Oracle Bone Inscription Detection

A Survey of Oracle Bone Inscription Detection Based on Deep Learning Algorithm

Jici Xing
Key Laboratory of Oracle Bone Inscriptions Information Processing Ministry of Education
jicixing@gmail.com

Guoying Liu*
School of Computer and information Engineering
Anyang Normal University
Anyang
guoying.liu@aynu.edu.cn

Jing Xiong
School of Computer and information Engineering
Anyang Normal University
Anyang
jingxiong@aynu.edu.cn

ABSTRACT

The goal of this paper is to serve as a guide for selecting a detection architecture for Oracle Bone Inscription (OBI) that achieves the right speed/memory/accuracy balance for a given platform. Many successful systems have been proposed in recent years, but directly migrating these methods to OBI data may lead to unsatisfying performance due to the corrosion, noise, and distribution. We present a unified implementation of the Faster R-CNN [1], SSD [2], YOLOv3 [3], RFBnet [4], and RefineDet [5] systems which we view as experiment architectures and trace out the speed/accuracy trade-off. The method with the best overall performance was selected as the baseline. Then a series of improvements were made according to the characteristics of the data. The experiment shows that our method reaches the F-measure of 80.1, nearly 2% higher than the baseline. Experimental data and algorithms will soon be available at http://jgw.aynu.edu.cn.

CCS CONCEPTS

• Software and its engineering • Software creation and management • Designing software

KEYWORDS

Oracle, Bone, Inscription, Detection

1 Introduction

Oracle Bone Inscription (OBI) refers to some of the oldest characters of Chinese words, which are hieroglyphic signs inscribed onto cattle bones or turtle shells with sharp objects about 3000 years ago. OBI is an important medium for exploiting the political systems, economic status, and social lives of the Shang Dynasty (about 1600 B.C. -1046 B.C.). However, few people have the literacy of OBIs. The detection and recognition of OBIs, which combines archaeology, history, philology, and literature, requires people to have plenty of knowledge and years of experience.

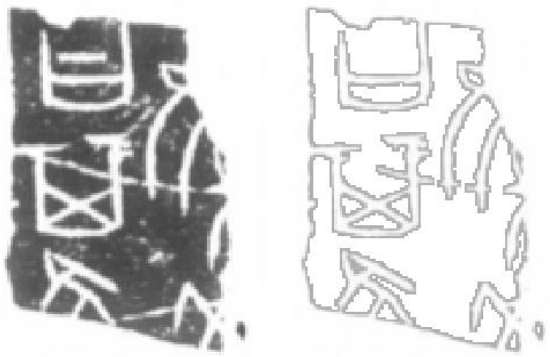

Figure 1: Oracle Bone Inscription Image.

As shown in Figure 1, some OBI examples are depicted. It is easy to find that each character looks like an undirected graph composed of intersections, lines, or curves. Therefore, many researchers attempted to recognize OBIs by making use of graph theory. By cascading two-level graph coding and one-level endpoint feature coding, Feng Li and Xinlun Zhou [6] proposed to automatically recognize off-line OBIs. Meng Lin [7] described OBIs by topological coding and recognized them based on topological registration. Qingsheng Li [8] coded each OBI by the adjacent inverse matrix and recognized OBIs based on the theory of graph isomorphism. Some other researchers tried to recognize OBIs by different kinds of character features. Xiaoqing Lv etc. [9] used Fourier descriptor of curvature histogram to describe OBIs, Lin Meng, etc. [10-13] employed Hough Transform to extract line features of OBIs, While Feifei Feng, etc. [14] adopted Mathematical Morphology to extract character features. In [15], Jun Guo, etc. proposed a hierarchical representation for OBIs, which combines a Gabor-related low-level representation and a sparse-encoder-related mid-level representation.

AIIPCC '19, December 19–21, 2019, Sanya, China

ACM ISBN 978-1-4503-7633-4/19/12…$15.00
https://doi.org/10.1145/3371425.3371434

To accurately identify the OBIs, first of all, we need to detect their location. However, these efforts mentioned above are aiming at solving the problem of the classification for each character on OBIs. Few studies have been found to perform the detection task.

Inspired by object detection and scene text detection in the field of deep learning, we regard oracle script as a special kind of data objects and their detection task can be performed by resorting to some popular methods [1-5], but directly migrating these methods to oracle datasets presents some problems that can lead to performance degradation. Compared with the usual data, oracle detection has the following features:

1. It is easy to connect two independent characters together by nick and noise.
2. Noise points caused by corrosion or excavation brings many false positives.
3. Incomplete characters which were not labeled may be classified as true positive.
4. Densely distributed patterns may cause a secondary overlap dilemma.

In this paper, we compared the performance of the popular object detection algorithms in the OBI dataset and chose the one with the best performance as our baseline. Moreover, a series of improvements were introduced according to the characteristics of OBI data. To be specific, we used the K-means++ algorithm [16] to cluster anchor boxes, making it closer to the original size. Then, a variety of synesthetic noises were inserted to augment the training data simulating the real rubbing interference. Finally, Focal Loss [17] and mixed-precision [18] were utilized to enhance model accuracy while compressing the memory footprint.

2 A Brief Review of Object Detection

2.1 Methods before the Deep Learning Era

Before the deep learning era, most detection methods [19] adopt Connected Components Analysis [20-25] or Sliding window [26-29] based methods. Connected Components Analysis-based methods firstly extract candidate components through a variety of ways (e.g., color clustering or extreme region extraction), and then filter out non-text components using manually designed rules or classifiers automatically trained on hand-crafted features. In sliding window classification methods, windows of varying sizes slide over the input image, where each window is classified as text regions or not. Those classified as positive are further grouped into text regions by morphological operations, Conditional Random Field, and other alternative graph-based methods.

2.2 Object Detection in Deep Learning

Motivated by the thriving of deep learning-based object [30] or text [31] detection architectures, we thought that oracle characters as a particular object could get benefits from these fields. There are two main trends in the field of object detection: one-stage and two-stage. The two-stage approach consists of two parts, where the former (e.g., Selective Search [31], EdgeBoxes [32], DeepMask [33] [34], RPN [1]) generates a sparse set of candidate object proposals, and the latter determines the accurate object regions. Notably, the two-stage approach (e.g. R-CNN [35], SPPnet [36], Fast R-CNN [37], Faster R-CNN [1]) achieved dominated performance on several challenging datasets (e.g. PASCAL VOC 2012 [38] and MS COCO [39]). After that, numerous effective techniques were proposed to further improve the performance, such as architecture diagram [40-42], training strategy [43, 44], contextual reasoning [45-48] and multiple layers exploiting [49-52].

The one-stage approach considers high efficiency, which attracts much more attention recently. Sermanet et al. [53] present the OverFeat method for classification, localization, and detection based on deep convolutional networks, which is trained end-to-end from raw pixels to ultimate categories. Redmon et al. [54] use a single feed-forward convolutional network to directly predict object classes and locations, called YOLO, which is extremely fast. After that, YOLOv2 [55] is proposed to improve YOLO in several aspects, add batch normalization, high resolution classifier, and anchor boxes. Liu et al. [2] propose the SSD method, which spreads out anchors of different scales to multiple layers within a convolution network and enforces each layer to focus on predicting objects of a specific scale. DSSD [56] introduces additional context into SSD via deconvolution operation to improve accuracy. DSOD [57] designs an efficient framework and a set of principles to learn object detectors from scratch, following the network structure of SSD. To improve accuracy, some one-stage methods [58] aim to address the extreme class imbalance problem by redesigning the loss function or classification strategies. Although the one-stage detectors have made good progress, their accuracy still trails that of two-stage methods.

3 Experimental Architecture

In this paper, we primarily focus on five popular architectures: Faster R-CNN, SSD, YOLOv3, RefineDet, and RFBnet.

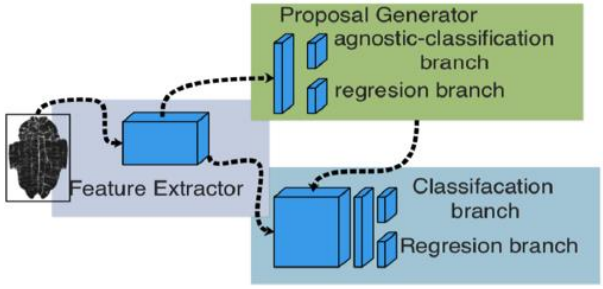

Figure 2: Faster R-CNN.

3.1 Faster R-CNN

In Faster R-CNN settings (Figure 2), detection is carried out in two stages. In the first stage, called the region proposal network, images are processed by a feature extractor (e.g., VGG-16), and

features at some selected intermediate level (e.g., conv5) are used to predict class-agnostic box proposals. In the second stage, these box proposals are used to crop features from the same intermediate feature map which are subsequently fed to the remainder of the feature extractor to predict a class and coordinates for each proposal.

3.2 SSD

SSD (Figure 3) is a typical representative of the one-stage method. Like YOLO, feature maps in different layers are directly used to predict without region proposal processing. Based on the VGG feature extractor, six convolutional layers (two of them modified from linear layers, others are additional) are inserted. Then two branches are used to predict confidence and coordinates respectively. Compared with Faster R-CNN, SSD scarifies a little accuracy but get a noticeable improvement in speed.

Since YOLOv3 is similar to SSD in overall flow, we use one graph for simplicity.

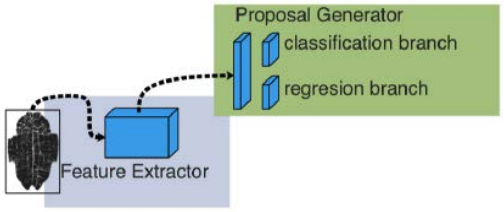

Figure 3: SSD and YOLO.

3.3 YOLO

YOLO is an object detection system targeted for real-time processing. It divides the input image into some grids. Each grid cell predicts a fixed number of boundary boxes. Every box has coordinates, size, and confidence score. All operations are performed simultaneously in one process (Figure 3). In this paper, YOLOv3 is mainly discussed.

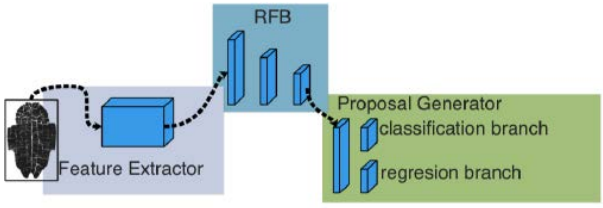

Figure 4: RFBnet.

3.4 RFBnet

RFBnet (Figure 4) belongs to one stage family where the feature generation processing is enhanced by some Receptive Field Blocks (RFB) with a lightweight computation. RFB simulates the human visual receptive field to enhance the power of feature extracting. Specifically, the RFB designing is moderated from Inception but insert dilated convolution [59] to enlarge the receptive field.

3.5 RefineDet

RefineDet (Figure 5) holds the superiority between one-stage and two-stage methods. Three modules compose the main architecture: Anchor Refinement Module (ARM), Object Detection Module (ODM), and Transfer Connection Block (TCB). Like RPN, the ARM can filter out some background bonding boxes, maintaining a balanced ratio in positive and negative samples. It also roughly regresses the selections for future processing. ARM also has two branches to get proposals, one for coordinates, the other for positive or negative classes. Deing different from RPN which extracts features from a certain layer, ARM works on multi-layers. Then TCB laterally connects feature maps and fuses features from the ARM to the ODM. Finally, ODM proceeds multi-classification and regression. Notably, boxes in ODM are refined by ARM and features are enriched by TCB.

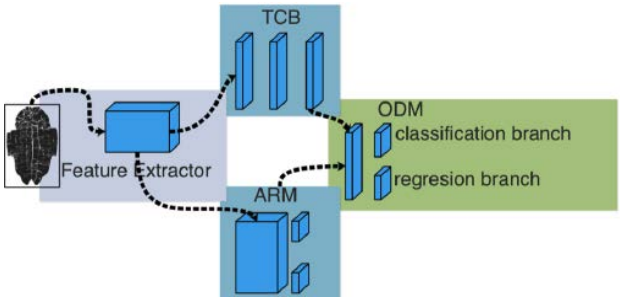

Figure 5: RefineDet.

3.6 Our Baseline and Designing

Our baseline is the method with the best overall performance in the experiment architectures mentioned above. Then, a series of improvements will be performed according to the characteristics of the data OBI data.

Data augmentation: To enhance the generalization ability in various scales and aspect ratios, we apply data augmentation to enlarge scene text datasets:

1. Randomly horizon flip with a probability of 0.5.
2. Randomly resize the height and width of images to 640- 1280 individually, without keeping the original aspect ratio.
3. Randomly select on 640×640 crop region from the resized image.
4. Synthesis of 9 different types of noise. There was a 30 percent chance that any one of these images would be superimposed on the original data during training, with transparency in [0.0, 0.15].

Anchor: The anchor can be quantified by three parameters: the base scale of anchors, the feature maps where anchors searched, and the aspect ratios of anchors.

The detail of generating aspect ratios is as follows: firstly, analyzing the data-augmented ground truth bounding box's size, then using the k-means++ algorithm to cluster sizes (width and

height). Finally, the center point of the front K cluster serves as the sizes of anchors.

Focal Loss: we adopt Focal Loss to learn the hard samples. Followed by [58]. Firstly, we sort the samples and then reweight them to update the network.

Mixed Precision: we use apex [60] to reduce the training requirements, which compresses memory bandwidth and storage requirements by 2X. Bandwidth-bound operations can be realized up to 2X speedup.

4 Experiment

4.1 Dataset

In this section, we discuss in detail the properties of oracle data and how it differs from standard benchmarks.

Noise: After being buried in the ground for more than 3,000 years, the characters on the bones became blurred after long-term corrosion. As shown in Figure 6, this image was first printed by rubbing the original oracle bones, then bound into a book, and finally sampled by a high-resolution scanner, which had a lot of texture noise and cracks. These naturally occurring disturbances are irregular and difficult to reproduce through traditional data augmentation.

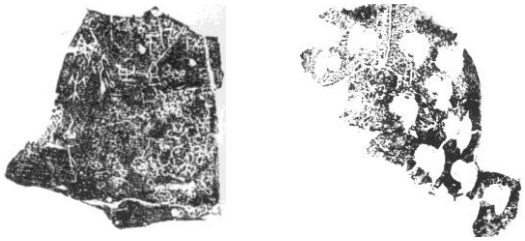

Figure 6: Noise on OBIs.

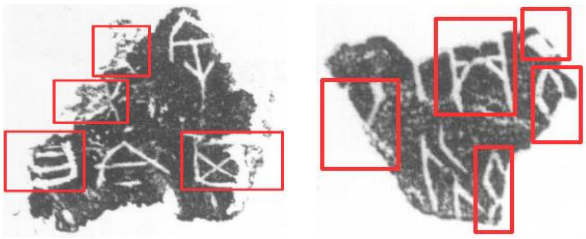

Figure 7: Fragment.

Fragment: The pieces of oracle bones are easy to be broken when unearthed, so a large number of oracle bones fragments were produced. As illustrated in Figure 7, the broken characters often appeared on the edge of the fragments, which were very similar to the natural texture of the corresponding complete ones, making it extremely difficult to detect and identify them.

Distribution: The characters on the same piece of oracle bones vary in size, direction, and distribution, significantly increasing the difficulty of detection and recognition. In the 56,743 oracle bone inscriptions, there are 1,425 words. Among them, there are 366 common words, 500 secondary common words, and 559 rare words.

Variant: The oracle bone characters appeared in the Yin and Shang dynasties when there was no uniform writing standard. Besides, the Shang and Zhou dynasties (about 1600 B.C. – 256 B.C.) spanned more than 1300 years. The evolution of the characters is noticeable, leading to a large number of variant characters in oracle bone inscriptions. Statistically, there are 1,032 sets of variants in a total of 3,085 characters, accounting for 49.5 percent. The appearance of variant characters in oracle bone inscriptions is very frequent, which brings great difficulties to the detection and recognition.

Dataset Generation: We have collected 9500 OBI rubbings (up to now) from [61, 62] by a high-resolution scanner then labeled every character with the upper left and lower right coordinates. Notably, this work only focuses on the detection task, so the number of classes in this dataset are all regarded as one. Our data will be updated and published at http://jgw.aynu.edu.cn.

4.2 Experiment Analyses

Table 1: Evaluations on Experiment Architecture.

Method	F	P	R	Speed	Memory
FRCNN	0.766	0.754	0.778	2FPS	1714MB
SSD	0.753	0.748	0.758	5FPS	668MB
YOLOv3	0.78	0.776	0.784	17FPS	828MB
RefDet	0.778	0.752	0.805	14FPS	1451MB
RFBnet	0.775	0.761	0.789	16FPS	991MB

Experiment architecture: We choose five mature object detection frameworks as experimental objects: Faster R-CNN [1], SSD [2], YOLOv3 [3], RefineDet [4], RFBnet [5]. All the experiments architectures are implemented in pytorch [63] and trained on four Nvidia Titian GPU cards. We replaced the original data loader to our images and set the number of classes to one. Specifically, our images are all grayscale. To adapt to the input of the network, we make three copies of the channel to form pseudo-RGB images. We use the F-measure (Equation (1)) to evaluate these modified architectures.

$$F_{measure} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (1)$$

The performance is summarized in Table 1. Notably, we take the maximum at 200 training epochs for a fair comparison. Interestingly, some characters we do not label due to they were damaged or blurred can still be captured. These otherwise correct predictions would be treated as false positives, so we only use this measure to ensure architectures converged. We believe that more training time and fine-tuning can lead to better results. To observe the performance of meta-architectures, we select three sets of representative images to observe the performance. In Figure 8, characters are a similar size and are uniformly distributed and closed to each other. In this case, the performance met our expectations. In Figure 8, the image

contains harsh noise and irregular nicks, and textures are carved around characters, which makes the background more complex and the clues challenging to recognize.

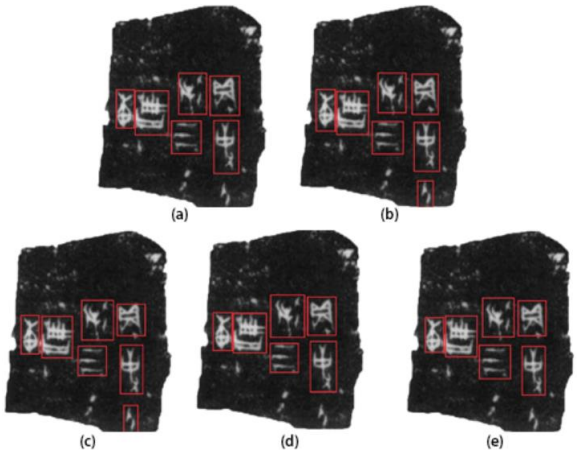

Figure 8: The OBI Characters are Evenly Distributed and of Similar Size: (a) Faster Rcnn; (b) SSD; (c) RFBnet; (d) RefineDet; (e) YOLOv3.

In Figure 9, the image contains harsh noise and irregular nicks, and textures are carved around characters, which makes the background more complex and the clues challenging to recognize. Consequently, the accuracy of this image has declined respectively. In Faster R-CNN, the result presents overlapping marks that may be due to the surrounding nicks and textures which connects the characters that should be independent. In RefineDet, it still works fine during these disturbances. The RFBnet only detects one character and misses another one. SSD and YOLOv3 contain overlaps. They treat half a character as a whole. It turns out that the incomplete character with surrounding noise being convoluted makes its feature closer to a full character.

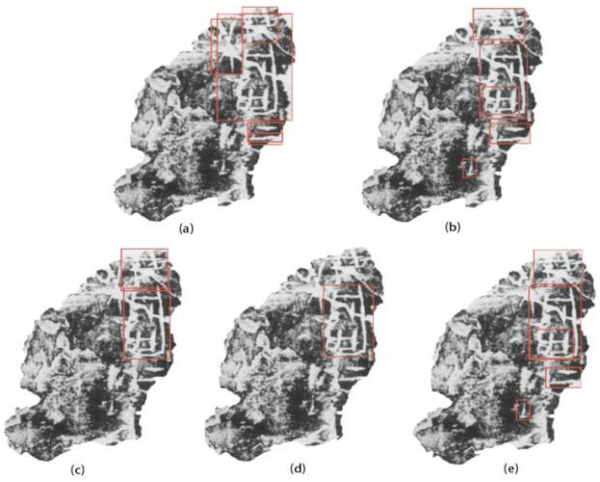

Figure 9: The OBI Image Contains Serious Noise: (a) Faster Rcnn; (b) SSD; (c) RFBnet; (d) RefineDet; (e) YOLOv3.

Figure 10 is a severe challenge for these detectors, which covers almost all the difficulties of OBI data. The size of this image is large, but it will be reduced when the image was fed into detectors, which makes the clues smaller. RFBnet failed. RefineDet also misses a lot of characters. The SSD is satisfying but also misses some incomprehensible characters. The results of Faster R-CNN and YOLOv3 are fantastic and they even detect some characters that we cannot label, but overlaps still exist.

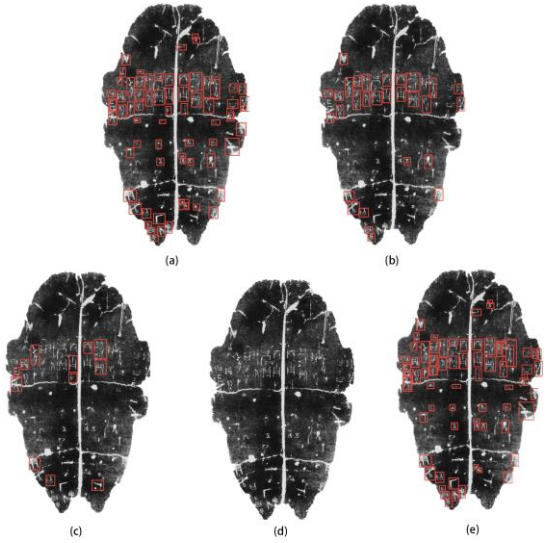

Figure 10: The OBI Characters on the Image are Scattered, with Different Sizes, Random Directions and Fracture Marks: (a) Faster Rcnn; (b) SSD; (c) RFBnet; (d) RefineDet; (e) YOLOv3.

Our Baseline and Modified Method: It can be concluded from the above experiment that YOLOv3 has the best comprehensive performance. Interestingly, it has been found that YOLO is also widely used in other fields, including video game detection [64], license plate detection [65], and 3d detection [66], all of which have excellent performance. That may be due to the sophisticated design of DarkNet and generalization capabilities of the entire processing mechanism. YOLOv3 runs faster than the others on images because it is simpler in structure and enjoys on-stage fashion (without RPN [1] head). It has been found that YOLOv3 performs better among other methods with satisfying precision, lightweight computation capacity, and high speed. Therefore, we use the YOLOv3 as our baseline and modify it as demonstrated in Section 3.6.

Ablation Study: To verify the effectiveness of our modified method, we do a series of comparative experiments on the collected OBI dataset as described in Table 2.

The Influence of Clustered Anchor: We study the effect of the use of clustered anchor by training a K-means++ [67] cluster, and set K (number of clusters) gradually from 6 to 11 as shown in Figure 11. However, a large K (number of anchors per gird) will slow down the model convergence. To keep a good balance of performance and speed, we keep k to 9 with corresponding centers ([48,50], [91,105], [159,238], [192,112], [288,409],

[339,203], [464,698], [577,395], [973,911]) by default in the following experiments.

Table 2: Ablation Study.

Clustered Box	Data Augmentation	Focal loss	Mixed Precision	F measure
√				0.785
√	√			0.791
√	√	√		0.801
√	√	√	√	0.801

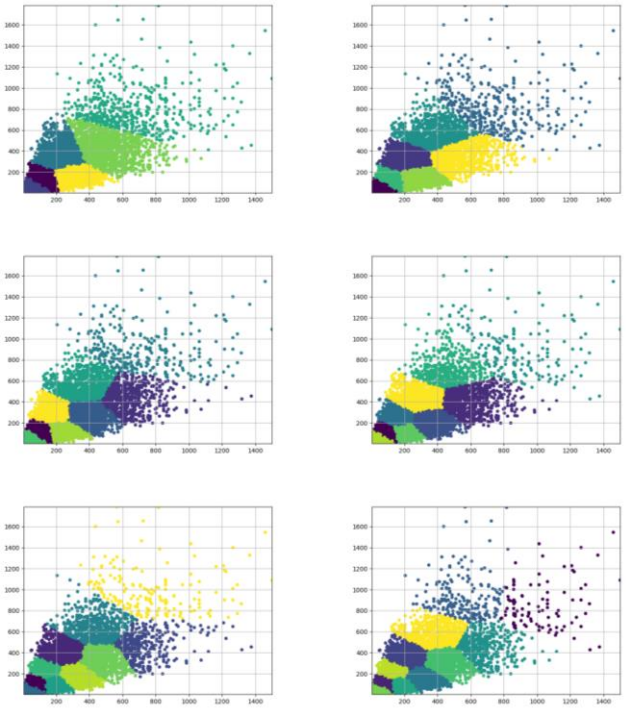

Figure 11: Anchor Clustering Result.

The Influence of Data Augmentation: Data augmentation has proven to play an essential role in various tasks in the field of computer vision, especially when data is insufficient. Mirroring, rotation, and scaling have been widely used in object detection, so this part mainly displays the syn- thesis of noise for augmentation. We synthesized 9 different types of noise, as illustrated in Figure 12. There was a 30 per- cent chance that any one of these images would be superimposed on the original data during training, with transparency in [0.0, 0.15].

We make a comparison between the model with data augmentation and without that. Compared to the model without data augmentation (see Table 3), the model with data augmentation can make about 0.6% improvement on F-measure.

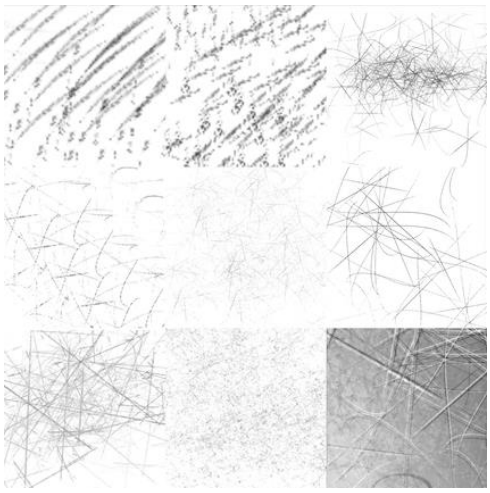

Figure 12: Artificial Synthesis Simulation OBI Noise Image.

Table 3: Influence of Data Augmentation.

	Data Augmentation	Without Data Augmentation
P	0.791	0.794
R	0.799	0.806
F	0.795	0.800

The Influence of Focal Loss: To investigate the effectiveness of focal loss, we firstly replace the loss function with the traditional cross-entropy loss to make the final prediction. The F-measure drops 0.6% when focal Loss is removed (see Table 4), which indicates that the imbalance of positive and negative sample proportion also has adverse effects on OBIs single-category data.

Table 4: Influence of Focal Loss.

	Cross Entropy Loss	Focal Loss
P	0.789	0.794
R	0.795	0.808
F	0.792	0.801

Table 5: Influence of Mixed-Precision.

Batch:32 Scale:416	Without Mixed-Precision	With Mixed-Precision
Training Memory	8.23GB	3.26GB
Training Time	14H	6H

The Influence of Mixed-Precision: We study the validity of Mixed-precision by using Nvidia apex tools [60]. Specifically, we set up the net-work and optimizer to prioritize Float16 as the default data type of tensors. Compared with the method using mixed-precision (see Table 5), F-measure was not affected at all, but the memory occupancy and training time were reduced by nearly 50%, which indicates the effectiveness of mixed-precision.

5 Conclusions

In this paper, we investigate the performance of popular object detection architecture on OBIs dataset. From the experiment, YOLOv3 performed well in all tests, with higher precision, faster speed, less overlap, and sensitivity to small targets and tolerance noise. From the perspective of studying oracle, we thought that YOLOv3 would be a good choice.

Furthermore, we propose a modified YOLOv3 method which has reached 80.1 F-measure. Although our method has made some progress in the field of OBI detection, there are still some issues needed to be further studied. For example, some OBIs have too few samples to train an accurate detector which can avoid overfitting, resulting in serious degradation with these OBIs. The study of all of this issue is left as our future work.

ACKNOWLEDGMENTS

This work is supported by the joint fund of National Natural Science Foundation of China (NSFC) and Henan Province of China under Grant U1804153, partly supported by the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) under Grant 2017PT35, and partly supported by the Program of Innovative Research Team (in Science and Technology) in University of Henan Province of China under Grant 17IRTSTHN012.

REFERENCES

[1] Ren S, He K, Girshick R and Sun J (2015). Faster r-cnn: towards real-time object detection with region proposal networks.
[2] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C Y and Berg A C (2016). Ssd: Single shot multibox detector. In: European conference on computer vision, Springer, 21–37.
[3] Redmon J and Farhadi A (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.
[4] Zhang S, Wen L, Bian X, Lei Z and Li S Z (2018). Single-shot refinement neural network for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4203–4212.
[5] Liu S, Huang D, et al. (2018). Receptive field block net for accurate and fast object detection. In Proceedings of the European Conference on Computer Vision (ECCV), 385–400.
[6] Li, F., & Zhou, X. (1996). Recognition of Jia Gu Wen based on Graph theory. Journal of Electronics, 18(suppl.), 41-47.
[7] Meng, L., & Izumi, T. (2017). A combined recognition system for oracle bone inscriptions. Int. J. Mechatron. Syst., 7(4), 235-244.
[8] Li Q, Yang Y and Wang A (2011). Recognition of inscriptions on bones or tortoise shells based on graph isomorphism. Computer Engineering and Applications, 47(8), 112-114.
[9] Lv, X., Li, M., Cai, K., Wang, X., & Tang, Y. (2010). A graphic-based method for Chinese oracle-bone classification. Journal of Beijing Information Science and Technology University, 25(Z2), 92-96.
[10] Lv X, Li M, Cai K, Wang X and Tang Y (2010). A graphic-based method for Chinese oracle-bone classification. Journal of Beijing Information Science and Technology University, 25(Z2), 92-96.
[11] Meng L (2017). Two-stage recognition for oracle bone inscriptions. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10485, 672-682.
[12] Meng L and Izumi T (2017). A combined recognition system for oracle bone inscriptions. Int. J. Mechatron. Syst., 7(4), 235-244.
[13] Meng L (2017). Recognition of oracle bone inscriptions by extracting line features on image processing. Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods, pp. 606-611.
[14] Feng G, Gu S and Yang Y (2013). Feature extraction method of Oracle-bone inscriptions based on mathematical morphology. Journal of Chinese Information Processing, 27(2), 79-85.
[15] Guo C, Wang E, Roman R, Chao H and Rui Y (2016). Building hierarchical representations for oracle character and sketch recognition. IEEE Trans. Image Process., 25(1), 104-118.
[16] Arthur D and Vassilvitskii S (2007). K-means++: The advantages of careful seeding.
[17] Lin T Y, Goyal P, Girshick R, et al. (2017). Focal Loss for Dense Object Detection. IEEE Transactions on Pattern Analysis & Machine Intelligence, PP (99), 2999-3007.
[18] Bordes J, Maher M and Sechrest M (2009). Nvidia apex: High definition physics with clothing and vegetation. In Game Developers Conference.
[19] Long S, He X and Ya C (2018). Scene text detection and recognition: The deep learning era. arXiv preprint arXiv:1811.04256.
[20] Epshtein B, Ofek E and Wexler Y (2010). Detecting text in natural scenes with stroke width transform. In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2963–2970.
[21] Huang W, Lin Z, Yang J and Wang J (2013). Text localization in natural images using stroke feature transform and text covariance descriptors. In Proceedings of the IEEE International Conference on Computer Vision, 1241–1248.
[22] Jain A K and Yu B (1998). Automatic text location in images and video frames. Pattern recognition, 31, 2055–2076.
[23] Yao C, Bai X, Liu W, Ma Y and Tu Z (2012). Detecting texts of arbitrary orientations in natural images. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 1083–1090.
[24] Yi C and Tian Y (2011). Text string detection from natural scenes by structure-based partition and grouping. IEEE Transactions on Image Processing, 20, 2594–2605.
[25] Yin X C, Yin X, Huang K and Hao H W (2014). Robust text detection in natural scene images. IEEE transactions on pattern analysis and machine intelligence, 36, 970–983.
[26] Coates A, Carpenter B, Case C, Satheesh S, Suresh B, Wang T, Wu D J and Ng A Y (2011). Text detection and character recognition in scene images with unsuper-vised feature learning. In ICDAR, Volume 11, 440–445.
[27] Lee J J, Lee P H, Lee S W, Yuille A and Koch C (2011). Adaboost for text detection in natural scene. In 2011 International Conference on Document Analysis and Recognition, IEEE, 429–434.
[28] Wang K, Babenko B and Belongie S (2011). End-to-end scene text recognition. In 2011 International Conference on Computer Vision, IEEE, 1457–1464.
[29] Wang T, Wu D J, Coates A and Ng A Y (2012). End-to-end text recognition with convolutional neural networks. In Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), IEEE, 3304–3308.
[30] Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, Fischer I, Wojna Z, Song Y, Guadarrama S, et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE conference on computer vision and pattern recognition, 7310–7311.
[31] Uijlings J R, Van De Sande, K E, Gevers T and Smeulders A W (2013). Selective search for object recognition. International journal of computer vision, 104, 154–171.
[32] Zitnick C L and Doll´ar P (2014). Edge boxes: Locating object proposals from edges. In European conference on computer vision, Springer, 391–405.
[33] Pinheiro P O, Collobert R and Doll´ar P (2015). Learning to segment object candidates. In Advances in Neural Information Processing Systems, 1990–1998.
[34] Pinheiro P O, Lin T Y, Collobert R and Doll´ar P (2016). Learning to refine object segments. In European Conference on Computer Vision, Springer, 75–91.
[35] Girshick R, Donahue J, Darrell T and Malik J (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, 580–587.
[36] He K, Zhang X, Ren S and Sun J (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37, 1904–1916.
[37] Girshick R (2015). Fast R-CNN. In Proceedings of the IEEE international conference on computer vision, 1440–1448.
[38] Everingham M, Eslami S A, Van Gool, L, Williams C K, Winn J and Zisserman A (2015). The pascal visual object classes challenge: A retrospective. International journal of computer vision, 111(2015), 98–136.
[39] Lin T Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Doll´ar P and Zitnick C L (2014). Microsoft coco: Common objects in context. In: European conference on computer vision, Springer, 740–755.
[40] Dai J, Li Y, He K and Sun J (2016). R-fcn: Object detection via region-based fully convolutional networks. In Advances in neural information processing systems, 379–387.

[41] Lee H, Eum S and Kwon H (2017). Me r-cnn: Multi-expert r-cnn for object detection. arXiv preprint arXiv:1704.01069.
[42] Zhu Y, Zhao C, Wang J, Zhao X, Wu Y and Lu H (2017). Couplenet: Coupling global structure with local parts for object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 4126–4134.
[43] Shrivastava A, Gupta A and Girshick R (2016). Training region-based object detectors with online hard example mining. In IEEE Conference on Computer Vision Pattern Recognition.
[44] Wang X, Shrivastava A and Gupta A (2017). A fast R-CNN: Hard positive generation via adversary for object detection.
[45] Bell S, Lawrence Zitnick, C Bala K and Girshick R (2016). Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2874–2883.
[46] Gidaris S and Komodakis N (2015). Object detection via a multi-region and semantic segmentation-aware cnn model. In Proceedings of the IEEE International Conference on Computer Vision, 1134–1142.
[47] Shrivastava A and Gupta A (2016). Contextual priming and feedback for faster r-cnn. In European Conference on Computer Vision, Springer, 330–348.
[48] Zeng X, Ouyang W, Yang B, Yan J and Wang X (2016). Gated bi-directional cnn for object detection. In European Conference on Computer Vision, Springer, 354–369.
[49] Cai Z, Fan Q, Feris R S and Vasconcelos N (2016). A united multi-scale deep convolutional neural network for fast object detection. In European conference on computer vision, Springer, 354–370.
[50] Kong T, Sun F, Yao A, Liu H, Lu M and Chen Y (2017). Ron: Reverse connection with objectness prior networks for object detection.
[51] Lin T Y, Dollar P, Girshick R, He K, Hariharan B and Belongie S (2017). Feature pyramid networks for object detection. In IEEE Conference on Computer Vision Pattern Recognition.
[52] Shrivastava A, Sukthankar R, Malik J and Gupta A (2016). Beyond skip connections: Top-down modulation for object detection.
[53] Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R and Lecun Y (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. Eprint Arxiv.
[54] Redmon J, Divvala S, Girshick R and Farhadi A (2015). You only look once: United, real-time object detection.
[55] Redmon J and Farhadi A (2017). Yolo9000: Better, faster, stronger. In: IEEE Conference on Computer Vision Pattern Recognition.
[56] Fu C Y, Liu W, Ranga A, Tyagi A and Berg A C (2017). Dssd: Deconvolutional single shot detector.
[57] Shen Z, Zhuang L, Li J, Jiang Y G, Chen Y and Xue X (2017). Dsod: Learning deeply supervised object detectors from scratch.
[58] Lin T Y, Goyal P, Girshick R, et al. (2017). Focal Loss for Dense Object Detection. IEEE Transactions on Pattern Analysis & Machine Intelligence, PP (99), 2999-3007.
[59] Yu F and Koltun V (2015). Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122.
[60] Bordes J, Maher M and Sechrest M (2009). Nvidia apex: High definition physics with clothing and vegetation. In Game Developers Conference.
[61] Guo Moruo and Hu Houxuan (1982). Collection of oracle bone inscriptions. Beijing: zhonghua book company, 21.
[62] Peng Bangjiong, Xie Ji and Ma Jifan (1999). Supplement of oracle bone inscriptions.
[63] Paszke A, Gross S, Chintala S, et al. (2017). Automatic differentiation in pytorch.
[64] Yao W, Sun Z and Chen X (2019). Understanding video content: Efficient hero detection and recognition for the game “honor of kings”. arXiv preprint arXiv:1907.07854.
[65] Laroca R, Zanlorensi L A, Gonc ̧alves G R, Todt E, Schwartz W R and Menotti D (2019). An efficient and layout-independent automatic license plate recognition system based on the yolo detector. arXiv preprint arXiv:1909.01754.
[66] Simon M, Amende K, Kraus A, Honer J, Samann T, Kaulbersch H, Milz S and Michael Gross H (2019). Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 0–0.
[67] Arthur D and Vassilvitskii S (2007). K-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM sympo- sium on Discrete algorithms, 1027–1035.