Ball detection using Yolo and Mask R-CNN

Matija Buric
Hrvatska elektroprivreda d.d.
Rijeka, Croatia
matija.buric@hep.hr

Miran Pobar
University of Rijeka Department of Informatics
Rijeka, Croatia
mpobar@uniri.hr

Marina Ivasic-Kos
University of Rijeka
Department of Informatics
Rijeka, Croatia
marinai@uniri.hr

***Abstract*—Many computer vision applications rely on accurate and fast object detection, and in our case, ball detection serves as a prerequisite for action recognition in handball scenes. We compare the performance of two of the state-of-the-art convolutional neural network-based object detectors for the task of ball detection in non-staged, real-world conditions. The comparison is performed in terms of speed and accuracy measures on a dataset comprising custom handball footage and a sample of images obtained from the Internet. The performance of the models is compared with and without additional training with examples from our dataset.**

Keywords—action recognition, object detection, ball detection

I. Introduction

One of the fundamental tasks in computer vision is object detection, that deals with the detection of real-world objects such as people, cars or traffic signs in digital images and videos. Various machine learning approaches have been developed for the detection of specific object classes, e.g. faces [1] or pedestrians [2]. Recently, deep learning techniques based on convolutional neural networks have been successfully applied for the detection of a great number of object classes at once, without having to develop specific class-dependent features by hand. However, the choice of the object detection method still depends on the task that should be solved, as the detectors offer trade-offs in terms of their speed, accuracy and the granularity of the results of detection, with some providing bounding boxes while others provide pixel-level segmentation masks.

Here, we focus on ball detection as a part of action detection task in handball scenes, since the previous experiments show that while the state-of-the-art detectors trained on datasets of general images perform rather well with player detection, they struggle with detecting the ball in real-world contexts [3]. The detection of the ball is important for the task as it carries a lot of information relevant for interpreting the player actions [4], but it is also challenging since the ball can occupy only a small number of pixels in the image, due to its size and distance from the camera, it can move very fast, causing motion blur, and in many frames it can be completely or partially occluded by players [5] or other objects in the scene. Since balls come in simple and common round shape, they are easily confused with other objects like head, lamp, etc. This task proves to be challenging for computer vision even though people have no trouble distinguishing balls from other round objects.

We have compared the performance of Yolo [6] and Mask R-CNN [7] for ball detection before and after additional training on our dataset, in terms of detection accuracy and speed. These two methods differ in the way they denote the detected object, one aiming at being fast while detecting only the bounding box, and the other at being more precise, resulting with a segmentation mask around an object, but for ball detection, this difference is not decisive. In [3, 4] ball and player detection were performed but since predefined weights were used in models it is expected to improve the reported results with additional training specifically for ball detection.

In the next section, the tested detectors are described. Section III describes the dataset used for training and testing the detectors. The details of the experimental setup and the results are presented in Section IV, followed by a conclusion.

II. Detectors

A. YOLO

Since YOLO was first announced, few versions were released and the latest is called YOLOv3 [8]. Unfortunately, version 3 is adjusted mainly for GPU use so, therefore, in order to be consistent to previous research, YOLOv2 [6] will be applied in the experiment using the only CPU. The way YOLO works is by dividing the input into potential bounding boxes from which convolutional features are extracted. Its name – “You Only Look Once” is derived from the fact that a single-stage network architecture is used to predict class probabilities and surround them with bounding boxes without a need for a separate stage like in R-CNN family to get the region of interest proposal. In the YOLO approach [9], as shown in Fig. 1, the input image is sliced into an SxS grid and for each cell, a probability distribution of an object class is simultaneously calculated to predict a corresponding confidence score with its bounding box for each sliced cell which makes algorithm fast.

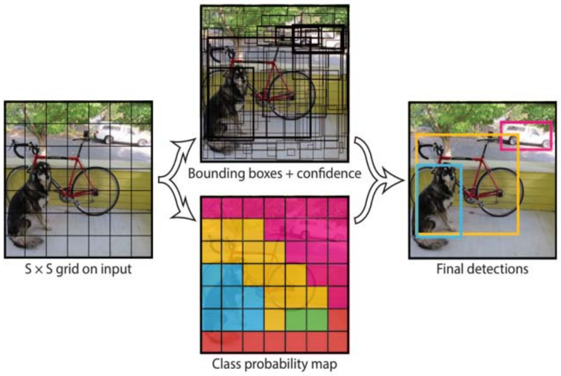

Fig. 1. YOLO divides the input image into S x S grid and simultaneously predict bounding box, confidence, and class probability to make the final decision [9]

The measurement of confidence score gives out the confidence level that bounding box surrounds an object along with the accuracy. If there is no object in the bounding box, confidence value will result in zero, but if there seems to be a

This research was fully supported by Croatian Science Foundation under the project IP-2016-06-8345 “Automatic recognition of actions and activities in multimedia content from the sports domain” (RAASS).

978-1-7281-1360-9/18/$31.00 ©2018 IEEE
DOI 10.1109/CSCI46756.2018.00068

detection of an object inside, confidence level with provided intersection-over-union score of predicted and the ground truth boxes. To avoid false positives (FP) if the object spans through several cells YOLO will designate the center cell to be the carrier of prediction for that object. The original YOLO model network architecture consists of 24 convolutional layers and additional 2 fully connected layers. Features are extracted in these convolutional layers whereas the bounding box predictions and probabilities are calculated in fully connected layers. The YOLOv2 model architecture was altered in a way where 5 max-pooling layers were added, and the number of convolution layers was reduced to 19 with filters size 3 x 3. Fully connected layers were removed altogether in order to adjust bounding box proposal to use predefined anchor boxes instead of box coordinates. A similar approach was applied in Mask R-CNN [7]. In order to determine anchor boxes in a training set of Ground Truth (GT) bounding boxes, YOLOv2 uses k-means clustering where translations of the boxes are relative to the grid cell.

Preliminary results achieved through an experiment in [3] were acquired using publicly available pre-trained models with their corresponding weights trained on the COCO dataset, with no additional training.

B. *Mask R-CNN*

Mask R-CNN comes as the successor to Faster R-CNN which could instead be used for comparison with YOLO. Since MASK R-CNN provides instance segmentation it would seem wiser to use it in a test in spite of the slight computer overhead. Instance segmentation, unlike object detection, gives more information in such a way that object doesn't get localized with a bounding box, instead, given output consists of exact pixel number and location of the desired object and each object is marked with its own color, Fig. 2.

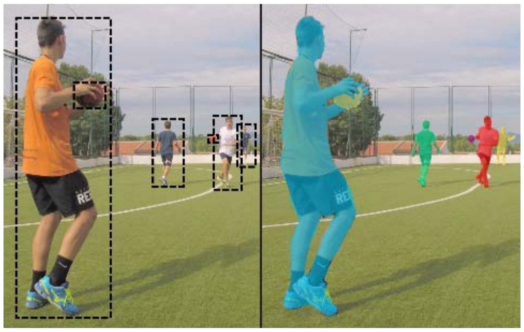

Fig. 2. Object detection vs. Instance segmentation on players and balls

Mask R-CNN framework is designed like Faster R-CNN in two stages. The first stage called Region Proposal Network (RPN) gives proposals of an area expected to contain the desired object. Second stage than classifies those areas and builds a bounding box and mask around them, Fig. 3.

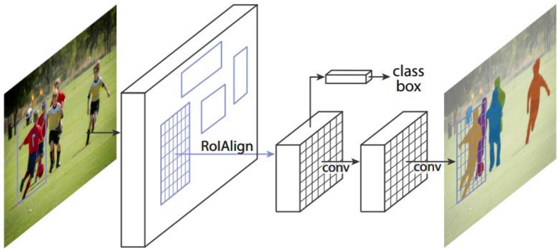

Fig. 3. The Mask R-CNN framework for instance segmentation [6]

Early layers detect low-level features such as edges and corners and as you advance through network later layers handle more complex higher-level features such as players and balls. This backbone network takes input image of size 1024x1024x3 and converts it to a map size 32x32x2048. To more efficiently handle multiple scales objects the Feature Pyramid Network (FPN) [10] is applied by combining two pyramids where one pyramid takes the high-level features from the other and then passes them down to lower layers. This way features of every level can access corresponding lower and higher-level features. The last section of the MASK R-CNN network is the mask branch that takes the positive regions selected by the classifier and generates a mask covering all the pixels that the object occupies in the image.

III. Datasets

The dataset used to train the models for ball detection consists of custom made and publicly sourced ball images.

The custom part of the dataset is extracted from the footage of handball events. The recorded footage consists of 751 videos at 1920x1080 resolution and 30 frames per second, with the total duration of 1990 seconds. The scenes were captured using stationary GoPro cameras from different angles and in different lighting conditions. From the videos, 394 training and 26 validating images were picked for training the models. The dataset shows players both during practice, where multiple balls are present and during the match, where one ball is passed between players. There were two scenarios: indoors – with camera placed at 1.5m height at different positions on the edge of the field and from the spectator's perspective at > 3m height with combination of artificial lighting and sun from the windows; outdoors – with camera placed at 1.5m height at longer line of the field on both sides during daylight. Balls were in different colors so that most often a ball has more colors.

The other part of the dataset consists of 420 training and 15 validating images of variable sizes from 174x174 up to 5184x3456 pixels sampled from the Google image search results for the term "sports ball". The objects in the dataset are essentially balls of different sizes and colors, intended for various sports, completely or partly visible, static and blurred due to a motion. The reason for adding the public dataset is to avoid overfitting on the custom dataset.

Altogether there are 855 images with 1 to 10 ball objects on each image. The dataset is manually annotated with bounding boxes for YOLO and segmentation masks around the balls for the Mask R-CNN.

IV. Experiment

In our experiment, we have first evaluated the performance for the ball detection task on our test set of images of the Mask R-CNN detector in its standard configurations and YOLOv2 in its tiny configuration, both with predefined weights for MS-COCO. We then performed additional training of both models on our training set and re-evaluated the performance. The detectors were trained and tested using CPU only on the same hardware inside same virtual environment (VMware). Programming was made in Python language under Ubuntu Linux environment. Detection speed results of a mentioned experiment were in favor of YOLO which outperformed Mask R-CNN by almost 20 times.

The input image size to the Mask R-CNN is 1024x1024, while the YOLO network uses the input size of 416x416 pixels. Since the ball object often occupies very few pixels even in full HD video, it can almost completely disappear when the images are resized to 416x416 pixels. For this reason, and to be more comparable with Mask R-CNN, the input size of the YOLO network was increased to 1088x1088. For both detectors, RGB images were used for inputs. In order to reduce the training requirements, transfer learning [11] was applied to both methods.

To avoid training the models from scratch, weights trained on COCO dataset [12] are used as the starting point. The COCO dataset, among a variety of different classes, includes the sports ball class in over 123 thousand images, therefore the features usually found in images are already fused into trained weights. Due to a custom model used, detection speed will be reevaluated in this research. To avoid great speed variance tiny-YOLO is used for training instead of full YOLO model. Training was performed in 50 epochs (5000 steps with weight backup at every hundredth step) from which the one with the smallest loss was picked. For better efficiency the number of samples that are passed through the network at one time - batch size varied. First 20 epochs of training are made with a lower value (YOLO: 2 and Mask R-CNN:1) and the second part with a higher value (up to 32). For the same reason, learning rates are also made variable by using higher learning rate at the beginning of training to more quickly descend to a local minimum and lower learning rate at the end to avoid overshooting minimum loss. Time needed for a complete cycle of training for YOLO took approximately 97 hours and 25 hours for Mask R-CNN.

Performance of sports ball detectors is measured by taking bounding box around an object, in case of Mask R-CNN bounding box around the mask of an object and comparing it with ground truth. Measures are presented in a form of precision, recall and F1 score [13]. In order to avoid clutter caused by many false positives only detections with a confidence score of 85% and more are taken into consideration. The detection will be marked as true positive if more than half of the area ball occupies is inside the bounding box. Custom and public datasets were used during testing on both methods.

The results of the detection in terms of the F1 score are shown in Fig. 4. In all cases, additional training improves the F1 score when the full dataset is considered. Examined separately, it can be seen that the results on the custom part of the dataset have considerably improved after additional training for both YOLO and Mask R-CNN, however, have actually degraded for the public part of the dataset, slightly for Mask R-CNN, and noticeably in case of YOLO.

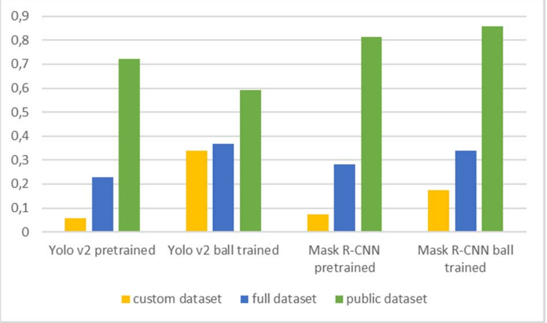

Fig. 4. F1 score on custom and publicly available dataset for both methods on ball objects

Results of precision and recall for ball detection models shown in Fig. 5. and Fig. 6. indicate higher recall values and lower precision when models are additionally trained on ball objects. Results in Fig. 6. indicate significantly higher recall values, up to 40%, on the custom dataset when Yolo v2 was used, but at the expense of dropping precision. On the same dataset, the recall increased only by 8% in the case of the Mask model, while precision decreased significantly.

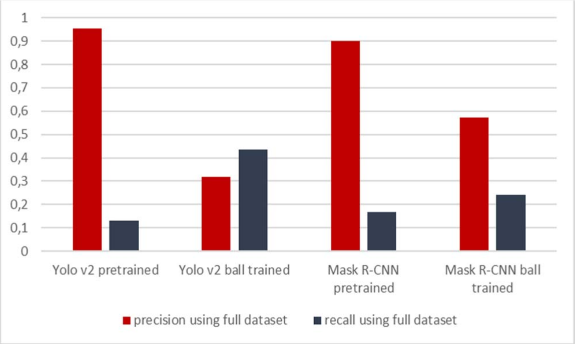

Fig. 5. Precision and recall of pre-trained and trained Yolo v2 and Mask R-CNN models for ball detection on the full dataset

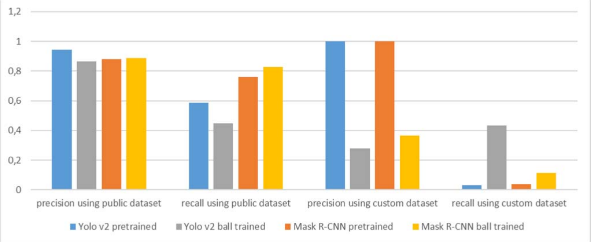

Fig. 6. Precision and recall of pre-trained and trained models for ball detection on public and custom datasets

Fig. 7. shows an example of ball detection before and after additional training. It can be seen that the detection of balls further away from the camera has improved especially in the case of YOLO. Even though Mask R-CNN detects objects more precisely than YOLO, it fails to detect ball further away to the right and both make a false positive on the backpack set against the wall.

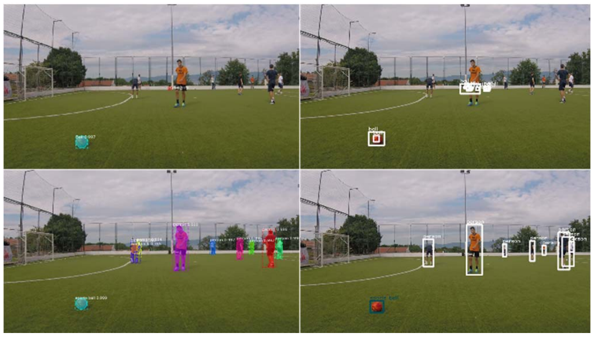

Fig. 7. Ball detection results with Mask R-CNN on the left and YOLO on the right, top showing models trained on ball object and bottom with pre-trained weights.

In the next example in Fig 8. both models failed to detect the ball close to the camera, which can easily be spotted by a human observer. Pre-trained models failed to detect ball objects altogether, while the custom trained models managed to correctly detect ball on the left side. The problem YOLO has compared to Mask R-CNN is a much higher number of false positives, but can better handle objects further away, as shown in the example in Fig 9.

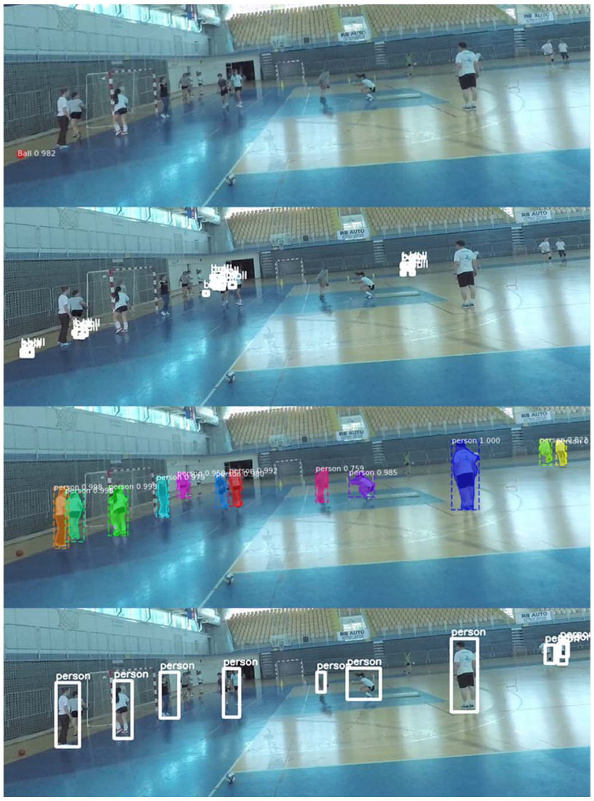

Fig. 8. Indoor detection using Mask R-CNN and YOLO with additional training (top rows), and Mask R-CNN and YOLO with pre-trained weights (bottom rows).

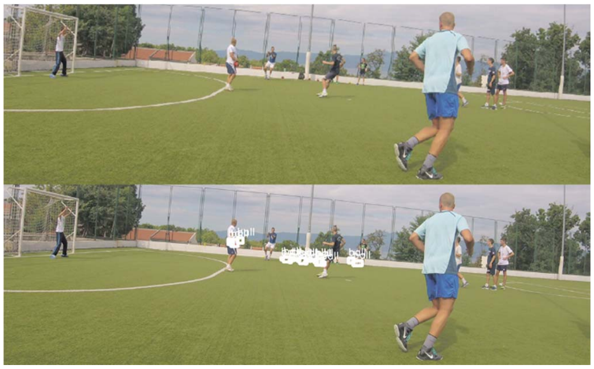

Fig. 9. Results of detection when the balls are far away from the camera. Top row: custom trained Mask R-CNN, bottom row: custom trained YOLO.

On the public part of the dataset, which likely more resembles the examples from the COCO dataset, Mask R-CNN performs much better and even handles closer objects more successfully. Fig 10. shows one of such examples.

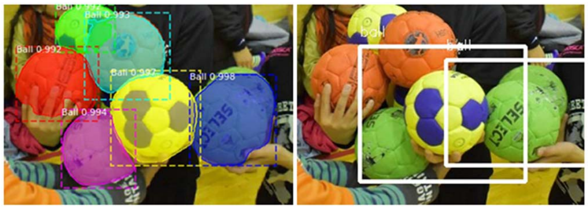

Fig. 10. Custom Mask R-CNN and YOLO on publicly available image

There are some examples on the custom dataset where detections are less accurate with models that were additionally trained with ball objects. Fig 11. shows one such example, where an object that was correctly detected as a person using the pre-trained model is partly and falsely detected as a ball with a high confidence score with the model after additional training.

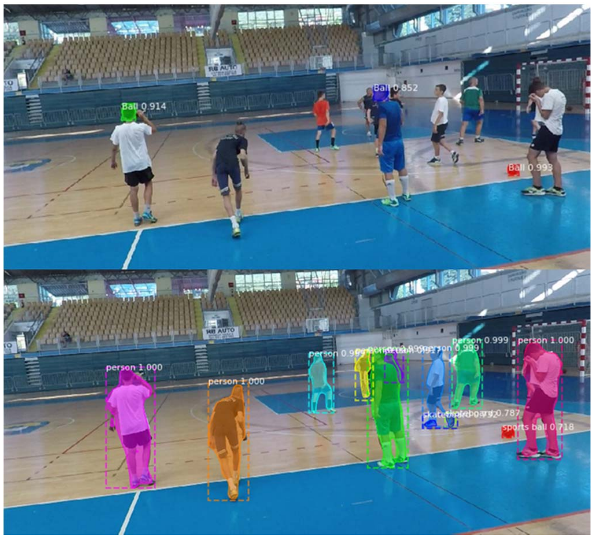

Fig. 11. An example of erroneous detection after additional training of Mask R-CNN with ball examples (top). The detections on the same image with pre-trained weights (bottom)

V. Discussion and conclusion

For the task of sports ball detection, both YOLO and Mask R-CNN with pre-trained weights struggled on our custom dataset of handball scenes. After being trained on additional examples of sports balls, sampled from the Internet and from our dataset, the performance of the models has significantly improved.

This is particularly evident in the case of YOLO, where the F1 score achieved on our dataset increased from 6% to 34%. True positive detections greatly increased at the cost of additional false positives. The reason it got better, probably, is due to a fact that it uses bounding boxes for denoting objects that in many cases are not the smallest closure of the detected ball. The increased input image size also enabled it to more easily handle smaller distant ball objects in comparison with the pre-trained model in its default configuration. By including some sort of bounding box refinement it is reasonable to believe the results could be even more improved. With higher resolution the detection speed of YOLO decreased by 43%, however, it still has a great advantage over Mask R-CNN.

Mask R-CNN didn't improve its recall as much as YOLO, however, it also had fewer false positives and thus higher precision. It should be kept in mind, that Mask R-CNN is more precise in denoting object and provides the additional segmentation information.

There were no significant differences in the performance of the detectors on outdoor and indoor scenes, except in a few cases the color of the ball seemed to confuse the detectors under artificial illumination. There are situations like in Fig 10 where detection is likely degraded due to lost information about the person object. It would seem that training models with objects similar in shape and color to a ball (like head, lamp, backpack, etc.) would further improve detection results.

Also noted is that YOLO has more trouble with occlusion (opposite to research in [3]) than Mask R-CNN when overlapping ball objects occur. This can be an issue in videos showing game practice but can be avoided in a case of real gameplay when only one ball is present in the field.

After training on additional examples of sports balls, both methods achieved usable results for integrating within an action recognition framework. Since both tested methods perform detection on individual frames, it is expected that the results can be further improved by tracking the detected balls across frames and by using the motion information, which we will consider in further work.

References

[1] P.A. Viola, M.J. Jones, "Rapid object detection using a boosted cascade of simple features", in CVPR, issue 1, 2001, pp. 511–518.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005., 2005, vol. 1, pp. 886–893.

[3] M. Buric, M. Pobar, and M. Ivasic-Kos, "Object Detection in Sports Videos," in 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2018.

[4] M. Ivašić-Kos, M. Pobar "Building a labelled dataset for recognition of handball actions using Mask R-CNN and STIPS", in EUVIP 2018 (in press)

[5] M. Pobar, M. Ivašić-Kos, "Detection of the leading player in handball scenes using Mask R-CNN and STIPS", in ICMV 2018

[6] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," arXiv preprint, 2017.

[7] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-CNN," in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988.

[8] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[10] T.-Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature Pyramid Networks for Object Detection.," in CVPR, 2017, vol. 1, no. 2, p. 4.

[11] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," Knowledge and information systems, vol. 36, no. 3, pp. 537–556, 2013.

[12] T.-Y. Lin et al., "Microsoft coco: Common objects in context," at a European conference on computer vision, 2014, pp. 740–755.

[13] Ivasic-Kos, M. Ipsic, I., Ribaric, S. A knowledge-based multi-layered image annotation system. Expert systems with applications. 42 (2015), 2015; 9539-9553.