IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 5, 2019, accepted May 9, 2019, date of publication May 15, 2019, date of current version June 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2916842

Broken Corn Detection Based on an Adjusted YOLO With Focal Loss

ZECHUAN LIU AND SONG WANG

Department of Automation, University of Science and Technology of China, Hefei 230027, China

Corresponding author: Song Wang (wsong@ustc.edu.cn)

This work was supported in part by the Intelligent Networked Electric Vehicle Key System Integration Development and Industrialization Project, and in part by the National Key Research and Development Program of China under Grant 2016YFD0701904.

ABSTRACT Corns may be broken during corn mechanical harvesting. The ratio of broken corns measures the quality of mechanical harvesting and should be monitored in real time. This paper presents a method of detecting both broken and non-broken corns at the conveyor belt of a corn harvester based on the YOLO. The network structure of the YOLO is adjusted here to obtain more robust features so that it can work well in the open working space of the corn harvesting. Moreover, we improve the loss function to ensure that the hard examples can catch more attention during training. As it is difficult to obtain many training data of broken corns, the simulated broken corn images are generated from the real images of corns by a simple synthetic method. The concerned corn detection network of the proposed YOLO-based method is first trained with plenty of simulated samples and then fine-tuned with the real corn images. The experiments on real corn data confirm that the proposed YOLO-based method can achieve good accuracy and fast speed on the NVIDIA TX2.

INDEX TERMS Corn detection, YOLO, focal loss.

I. INTRODUCTION

Corn is a prime type of grain in China and widely planted. So corn harvesting is of great importance. Recently, more and more machines are implemented in harvesting corns. In a corn harvesting machine, corns are usually stripped from their stalks and then move through the header to the intake conveyor belt. The header is one of the major components of a corn harvesting machine and its performance directly determines the quality of the corn harvesting. To improve the harvesting efficiency, the header may be adjusted to collect corns faster, which, however, may break corns. The ratio of broken corns is required to lie below a tolerable bound. In order to ensure the quality of corn harvesting, the ratio of broken corns should be detected in real time, which is exactly the main task of the present paper. To accomplish this task, a real-time monitoring system is designed and shown in Figure 1. That monitoring system consists of a camera and a hardware platform, and is equipped with a corn detection method to be designed. The camera captures the images of corns, including both broken and non-broken corns, before corns are sent to the peeling device.

The associate editor coordinating the review of this manuscript and approving it for publication was Donato Impedovo.

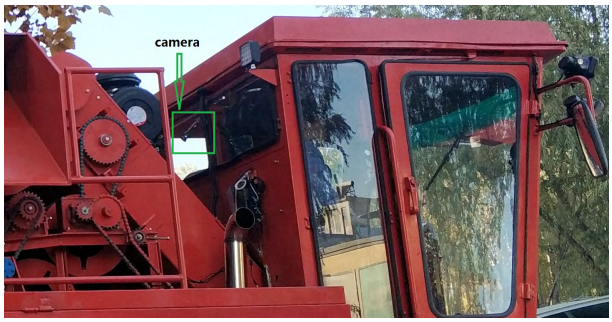


FIGURE 1. A real-time monitoring system in a corn harvesting machine. The camera is mounted on a fixed bracket above the conveyor belt to collect images.

To monitor the broken corn ratio, the detection method plays a critical role and needs to detect the numbers of broken and non-broken corns, which are used to compute the broken corn ratio. The scene of the detection method, however, is very complicated and confronts the following issues.

- As the corn harvester works outside, the sun light may incur dramatic illumination change.
- Because corns are overwhelmed by a lot of weeds and straws, occlusion becomes very serious.
- The background of corns changes very fast as the conveyor belt moves.

Due to the above issues, the traditional background subtraction method cannot be simply implemented to detect corns just based on color features. So we resort to object detectors based on Deep Neural Networks (DNN), which have made great progresses recently, can detect a large number of generic objects and are fast enough to work on mobile devices [1]. In order to train such DNN-based detector, a lot of image samples are required. Unfortunately, the number of broken corn images is limited and much less than the number required by DNN methods, which prevents from the implementation of modern DNN-based object detection methods. To compensate the lack of broken corn images, we use a simple method to synthesize corn images for training, i.e., some detected broken corns are attached to various backgrounds to produce broken corn images. Then these generated synthetic images can be used for training a corn detection network, which detects broken corns in a given image. The large number of synthetic images enables the training of the DNN-based corn detection network. Of course, the trained corn detection network will be further fine-tuned by real images with broken corns.

The aforementioned corn detection network is DNN-based and requires plenty of computations. Unfortunately, it is impossible to equip hardwares with rich computing resources, such as GPUs, on the corn harvester due to power/space limitation. Instead, embedded computing devices, such as NVIDIA TX2, may be implemented to fulfill the above corn detection task. Then the limited computing capability of these embedded devices has to be taken into account of the design of corn detection methods. Although YOLOv3 [2] can realize the real-time object detection on a powerful GPU, it still remains very challenging for leveraging this approach for real-time object detection on embedded computing devices with limited computational power and limited memory. Compared with YOLOv3, YOLOv3–tiny is faster by using a neural network with fewer convolution layers and fewer filters in those layers. Of course, the fast speed of YOLOv3-tiny is achieved at the cost of reduced detection accuracy. To further speed up the corn detection, we modify YOLOv3–tiny to reduce the computational complexity. To compensate the incurred loss of corn detection accuracy, different scales of pooling operations are introduced into the top hidden layer of the backbone network. While training detectors, the imbalance between positive and negative samples may seriously harm the detection accuracy [3]. Unfortunately, such imbalance may be unavoidable. For example, we may have much more non-broken corn images than broken corn images. To compensate that sample imbalance and pay more attention to hard training samples, a novel focal loss is proposed to reshape the standard cross entropy loss and greatly enhance the importance of broken corn images. The novel loss is dynamically scaled by using a tunable factor, which automatically decays to down-weigh the contribution of easy samples during training and mainly focus on hard samples. With the obtained corn detection network, corn images are processed to detect both broken and non-broken corns. Then the ratio of broken corns is computed and used to adjust the header of the corn harvester in real time.

The rest of this paper is organized as follows. The related work is introduced in Section II. Section III presents the overview of the proposed corn detection method. Two major contributions of the proposed method, an improved loss function to further enhance the detection accuracy and a simple method of generating broken corn images, are presented in Section IV. Experimental results are demonstrated and analyzed in Section V. Some concluding remarks are placed in Section VI.

II. RELATED WORK

Machine vision methods were implemented to measure the quality loss of agriculture products, such as corn kernel mechanical damage and mold damage [4], and efficient in extracting profile shape features of grain kernels for classification and quality inspection [5]. Some machine vision methods were developed for the detection of corn kernel breakage and demonstrated excellent performance [6]. These methods, however, cannot detect a corn on the conveyor belt in Fig. 1 or determine whether that corn is broken. To achieve our corn detection goal, we resort to image-based detection methods.

Corn detection is closely related to the existing object detection methods, some of which are compared and analyzed here. Traditional object detection methods first scan the whole image with a multi-scale sliding window, then extract some hand-crafted features, such as SIFT [7], HOG [8] or Haar-like [9] features, and finally input the extracted feature to a supported vector machine(SVM) [10] or other classifiers to distinguish a target object from all other categories. The extracted hand-crafted features are critical for detection accuracy, but may not be robust enough to describe objects in complex scenes [11].

To compensate the weakness of hand-crafted features, modern object detection methods usually implement deep neural networks to extract features. Modern methods can be divided into two categories, including the two-stage detection methods, such as Fast-RCNN [12] and Faster R-CNN [13], and the single-stage detection methods, such as YOLO [14] and SSD [15]. Under the Faster R-CNN, detection is performed in two stages. In the first stage, a region proposal network is used to process images and predict box proposals where objects may exist. In the second stage, these box proposals are used to crop features from the intermediate feature maps. Then these features are fed to the final layers to predict a class and class-specific box refinement for each proposal. The two-stage methods usually use more proposal regions which help to obtain local optimal solutions and improve detection accuracy at the cost of longer computational time.

On the contrary, the single-stage detection methods are usually faster and yield less desirable performance than two-stage methods. YOLO is one typical single-stage detection method. It divides the input image into regions and predicts

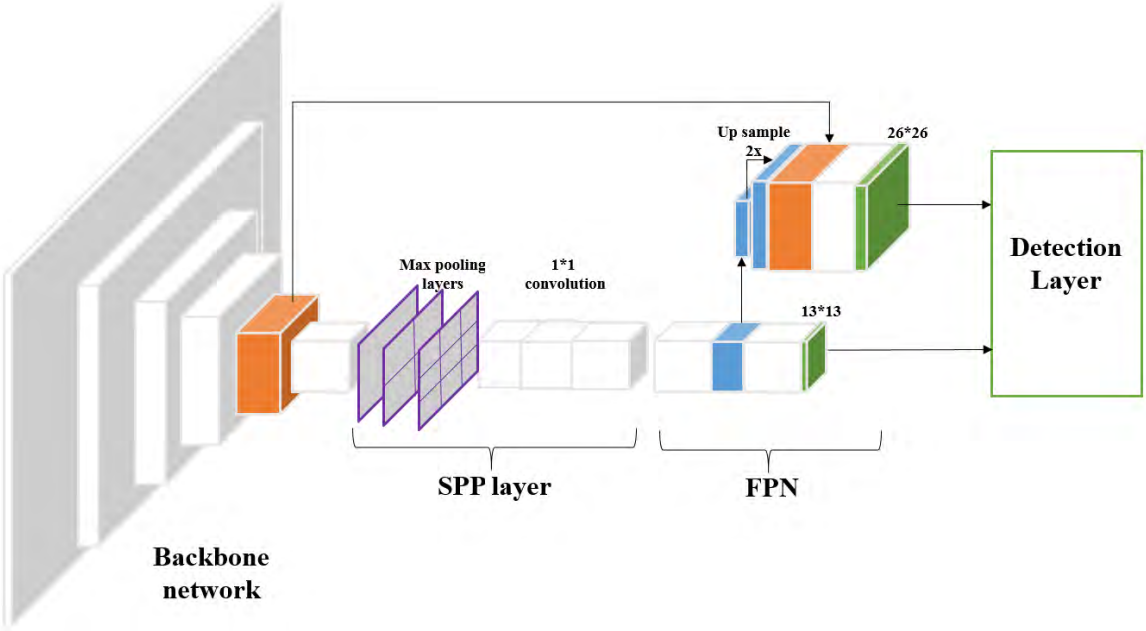


FIGURE 2. Network structure for corn detection. The backbone network, which is composed of convolutional layers and max pooling layers, is used for feature extraction. Convolutional layers use 3 × 3 filters. Max pooling layers are used for downsampling. The SPP layer is composed of three different scale max pooling layers. The feature extracted by the SPP layer is sent to the feature pyramid network (FPN). The detection layer will make prediction based on two scale feature maps which are 13 × 13 and 26 × 26.

bounding box coordinates and class probabilities for each region [14], i.e., it transforms the object detection problem into a regression problem, which can truly achieve end-to-end detection. Now YOLO has developed to its third version, YOLOv3 [2], which extracts features at different scales with a feature pyramid network and makes use of anchor boxes in the bounding box prediction. YOLOv3-tiny achieves a good tradeoff between detection accuracy and speed, and will be used in the present paper.

III. THE OVERVIEW OF THE PROPOSED CORN DETECTION METHOD

A. NETWORK STRUCTURE

To resolve the corn detection issues mentioned in Section I, the backbone network of YOLOv3–tiny is used for the feature extraction of corns, which is composed of convolutional layers and max pooling layers. In order to stabilize training, speed up the convergence and regularize the model, batch normalization is used on all convolutional layers. The final detection layer uses a linear activation function and all other layers use a leaky rectified linear activation function [2], [14]. Since the concerned backbone network is shallow, its detection speed is fast. We add several layers to improve its detection accuracy without significantly affecting its real-time performance.

One spatial pyramid pooling (SPP) layer is added on the top of the last convolutional layer of the backbone network. As one of the most successful methods in computer vision, it partitions the image into divisions from finer to coarser levels and aggregates local features at all levels [16]. This multi-level pooling is robust to the variation of object deformations and spatial layout. The above SPP layer consists of several max pooling layers. The filters of these pooling layers have the sizes of 5×5, 7×7 and 9×9, and the stride of 1. Through padding operations, all output feature maps of these max pooling layers have the same size and can be aligned. Then all these output feature maps are concatenated and fed to the next convolutional layer using 1×1 filters [17]. Since features of different spatial sizes are concatenated together, the network can utilize more spatial information through convolution operations. The features of the surroundings of the concerned object serve as the context information to improve the detection accuracy. So the 1×1 convolution filter realizes the information integration of cross-channel feature maps, and introduces new parameters and new non-linearity into the network, which help in improving the corn detection accuracy. Moreover, the 1×1 convolutional filter can reduce channel dimensions, and yield a more compact network structure [18]. The detection layer predicts bounding boxes of corns at 2 different levels. Furthermore, features are extracted at those 2 scales using a similar concept to feature pyramid networks (FPN) [19]. Among the extracted features, the high-level semantic features are first upsampled, concatenated with the feature maps of the same size in the backbone network, and processed by the convolution operations to yield the low-level feature maps. These feature maps own stronger feature expression capability and can efficiently cope with the scale change of corns. On the prediction feature maps at each level, 2 anchors are used in each cell by clustering. For the task of detecting corns, this parameter setting is sufficient to deal with corn scale changes. The network structure of corn detection is illustrated in Figure 2.

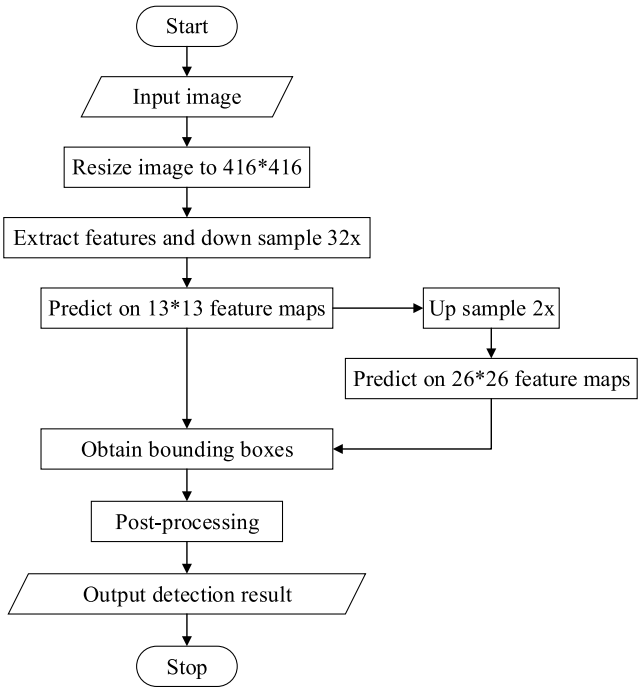


FIGURE 3. The main detection procedure.

B. THE MAJOR CORN DETECTION PROCEDURE

After training the corn detection network in Figure 2, we perform corn detection according to the procedure in Figure 3. Now we explain the main operations.

First we have to adjust the input image size. Due to the use of the full convolutional network and multi-scale training, the input image can be resized into any desired size. But the larger the image is, the longer time the network prediction takes. So the input image is resized into 416×416 pixels. In order to maintain the original aspect ratio of the input image, it is necessary to perform padding on the short side of the image.

Then the backbone network extracts features, downsamples the image by the stride of 32, finally obtains the output feature maps of the size of 13×13. Each cell of the output feature maps predicts 2 bounding boxes based on anchor boxes. The prediction is made at 2 scales, including 13×13 and 26×26, for two classes (corns and non-corn objects). 1690 bounding boxes are produced. The network also predicts an objectness score (i.e., the likelihood for the box to contain corns) and class probabilities (the probabilities for the box to contain broken corns, non-broken corns or no corns) for each bounding box.

In fact, the number of corns in an image is limited. So there are a large number of invalid boxes among the obtained bounding boxes. To eliminate duplicate detection results, post processing in Figure 4 is implemented.

C. POST PROCESSING

As the post processing in Figure 4 is of great importance, we provide its details here.

When the objectness score of one predicted bounding box is less than a certain threshold, the object in the box is quite possible the background and discarded by setting its score to 0. The above objectness threshold is adjustable. In experiments, we compared different thresholds and found that the detection accuracy of corns is acceptable when the threshold is set at 0.40.

Then the non-maximum suppression (NMS) is implemented to merge detection results that belong to the same object. NMS greedily selects detection results with high scores and deletes close-by neighbours with low scores since they are likely to cover the same object. So we first sort the objectness scores of all boxes, and select the bounding box with the highest score. Then the remaining boxes are compared with the selected bounding box. When the overlapping ratio between the selected bounding box and one of the remaining boxes is greater than a certain threshold, e.g. 0.45, that box has a large area overlapping with the selected box and should be deleted. If the class probability of one bounding box is greater than a certain threshold, e.g., 0.50, it is predicted as the corresponding class, broken corns or non-broken corns. By iteratively performing the above operation, we can finally obtain the numbers of broken and non-broken corns and their locations.

IV. MAIN CONTRIBUTIONS

A. FOCAL LOSS

The parameter training of YOLOv3 is based on the following binary cross entropy loss for each sample,

$$L(p_t) = \begin{cases} -\log(p_t), & y_t = 1 \\ -\log(1 - p_t) & y_t = 0, \end{cases} \tag{1}$$

where t is the sample index, p_t is the predicted objectness score, i.e., $p_t \in [0, 1]$ measures the predicted likelihood that the t-th sample is an expected object (corn), and y_t stands for the ground truth. Particularly, $y_t = 1$ means the t-th sample really belongs to the class of objects (corns) while $y_t = 0$ means the t-th sample is outside of the class of corns. By $\min \sum_t L(p_t)$, the parameters of YOLOv3 are trained.

The binary cross entropy loss in (1), however, confronts the following challenges, which prevents from efficiently parameter training of YOLOv3.

1) *More emphasis should be placed on hard samples:* When the t-th sample really belongs to the class of corns ($y_t = 1$), its predicted objectness score p_t should be close to 1. But if p_t is unexpectedly far from 1, the current parameters are not appropriate, and the t-th sample is a hard sample and should play a more important role than others. On the other hand, when the t-th sample does not belong to the class of corns ($y_t = 0$), p_t should be close to 0. If p_t is much larger than 0, the t-th sample is a hard sample and should also make more impact in training the parameters. If a sample is not a hard sample, it is an easy sample. Some hard and easy samples are shown in Figure 5.

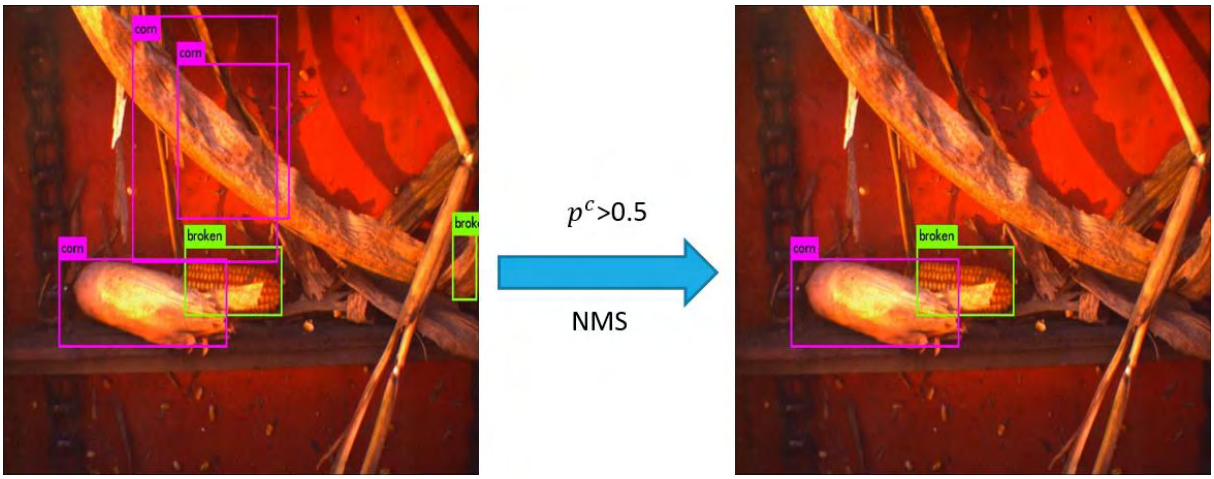


FIGURE 4. Post processing.

(a)

(b)

FIGURE 5. Some corn images: (a) Easy samples (b) hard samples.

Due to the disturbance of corn leaves and stalks, it is more challenging to detect corns in the hard samples than in the easy samples.

2) *Sample imbalance:* In a corn image, most of the area is background and the objects to be detected, i.e., (both broken and non-broken) corns, only occupy a small area. So the number of positive samples from corn regions can be much less than the number of negative samples from the background. Such serious imbalance between positive and negative samples makes it quite difficult for the positive samples to make impact in training parameters so that the detection accuracy could be poor.

To resolve the above two issues, we improve the original loss $L(p_t)$ in (1) with the following focal loss $FL(p_t)$, which is motivated by [3],

$$FL(p_t) = \begin{cases} -\alpha_t(1-p_t)^\gamma \log(p_t), & y_t = 1 \\ -\alpha_t p_t^\gamma \log(1-p_t) & y_t = 0, \end{cases} \tag{2}$$

where p_t is the predicted objectness score, γ is a tunable constant ($\gamma = 2$ in our experiments) and

$$\alpha_t = \begin{cases} \alpha, & y_t = 1 \\ 1-\alpha, & y_t = 0, \end{cases} \tag{3}$$

where α is constant in the range of $[0, 1]$ ($\alpha = 0.75$ in our experiments),

FIGURE 6. The major operations of the simple synthetic method.

In (2), the modulating factor $\alpha_t(1-p_t)^\gamma$ is introduced into the cross entropy loss of YOLOv3 and can efficiently resolve the above issues regarding hard samples and sampling imbalance. First we analyze its effects on the emphasis of hard samples.

- *When* $y_t = 1$, p_t of an easy sample is close to 1 and $(1-p_t)^\gamma$ is even closer to 0 with $\gamma > 1$. On the contrary, p_t of a hard sample is far from 1 and $(1-p_t)^\gamma$ is much larger than those of easy samples, which implies that the hard sample has much more impact on the loss function.
- *When* $y_t = 0$, p_t of an easy sample is close to 0 and p_t^γ is even closer to 0 with $\gamma > 1$. On the contrary, p_t of a hard sample is far from 0 and p_t^γ is much larger than those of easy samples. We can similarly conclude that the hard sample has much more impact on the loss function.

By choosing $\alpha > 0.5$ in (3), positive samples ($y_t = 1$) have larger α_t than negative samples ($y_t = 0$). Our experiments take $\alpha = 0.75$, which yields the α_t ratio of 3 between positive and negative samples, and can greatly enhance the impact of positive sample and well attenuate the sample imbalance.

B. DATA GENERATION

Detection methods based on convolutional neural networks need a lot of labeled samples to train parameters. But in general, a large amount of labeled real data may not be easy to obtain [20]. In this task, the number of images containing corns, particularly broken corns, is very small, which prevents the parameter training of the concerned networks. To remedy the lack of broken corn images, we propose a simple method to generate broken corn images, which is introduced below.

This method fuse a background image and a foreground image to generate a desired corn image. Hundreds of background images have been captured at the conveyor belt, which contain straws, corn husks and dust. A foreground image is a manually intercepted rectangular picture containing broken corns of different shapes, illuminations and postures. For a given foreground image, the pixels belonging to the background are first marked and removed by implementing Gaussian filtering, binarization, closing operations, etc. Then a certain area of the background image is randomly selected and replaced with a processed foreground image to generate a simulated image. The synthetic image is further processed through motion blurring, down sampling, saturation transformation and adding Gaussian noise. The operations of this simple synthetic method is shown in Figure 6 and some simulated images are shown in Figure 7. It is true that simulated

FIGURE 7. Simulated images by the simple synthetic method.

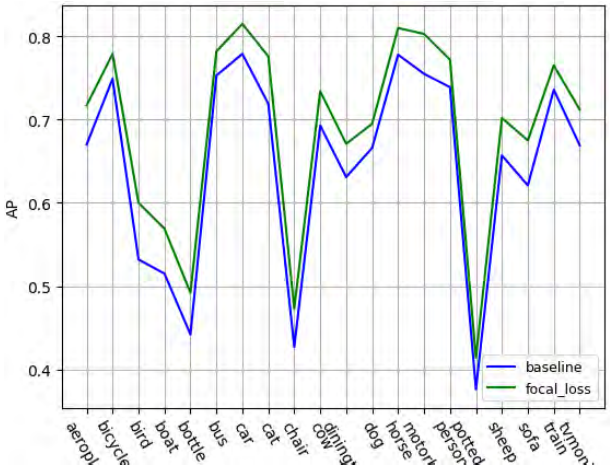


FIGURE 8. Average precision on VOC *val*2007 using different loss function.

broken corn images still look a little different from the real ones. For example, the position and boundaries of corns in these simulated image is unusual compared to the ones in real images. Fortunately, these simulated broken corn images are still good for pre-training the corn detection network in Figure 2. That pre-trained network will be further fine-tuned by real corn images. As shown in experiments, such pre-training based on simulated broken corn images does help improve the corn detection accuracy.

V. EXPERIMENTAL RESULTS AND ANALYSIS

A. VERIFY THE EFFECTIVENESS OF THE FOCAL LOSS THROUGH THE PASCAL VOC DATASET

In Section IV-A, focal loss is introduced. In order to quantitatively verify its effectiveness, we compare object detection with the focal loss and the one with the conventional binary cross entropy loss on Pascal VOC 2007 and VOC 2012 dataset [21]. We train the object detection networks with the set of *train*2007 + *trainval*2012, and evaluate them on *val*2007. Here *train*, *val* and *trainval* stand for *training* data, *validation* data and *training* + *validation* data, respectively; 2007 and 2012 stand for VOC 2007 and VOC 2012, respectively. Moreover, the object detection networked trained here provide the pre-trained parameters for the corn detection network in Fig. 2.

At the training stage, no-object loss is calculated by judging whether the objectness prediction is less than a certain threshold. We use two thresholds of 0.5 and 0.7 to train the networks. The average detection precision (AP) on 10 classes of objects using the threshold of 0.5 is shown as Figure 8.

TABLE 1. Mean average precision with different loss functions on VOC *val*2007. The parameter of focal loss with $\alpha = 0.75$ and $\gamma = 2.0$.

Loss	Threshold	$mAP_{50}(\%)$	$mAP_{75}(\%)$
baseline	0.5	64.53	30.83
baseline	0.7	63.90	30.11
focal loss	0.5	**68.78**	34.31
focal loss	0.7	68.10	33.78

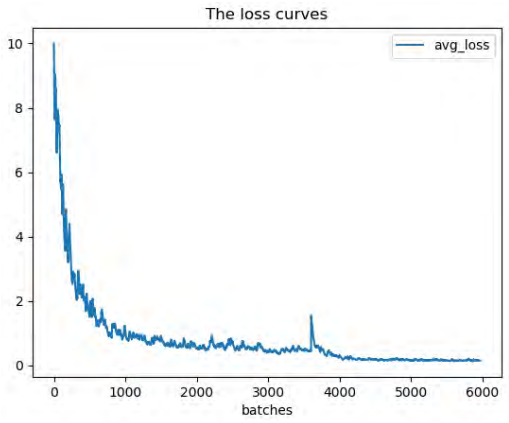


FIGURE 9. The training loss curve.

After using the focal loss training, the average precision of each class has been improved. The mean average precision (mAP), particularly mAP_{50} and mAP_{75},[1] is presented in Table 1. Under focal loss, mAP_{50} can reach 68.78%; Under the conventional binary cross entropy loss, we achieve the mAP_{50} of 64.53%. So focal loss does help to improve detection accuracy, which motivates the implementation of focal loss in our corn detection.

B. TRAINING WITH THE SIMULATED AND REAL CORN IMAGES

Here we introduce some experimental details for training on the corn dataset. We initialize the first 13 layers of the corn detection network using the weights pre-trained on the Pascal VOC dataset. Then the whole training process on corn detection is divided into two stages. The first stage is to train the corn detection network with the simulated images in Section IV-B. There are 4,000 simulated images generated by the simple synthetic method. Although the number of simulated images is large, they are created with particular cases and a little far from the real images. So the second training stage takes the real images to fine-tune the corn detection network.

When the training loss of the network is reduced to a small value at the first stage, the second stage starts by using real images for network training. Due to fine-tuning the network with real images, the corn detection accuracy is greatly improved. The training loss is shown in Figure 9. It can be seen that when the second stage starts (at 3, 600-th batch), i.e., real images are used for training, the training loss is abruptly increased and then gradually decreases. That abrupt loss increase comes from the difference between real images and simulated images.

[1] mAP_{50} stands for the mean average precision according to the criterion that a detected result is accepted only if the overlapping ratio between the detected bounding box and the true bounding box is above 50%. mAP_{75} is similarly defined by requiring the overlapping ratio to be no less than 75%.

TABLE 2. Accuracy (%) on the test set of corn images.

The number of real images	$simulate + real$	$real$
0	30.94	-
100	46.25	31.21
200	64.44	62.63
440	82.76	78.38

TABLE 3. Corn detection accuracy under different α and γ. There are 189 broken corns and 241 non-broken corns in the test set. The accuracy is defined as the total number of correct detected corns divided by the total number of corns. The following *broken* column stands for the number of correctly detected broken corns while *non-broken* for the number of detected non-broken corns.

α	γ	broken	non-broken	accuracy(%)
0.25	0	145	213	83.26
0.25	0.1	146	215	83.95
0.25	0.2	147	215	84.19
0.50	0.5	149	219	85.58
0.75	1.0	153	228	88.60
0.75	2.0	157	229	**89.77**
0.75	3.0	152	226	87.90
0.75	4.0	147	217	84.65
0.75	5.0	147	214	83.95

In order to evaluate the effects of the simulated images, we train the original YOLOv3-tiny network with two types of image combination. In the first one, simulated images and real images are used as above mentioned. In the second one, only real images are used for comparison. In order to evaluate the performance of the trained networks, they are tested with some real images which are not used in training. The test set is made up of 143 images, which contain 189 broken corns and 241 non-broken corns. We have manually labeled location coordinates and categories of the objects in these images. The test results are presented in Table 2. We can see that the simulated images do help to improve the detection accuracy.

C. PARAMETER SENSITIVITY OF FOCAL LOSS

According to (2), focal loss has two important parameters, γ and α, which aim to enhance the importance of hard samples and resolve the sample imbalance issue. Now we run some experiments to demonstrate the effects of γ and α on corn detection accuracy. For each γ, we find the best α, which yields the highest corn detection accuracy under the given γ. The corn detection results are presented in Table 3. The accuracy is calculate by dividing the total number of correctly detected broken and non-broken corns with the total number of real broken and non-broken corns. By Table 3, too small γ does not yield pleasant detection results because under small γ, the importance of hard samples is not enhanced enough. On the contrary, too large γ does not help detection either due to over-emphasizing hard samples, which may hurt the detection accuracy. We see that the best γ is 2.0, which

TABLE 4. Detection results of different methods.

Method	Threshold	Accuracy(%)	Speed(FPS)
YOLOv3	0.5	90.24	3
SSD	0.5	88.57	4.16
YOLOv3-tiny	0.5	82.76	10
Ours	0.5	89.77	10

FIGURE 10. Some detection result on the test set. We show some detection examples with classification scores higher than 0.5. The green and purple bounding boxes represent broken and non-broken corns, respectively.

may appropriately enhance the importance of hard samples and is also the best γ in [3]. Under $\gamma = 2$, the best α is 0.75, which yields the weight ratio of 3 between positive and negative samples.

In order to show the efficiency of the proposed method, we compare it with some state-of-the-art methods, including YOLOv3, YOLOv3-tiny and SSD. We train YOLOv3 and YOLOv3-tiny with the binary cross entropy loss on the set of *simulate* + *real*, and evaluate them on the test set. The backbone network of SSD is VGG16. The size of input image is 416 * 416. The detection accuracy and speed on NVIDIA TX2 are shown in Table 4. After using the focal loss to train our detection network, the detection accuracy of broken and non-broken corns is close to YOLOv3 and better than SSD [2]. Although the accuracy of the proposed method is lower than that of YOLOv3, the proposed method is much faster than YOLOv3. Note that the developed corn detection method has been on an embedded system, like the NVIDIA Jetson TX2. Our detection speed can reach up to 10*fps* (frame-per-second) on TX2 and is about 3 times faster than the standard YOLOv3 due to its tiny backbone network. Compared with YOLOv3-tiny, our corn detection network achieves the accuracy improvement of about 7% without any speed loss. The major reason of the detection performance improvement lies in the focal loss, which efficiently enhances the hard samples.

Some test results are also visually illustrated in Figure 10. We can see that the illumination may greatly change, the background is complex and there may exist serious occlusion. When there is no occlusion and corns are well separated, the detection can be very accurate. Even in the existence of illumination change, robust corn detection can still be obtained. When corns, leaves and corn husk are stacked together, too blurring boundaries and similar shapes may lead to missing detection and false alarms. To further reduce such missing detection and false alarms will be our future research.

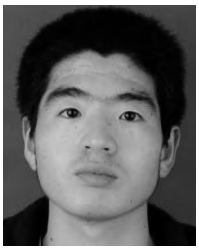
VI. CONCLUSION

This paper proposes a method to detect the broken and non-broken corns on the conveyor belt based on YOLOv3-tiny. It adjusts the network structure of YOLOv3-tiny to generate more robust features for corn detection. By introducing the focal loss, hard samples can be paid more attention to and the detection performance is improved. In order to resolve the issue of too few broken corn images, we implement a simple synthetic method to generate simulated broken corn images. With these simulated broken corn images, the corn detection network is pre-trained. Then real corn images are used to further fine-tune the corn detection network. The proposed corn detection method is implemented on NVIDIA TX2 and can achieves the speed up to 10*fps* speed, which can perform almost real-time detection.

REFERENCES

[1] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. CVPR*, vol. 4, Jul. 2017, pp. 7310–7311.
[2] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," Apr. 2018, *arXiv:1804.02767*. [Online]. Available: https://arxiv.org/abs/1804.02767
[3] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.
[4] H. F. Ng, W. F. Wilcke, R. V. Morey, and J. P. Lang, "Machine vision evaluation of corn kernel mechanical and mold damage," *Trans. ASAE*, vol. 41, no. 2, pp. 415–420, 1998.
[5] K. Liao, M. R. Paulsen, J. F. Reid, B. C. Ni, and E. Bonifacio-Maghirang, "Corn kernel breakage classification by machine vision using a neural network classifier," *Trans. ASAE*, vol. 36, no. 6, pp. 1949–1953, 1993.
[6] L. Zayas, H. Converse, and J. Steele, "Discrimination of whole from broken corn kernels with image analysis," *Trans. ASAE*, vol. 33, no. 5, pp. 1642–1646, 1990.
[7] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, vol. 1, no. 1, pp. 886–893.
[9] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1, Sep. 2002, pp. I–I.
[10] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intell. Syst. Appl.*, vol. 13, no. 4, pp. 18–28, Jul./Aug. 2008.
[11] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.

[12] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440–1448.
[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 779–788.
[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 21–37.
[16] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. Eur. Conf. Comput. Vis.* Zurich, Switzerland: Springer, 2014, pp. 346–361.
[17] M. Lin, Q. Chen, and S. Yan, "Network in network," Dec. 2013, *arXiv:1312.4400*. [Online]. Available: https://arxiv.org/abs/1312.4400
[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.
[19] T.-Y. Lin and P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 936–944.
[20] C. Wu, S. Xu, G. Song, and S. Zhang, "How many labeled license plates are needed?" in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis. (PRCV)*. Guangzhou, China: Springer, 2018, pp. 334–346.
[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.



ZECHUAN LIU received the B.E. degree from Shandong University, Jinan, China, in 2017. He is currently pursuing the M.E. degree with the University of Science and Technology of China. His research interests include computer vision, image processing, and machine learning.

SONG WANG received the B.S., M.E., and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 1997, 2003, and 2010, respectively. His research interests include image processing and machine learning.