ca	1	1
ra	1	0.99
ka	0.99	0.99
da	0.95	0.99
ta	0.99	0.99
sa	1	0.92
wa	0.97	0.91
la	0.96	1
ma	0.95	0.99
ga	1	0.82
ba	**0.83**	1
nga	0.89	0.83
pa	0.97	0.91
ja	1	0.95
ya	0.91	0.99
nya	0.92	1

Table 1 and Figure 10 shows that the model's most challenging class to detect is class "nol" because out of 76 ground truth classes zero, 25 ground truths cannot be detected by the model, so this class has the lowest recall compared to other classes. Meanwhile, the class with the most detection errors is class "ba" because out of 92 "ba" detections, 11 detections have ground truth "ga", and 5 detections have ground truth "nga", so it has the smallest precision value compared to other classes. This error may occur because there is an interclass similarity between "ba", "ga", and "nga" characters. Examples of "ba" detection errors in "ga" and "nga" classes can be seen in Table 2.

Table 2. Detection Examples for "ba"

Ground Truth Class	Handwritten Example
ba	
ga	
nga	

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

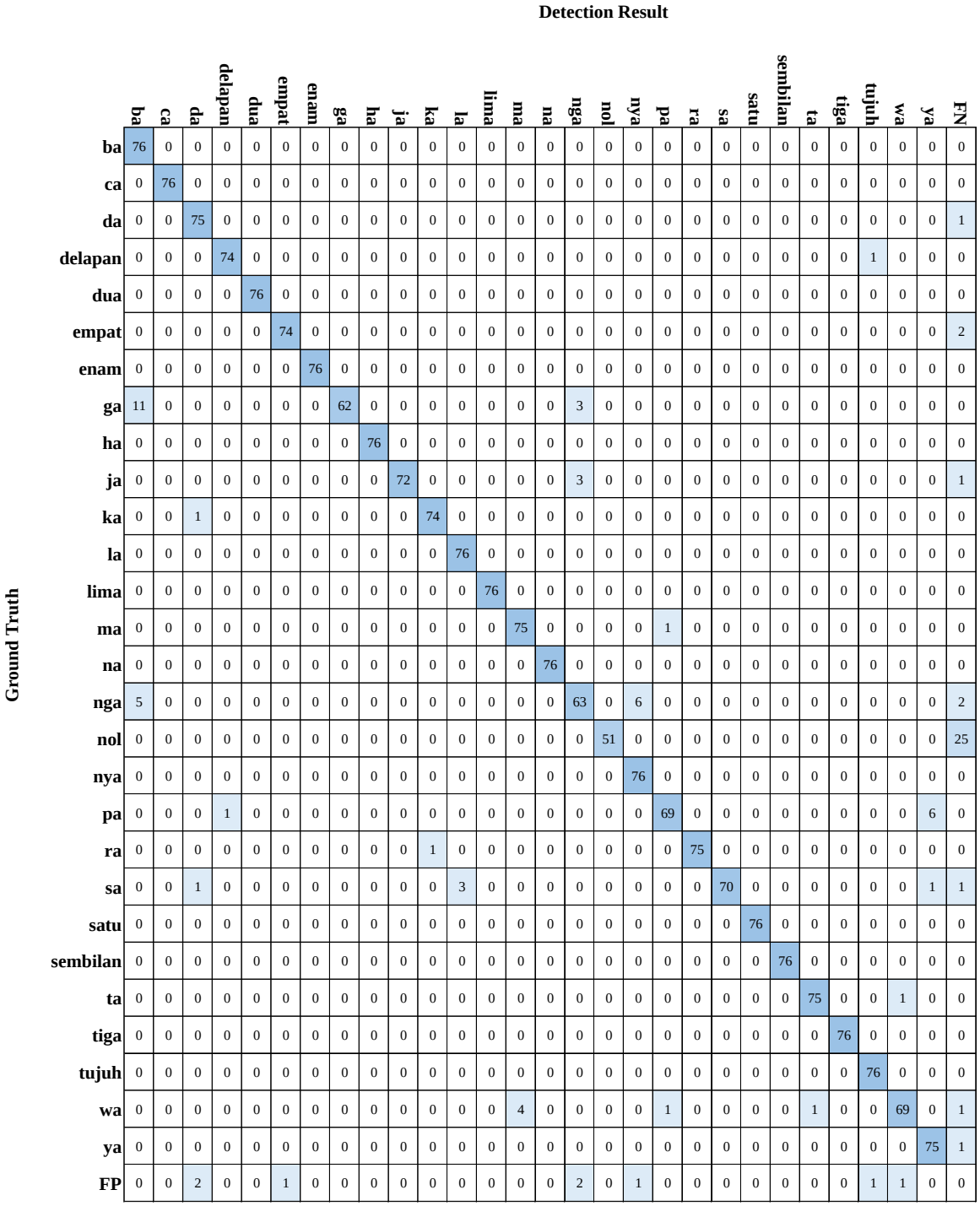


Figure 10. Confusion Matrix

Table 2 shows examples of “ba” detection with a different ground truth. The three classes share some similarities. Specifically, “ba” and “ga” have similar arches on both sides, while “ba” and “nga” have similar arches in the middle of the script.

4. Conclusion

This research addresses the challenge of using an object detection approach with Faster R-CNN to detect handwritten Balinese scripts more effectively and efficiently. The experimental results showed that the proposed deep learning-based model could learn the features of Balinese handwriting, as indicated by the decreasing total loss. The mAP evaluation metric shows that ResNet-50 and Inception ResNet V2 have the same mAP value of 0.991 at an IoU threshold of 0.5. However, Inception ResNet V2 excels at stricter IoU thresholds. Further analysis was conducted on the

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)

Inception ResNet V2 model by examining each class's precision and recall values, showing that the "nol" class has the lowest recall value due to many undetected ground truths. In contrast, the "ba" class has the lowest precision value due to an interclass similarity with "ga" and "nga" characters. For future work, we suggest using Balinese scripts written on Lontar leaves with more categories and comparing some object detection models such as Single Shot Detector (SSD) or You Only Look Once (YOLO).

Acknowledgement

The authors would like to express their gratitude to the Directorate of Research and Community Service, Telkom University for their generous support in a PTUPT research grant with the agreement letter number: 544/PNLT3/PPM/2023.

Reference

[1] H. M. I. Nahak, "UPAYA MELESTARIKAN BUDAYA INDONESIA DI ERA GLOBALISASI," *Jurnal Sosiologi Nusantara*, vol. 5, no. 1, pp. 65–76, Jun. 2019. doi: 10.33369/jsn.5.1.65-76.

[2] M. W. A. Kesiman, G. A. Pradnyana, and I. M. D. Maysanjaya, "Balinese glyph recognition with Gabor filters," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jun. 2020. doi: 10.1088/1742-6596/1516/1/012029.

[3] I. K. A. G. Wiguna and I. M. D. P. Asana, "Implementasi Zoning dan Fitur Arah Sebagai Ekstraksi Fitur Pada Pengenalan Tulisan Tangan Aksara Bali," *Jurnal RESISTOR (Rekayasa Sistem Komputer)*, vol. 4, no. 1, pp. 85–92, 2021. doi: 10.31598/jurnalresistor.v4i1.751.

[4] M. W. A. Kesiman, "Word recognition for the Balinese palm leaf manuscripts," in *Proceedings: CYBERNETICSCOM 2019 - 2019 IEEE International Conference on Cybernetics and Computational Intelligence: Towards a Smart and Human-Centered Cyber World*, Institute of Electrical and Electronics Engineers Inc., Aug. 2019, pp. 72–76. doi: 10.1109/CYBERNETICSCOM.2019.8875634.

[5] I. M. Windya, "Dinamika Pasang Aksara Bali: Sebuah Kajian Historis Sistem Ejaan," *Widyacarya: Jurnal Pendidikan, Agama dan Budaya*, vol. 2, no. 1, pp. 39–43, 2018. doi: 10.55115/widyacarya.v2i1.62.

[6] B. M. Vinjit, M. K. Bhojak, S. Kumar, and G. Chalak, "A Review on Handwritten Character Recognition Methods and Techniques," *Proceedings of the 2020 IEEE International Conference on Communication and Signal Processing, ICCSP 2020*, pp. 1224–1228, 2020. doi: 10.1109/ICCSP48568.2020.9182129.

[7] M. Abuzaraida, M. Elmehrek, and E. Elsomadi, "Online handwriting Arabic recognition system using k-nearest neighbors classifier and DCT features," *International Journal of Electrical and Computer Engineering*, vol. 11, pp. 3584–3592, Jun. 2021. doi: 10.11591/ijece.v11i4.pp3584-3592.

[8] G. and A. H. T. and S. D. Hailu Belay Birhanu and Belay, "Multi-font Printed Amharic Character Image Recognition: Deep Learning Techniques," in *Advances of Science and Technology*, T. and F. S. W. Zimale Fasikaw Atanaw and Enku Nigussie, Ed., Cham: Springer International Publishing, 2019, pp. 322–331. doi: 10.1007/978-3-030-15357-1_27.

[9] D. Made Sri Arsa, G. Agung Ayu Putri, R. Zen, and S. Bressan, "Isolated Handwritten Balinese Character Recognition from Palm Leaf Manuscripts with Residual Convolutional Neural Networks," in *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*, 2020, pp. 224–229. doi: 10.1109/KSE50997.2020.9287584.

[10] M. Jabde, C. Patil, S. Mali, and A. Vibhute, "Comparative Study of Machine Learning and Deep Learning Classifiers on Handwritten Numeral Recognition," Jun. 2022. doi: 10.1007/978-981-19-8094-7_10.

[11] A. Adole, E. Edirisinghe, B. Li, and C. Bearchell, "Investigation of Faster-RCNN Inception Resnet V2 on Offline Kanji Handwriting Characters," *ACM International Conference Proceeding Series*, 2020. doi: 10.1145/3415048.3416104.

[12] S. Albahli, M. Nawaz, A. Javed, and A. Irtaza, "An improved faster-RCNN model for handwritten character recognition," *Arab J Sci Eng*, vol. 46, Jun. 2021. doi: 10.1007/s13369-021-05471-4.

[13] S. K. and B. Bajpai Ashutosh and Wupadrasta, "Comparative Study of Faster Region-Based Convolutional Neural Networks with Inception V2 and Single Shot Detector with Inception V2 on Their Signature Detection Capabilities," in *Proceedings of International Conference on Big Data, Machine Learning and Applications*, S. and B. V. E. Patgiri Ripon and Bandyopadhyay, Ed., Singapore: Springer Singapore, 2021, pp. 227–240. doi https://doi.org/10.1007/978-981-33-4788-5_19.

[14] J. Yang, P. Ren, and X. Kong, "Handwriting Text Recognition Based on Faster R-CNN," in *2019 Chinese Automation Congress (CAC)*, 2019, pp. 2450–2454. doi: 10.1109/CAC48633.2019.8997382.

[15] A. Botalb, M. Moinuddin, U. M. Al-Saggaf, and S. S. A. Ali, "Contrasting Convolutional Neural Network (CNN) with Multi-Layer Perceptron (MLP) for Big Data Analysis," in *2018 International Conference on Intelligent and Advanced System (ICIAS)*, 2018, pp. 1–5. doi: 10.1109/ICIAS.2018.8540626.

[16] L. Alzubaidi *et al.*, *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, vol. 8, no. 1. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 6, pp. 1137–1149, 2017. doi: 10.1109/TPAMI.2016.2577031.

[18] W. Li, "Analysis of Object Detection Performance Based on Faster R-CNN," *J Phys Conf Ser*, vol. 1827, no. 1, p. 12085, Mar. 2021. doi: 10.1088/1742-6596/1827/1/012085.

[19] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. Da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics (Switzerland)*, vol. 10, no. 3, pp. 1–28, Feb. 2021. doi: 10.3390/electronics10030279.

[20] A. and M. R. and S. S. Srivastava Sharvani and Gangwar, "Sign Language Recognition System Using TensorFlow Object Detection API," in *Advanced Network Technologies and Intelligent Computing*, S. K. and P. K. K. and V. A. and V. P. Woungang Isaac and Dhurandher, Ed., Cham: Springer International Publishing, 2022, pp. 634–646. doi: 10.48550/arXiv.2201.01486.

[21] İ. Kılıç and G. Aydin, "Traffic Lights Detection and Recognition with New Benchmark Datasets Using Deep Learning and TensorFlow Object Detection API," *Traitement du Signal*, vol. 39, pp. 1673–1683, Jun. 2022. doi: 10.18280/ts.390525.

DOI: https://doi.org/10.29207/resti.v7i6.5176
Creative Commons Attribution 4.0 International License (CC BY 4.0)