Baltic J. Modern Computing, Vol. 9 (2021), No. 1, pp. 35–48
https://doi.org/10.22364/bjmc.2021.9.1.3

High F-score Model for Recognizing Object Visibility in Images with Occluded Objects of Interest

Bernardas CIAPAS, Povilas TREIGYS

Vilnius University, Institute of Data Science and Digital Technologies,
Akademijos str. 4, LT 08663, Vilnius, Lithuania

{bernardas.ciapas, povilas.treigys}@mif.vu.lt

Abstract. Article investigates recognition of partially occluded objects of interest. Images from retail store self checkout area often contain products that are covered by a customer's body parts, are placed inside semi-transparent plastic bags, include intensive glare, or some combination of these. In order to categorize objects of interest in images with partially occluded objects, the first step is to decide if an image contains enough information about the object of interest in order to be categorized. The most famous computer vision data sets - such as Imagenet, Canadian Institute for Advanced Research (CIFAR), Digits by National Institute of Standards and Technology (MNIST) - are made of images that contain clearly visible, distinctive objects of interest and are only labelled with binary information about object existence; reduced visibility objects are absent in the mentioned datasets. Such binary visibility labels are not suitable for solving the recognition task of object occlusion level. In this study authors categorize images into [not] containing enough information about objects of interest in order to be categorized. Authors analyze a dataset collected in a real retail store self checkout area where objects of interest are various products. The proposed method uses 6 categories of occlusion variously grouped. Authors received >0.9 F-score in best model separating images into object visible/invisible categories.

Keywords: self checkout images, occluded objects, image classification

1 Introduction

Self checkout machines were introduced in retail stores as a means to reduce need of cashiers and to shorten customer checkout time. However, self checkouts raised new problems to retailers: theft and long selection time of barcodeless products. Malignant customers use self checkouts in a variety of ways: they replace barcodes of expensive products with barcodes of cheaper ones, intentionally pick cheaper products from pick list menu. According to ECR Self Checkout report (Beck, 2018), retail stores with 50% of transactions being processed through self checkouts can expect their shrinkage losses to be 75% higher than the average rate found in Grocery retailing. This amounts to significant retailer losses in 300.000 self checkout instances worldwide as of 2020 (and growing). Benign customers suffer longer checkout times due to having to pick each barcodeless product from picklist menu that contains many similar products, has a hierarchical structure of 3-5 levels. Complex picklist menu often results in unintentional selection of wrong products and need for staff assistance. The prolonged checkout duration adds up over 1.400 weekly transactions on average per self checkout instance. Retail industry badly needs to solve these problems. Successful solutions would simplify product selection from picklist menu and raise alerts upon scanning/selecting incorrect products. In this research authors analyze a computer vision based approach to recognize products that could address each of the mentioned issues.

Self checkout process. Figure 1 shows the flow of product movement during self checkout process. A customer brings a shopping basket (left in the picture) or a trolley full of products to be purchased to the checkout area. Then she takes one product at a time from a basket/trolley and registers it in one of two ways: scans (products with barcode stickers - e.g. milk packs) or picks from a menu (barcodeless products - e.g. fruits). A scanner is usually located under the glass (green rectangle in the picture) and/or behind a glass in front of the customer (above the green rectangle in the picture). A picklist menu to select barcodeless products is displayed on a touch screen in front of a customer (above the green rectangle in the picture - not shown). Upon picking a barcodeless product from a menu, it is weighed by scales (green rectangle in the picture). Finally - after product is registered - a customer moves it to the bagging area.

Scanner/scales area (green frame in Figure 1) usually contains a single product, while other areas - shopping basket, bagging - usually contain more. Self checkouts register events: scanning of a barcode, weighing a picked from menu barcodeless product. Both at the time of scanning and weighing, a product is contained in the green frame. Thus, it is possible to take photos at the moment of scanning/weighing and label them with product ID. An average of 7 products in a shopping basket and on average 1.400 weekly transactions per self checkout result in almost 10.000 images registered per week per self checkout instance. Typically big retail stores can carry an assortment of up to 30.000 products. Additional burden is the fact that the assortment is constantly changing.

It is a much more complex task to recognize individual products in shopping basket or bagging area, where multiple products are placed. In terms of computer vision, this would be an object detection task that requires labels with product location bounding boxes. Authors refrain from detection task in basket and bagging areas in this research, although solving it has a variety of applications. This research focus is classification task in the scanner/scales area.

Convolutional neural networks like (Simonyan and Zisserman, 2015), (He et al., 2016) achieve impressive results on image classification task. Convolutional fil-

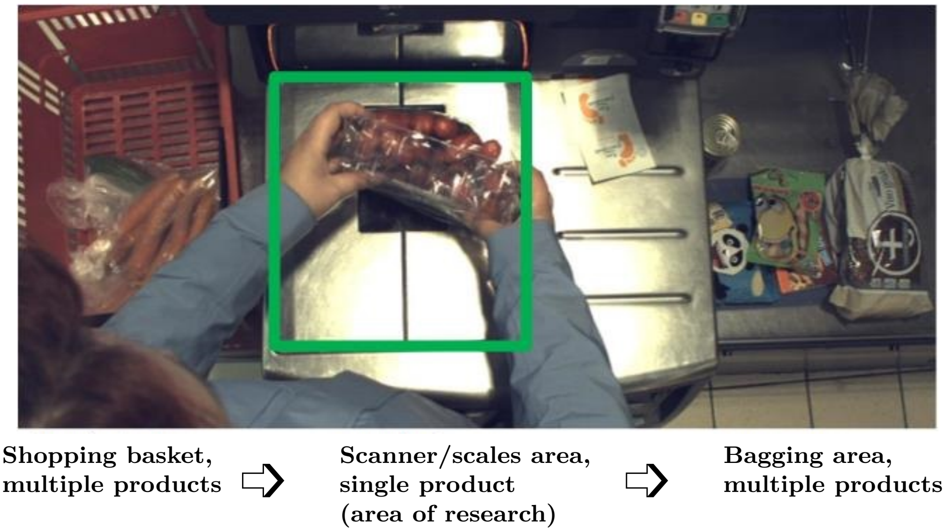
Shopping basket, multiple products
⇨
Scanner/scales area, single product (area of research)
⇨
Bagging area, multiple products

Fig. 1. Checkout flow

ters extract relevant object features as shown in (Zeiler and Fergus, 2014), then grouped by dense layers to decide on object class. However, most benchmark datasets (ImageNet, CIFAR[-10|-100], MNIST) only include images where visibility of objects of interest is binary: only images will clearly visible and distinctive objects are included.

Real life images, which need to be classified, often contain objects that are occluded to some degree. For example, most self checkout images contain products partially covered by a customer's hand or other body part; about 15% of barcodeless products are sold in plastic bags that are semi-transparent; specific locations within the scales area reflect light in a way that reduces recognizability, illumination differs during the day time in taken images, products differ in size, etc. Due to all these aspects, which are specific to self checkouts, images are likely to contain less information about the object of interest; simply applying classification techniques on images with occluded objects is likely to result in lower classification metrics. In order to obtain satisfactory classification metrics, images with occluded objects must be first categorized whether objects of interest are visible enough for classification, and only images containing well visible objects need to classified. As big retail stores carry huge assortment of products that is constantly changing (due to seasonality, change in suppliers, etc.), this implies holding individual product classification models is only practical in the cloud, but - in order to preserve network traffic - the preceding step of deciding on product visibility needs to be done in the self checkout. Moreover, self checkout images demonstrate such specifics:

- Very often products are covered by hand or other body part, i.e. product visibility is limited;

- Products are packed in plastic bags that limits product visibility as well;
- Products vary in size;
- Every self checkout camera has different illumination properties, illumination varies during the work hours and typically is non-uniform.

In this study authors aim to separate images with more occluded objects from images with less occluded objects while realizing that thresholds of separation could be multiple. Authors measure F-score as the main qualitative criteria of competing models.

2 Literature review

Recent advances in covered object recognition use a see-through terahertz beam such as (Wang et al., 2019) and analyze reflection signal amplitude and phase differences in materials. Such terahertz cameras are far from ubiquitous and will hardly ever be, and our method uses a more widespread Red-Green-Blue (RGB) image features.

Some publicly known datasets such as Imagenet (Deng et al., 2009), Pascal Visual Object Classes (VOC - (Everingham et al., 2010)) use rectangular bounding boxes as ground truth to mark object location and size. Others use even more precise object shape markings: Caltech 101 (Li Fei-Fei et al., 2006), LabelMe (Russell et al., 2008) use closed boundaries and Microsoft Research (MSRC - (Ali and Zafar, 2018)) uses pixel level segmentation. Each of the above object marking ways - rectangular bounding boxes, closed boundary shapes, and pixel level segments - are costly to label in new datasets. However, due to nature of some domains object location is bounded by a small area (such as products at the time of weighing at self checkouts), and it is only relevant to predict object class. Our method only requires class labels for images, thus making it less costly to label a new domain-specific dataset. Performance comparison with methods using location specific labels cannot be made due to different label nature.

Entropy is widely used in signal pre-processing for automatic label generation: (Liutvinavičienė and Kurasova, 2018) measure entropy between audio frames in order to extract time sequences belonging to the same syllable; (Nežerka and Trejbal, 2019) use entropy to segment images. In this research authors settled for manual image labelling, thus making it possible to formulate the task at hand - deciding if an image contains a visible enough object of interest - as a classification task.

To extract features from images authors train convolutional filters that are class-agnostic, but sensitive to object's existence. Very similar concept - class-agnostic convolutional filters on object-containing windows - was used in (Singh et al., 2018), but authors train on full images rather than object-containing crops (due to this dataset annotation nature). These methods generate region proposals, then extracts features from them: (Russell et al., 2006) extracts visual words from pixel level segments, then compares to those of known object bounding boxes; (Alexe et al., 2012) finds closed boundary shapes. Both of the above methods imply having learnt features from a dataset annotated with object locations, which didn't exist in the dataset used in this research.

Most methods use datasets where object location is defined - (Singh et al., 2018), (Cheng et al., 2019) - use intersection-over-union (IoU) to measure correctness of object localization. Since in this research authors didn't use dataset with annotations of object location, class labels (Is/Isn't an object) were used in measuring correctness.

To evaluate models authors used F-score. F-score is widely used in information retrieval, such as search, document classification, or query performance evaluation. (Jeyabharathi and Suruliandi, 2013) get 0.65 F-score measuring class match between searched vs. retrieved images in content based image retrieval. (Zhang et al., 2018) use F-score to classify search query difficulty and receive values up to 0.665. (Mowafy et al., 2018) classify textual documents into pre-defined classes and receive F-score value up to 0.92.

To optimize the neural network parameters, authors used cross entropy loss that is widely applied as a loss function in classification tasks since the beginning of artificial neural networks (Long et al., 2016) and (Krizhevsky et al., 2012). As opposed to cross entropy, many researchers use entropy to create unsupervised models: (Yin et al., 2017) attempts to maximize entropy among different image background/foreground pixels; (Kodors, 2019) and (Quinlan, 1986) try to reduce entropy when selecting next features in forming decision tree nodes; (Rikters, 2019) use entropy of output by competing translation systems in order evaluate translation quality. In this research authors decided to apply supervised training that invalidates usage of entropy as an evaluation metric.

3 Research setup

3.1 Description of dataset

Data collection and preparation. Images were collected from cameras placed over 4 distinct self checkout machines in a food retail store. Images were taken at self checkout events of scanning a product's barcode (for products having barcodes) and choosing a product from a picklist menu (for barcodeless products). The area of interest (Figure 1) crop size was 360x360 pixels.

Authors considered applying background removal techniques to automatically label images with object visibility labels, but such techniques would have treated customer body parts as foreground objects. Due to high variety of products, authors disqualified image segmentation for automatic labelling.

In order to apply classification techniques to the task of determining product visibility, authors labelled the images with visibility-level labels manually. Due to uncertainty of what portion of product needs to be visible in order to categorize images into product categories later, authors decided to use multiple ordinal labels, which can later be assigned to either Visible or Invisible by using a threshold. Labelling images into a bigger number of product visibility quantiles would have given more flexibility when splitting data into Visible/Invisible categories. However, considering a human labeller would make more mistakes if more quantiles were used, authors decided to limit the number of visibility quantiles to four. Images were randomly selected for labelling from the entire set with no pre-selection criteria. All selected images were labelled by a single human labeller.

Images with products packed in plastic bags showed very different features from images with unpacked products in early analysis: plastic bags are easily recognizable, but products inside the bags - not necessarily so. Due to some images with plastic bags having light reflection that makes product unrecognizable, authors decided to split images with plastic bags into classes Bag (recognizable products in plastic bags) and BagR (not recognizable to humans).

Finally, authors have labelled the images into 6 exclusive classes by applying these rules:

- By product visibility quartile (classes Q1–Q4) - for products not in bags
- Products in bags (class Bag) - when product can be recognized by a human
- Products in bags with reflection (class BagR) that makes a product unrecognizable

Samples of each data class are displayed in Table 1.

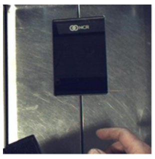	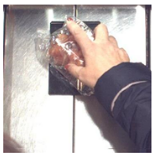	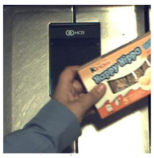	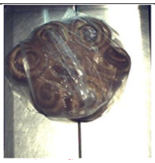	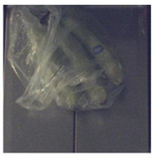	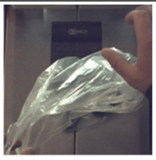
Q1 **32%**	Q2 22%	Q3 15%	Q4 21%	Bag 7.3%	BagR **2.6%**

Table 1. Each class samples and class ratios

Due to uncertainty of how many images needed to be labelled in order to create models that generalize, authors chose to label a similar number of samples per class as Imagenet dataset (~1000), where classification task was solved with high accuracy. The entire labelled dataset consists of ~6000 images.

As shown in Table 1, classes after labelling turned out to be unbalanced. The balancing was accomplished for train and validation sets by oversampling and then augmenting underrepresented class images. Only the biggest class - Q1 - was not oversampled. The test dataset was left intact since it contains a real world representation of image distribution, therefore real world classification metrics can be measured against it. The following augmentation parameters were used to balance the sets: rotation (up to 10 degrees), shifting (up to 32 pixels), zoom (up to 10%), and horizontal flip. Augmentation parameters were chosen small enough so that augmented images still mimic real photos taken by the checkout camera. Experiments on the augmentation with different augmentation parameter values results are presented in Results section. Finally, authors performed stratified split 64%, 16%, 20% into Train, Validation, Test sets. Train set was used to train classification models. Authors used validation set to tune model hyperparameters and stop model training early. Test set was used to evaluate models.

Bigger part of the image area usually contains background. In order to reduce the effect of neural networks learning background (instead of foreground) features, authors experimented with removing static background using (Zivkovic and Van Der Heijden, 2006) prior to training. In order to eliminate small foreground patches and fill small foreground mask gaps within products, authors applied morphological opening/closing on background masks.

As described in Introduction section, almost all self checkout camera images show non uniform illumination effect. In order to reduce variance in image illumination intensity, authors applied the following pre-processing techniques (one at a time) on train, validation and test sets; then trained and evaluated classification models of each (see Results section):

- Subtracted RGB mean of the self checkout instance;
- Subtracted mean "V" channel in Hue-Saturation-Value (HSV) color space of the self checkout instance;
- Subtracted mean "L" channel in Hue-Lightness-Saturation (HLS) color space of the self checkout instance;
- Applied contrast limited adaptive histogram equalization (Zuiderveld, 1994) on HSV "V" channel.

3.2 Neural network architecture

Authors used classical convolutional neural network architecture (Krizhevsky et al., 2012) by varying number of convolutional and fully connected layers. At first authors focused on reducing bias while leaving reducing variance for later: starting with one layer of each type, authors added layers until training accuracy saturated (validation accuracy not considered). Last dense layer contained a softmax activation function, all others contained ReLu. Authors used convolutional filter size 3x3; experiments of filter size 5x5, 7x7 were also performed. Network input was chosen to be 256x256, which is the nearest power of 2 smaller than the original image size. Every next convolutional layer was twice reduced in height and width using maxpooling and had 2 times number of convolutional filters (therefore, carried 1/2 of the features of previous layer). Next, authors focused on reducing variance and improving validation accuracy. Experiments were performed using batch normalization (Ioffe and Szegedy, 2015), dropout (Srivastava et al., 2014), and L2 regularizations. Authors started adding batch normalization after layers that showed sparse outputs, but then continued on using it after all the other layers - both convolutional and dense. In addition to batch normalization, experiments were performed by adding dropout after dense (except last) layers. In addition to batch normalization and dropout layers, L2 regularization was applied and tested after various dense layers. Authors optimized categorical cross-entropy loss function using Adam (Kingma and Ba,

2014) optimizer to train the models. At the end of each epoch of training, model was evaluated using validation set. Authors early stopped training models after validation accuracy did not improve for the last 20 epochs, then reverted parameters to the best epoch's. Trained models were additionally trained by halving the learning rate. Final model was chosen by the best classification accuracy obtained on validation set. All metrics reported in Results section were measured on the test set.

In order to increase variance in the dataset, authors used dynamic augmentation while training on training and validation sets. Authors experimented with the same augmentation parameter ranges as described in **Balance classes** step. In order to pick optimal values for augmentation parameters, they were doubled, tripled, halved, one was left out, augmentation was left out for validation set.

Finally, the experiments were run to investigate different neural network setup and find the best separating threshold between visible vs. invisible objects of interest in images. Authors assigned data labels in all possible ways into [Visible; Invisible] categories as shown in Figure 2 with the following restrictions:

- Q1 always Invisible;
- Q4 always Visible;
- Q3 can only be Invisible if Q2 is not Visible;
- Q2 can only be Visible if Q3 is not Invisible;
- Bag can only be Invisible if BagR is not Visible;
- BagR can only be Visible if Bag is not Invisible.

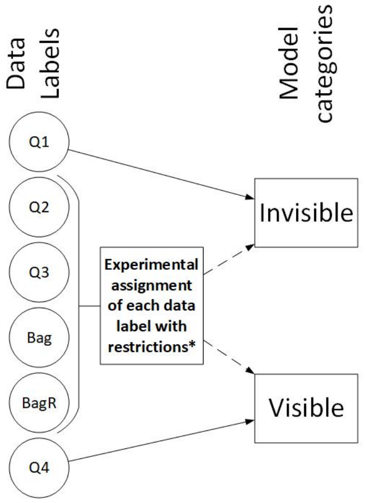
Data Labels
Model categories
Q1
Q2
Q3
Bag
BagR
Q4
Experimental assignment of each data label with restrictions*
Invisible
Visible

Fig. 2. Data labels grouping strategy

In all experiments authors used the same number of samples: undersampled data when model category [Visible; Invisible] contained more than a single label [Q1-Q4,Bag,BagR].
Neural networks were made using Keras 2.2.4-tf and Tensorflow 1.15.0. All experiments were performed on a PC with a single GPU Nvidia GeForce GTX 1070.

4 Results

Authors performed a number of experiments within the pipeline by changing certain hyperparameters and measuring the effect on validation accuracy, which led to choosing the optimal pipeline hyperparameters. The pipeline is shown in Figure 3. The effect of pipeline hyper-parameter tuning experiments is described below.

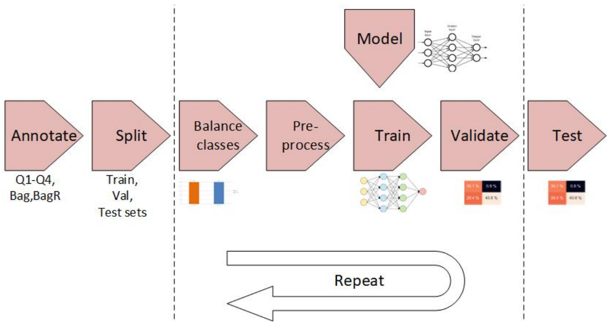

Fig. 3. Pipeline of experimentation

Balance classes step includes oversampling under-represented classes and augmenting the oversampled images. Best results were obtained using these augmentation parameters: rotation (random up to 10 degrees), shifting (random up to 32 pixels), zoom (random up to 10%), and horizontal flip (random 50% probability). Experiments of eliminating any one augmentation parameter or reducing augmentation range by half led to decline in validation accuracy of -3.9% – -1.7%.
Pre-process step includes cropping the scanner/scales area and applying CLAHE (Contrast Limited Adaptive Histogram Equalization - (Zuiderveld, 1994)) on HSV "V" channel. Model trained on images without applying CLAHE showed a negative impact of -1.4% on validation accuracy. Experiments of removing static background by applying (Zivkovic and Van Der Heijden, 2006) and further applying morphological opening/closing on mask prior to training negatively impacted validation accuracy (-3.6% and -5.4% respectively) and were excluded from the final pipeline.

Model resulted in a deep neural network architecture shown in Figure 4. Experiments with less convolutional and dense layers showed significant bias (training error); more layers of either kind did not further decrease bias. Authors experimented with larger convolutional filters (5x5, 7x7) and observed decline in validation accuracy of -3.0%. Finally, convolutional filters were all chosen to be of size 3x3. Upon adding batch normalization after various layers, authors observed generally increased validation accuracy of -0.6% - +2.2%. Final model includes batch normalization layers after each convolutional and dense layer. Experiments of adding dropout regularization after dense layers (except last) showed significant improvement on validation accuracy of +2.2% - +2.5%. The final model contains dropout after each dense layer (except last). In addition to dropout, trying L2 regularization did not help, and L2 was excluded from the final model. The final model architecture is presented in Figure 4.

Train. Authors dynamically augmented training data in each epoch. The final dynamic augmentation parameters were: rotation (random up to 10 degrees), shifting (random up to 32 pixels), zoom (random up to 10%), and horizontal flip (random 50% probability). Experiments of reducing dynamic augmentation range by half showed decline in validation accuracy of -3.4%.

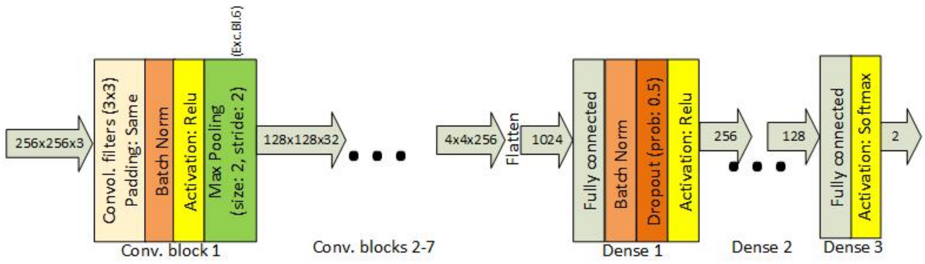

Fig. 4. Final model architecture

Validation. Model was validated after training each epoch in order to early

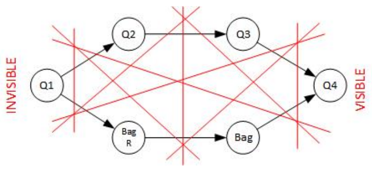

Fig. 5. Product visibility directed graph. Red lines show potential visibility thresholds

stop training and observe validation accuracy dynamics. Dynamically augmenting validation set in each epoch showed no impact on validation accuracy.

Labels in category		**F-score**	**Cross-entropy**	**Accuracy**	**Precision**	**Recall**
Visible	**Invisible**					
Q2, Q3, Q4, Bag, BagR	Q1	**0.906**	0.445	0.874	0.931	0.883
Q2, Q3, Q4, Bag	Q1, BagR	0.895	0.456	0.86	0.897	0.892
Q2, Q3, Q4	Q1, Bag, BagR	0.854	0.567	0.826	0.839	0.869
Q3, Q4, Bag, BagR	Q1, Q2	0.793	0.659	0.78	0.707	0.903
Q3, Q4, Bag	Q1, Q2, BagR	0.781	0.661	0.794	0.732	0.837
Q3, Q4	Q1, Q2, Bag, BagR	0.723	0.691	0.752	0.606	0.895
Q4, Bag, BagR	Q1, Q2, Q3	0.667	0.692	0.762	0.581	0.784
Q4, Bag	Q1, Q2, Q3, BagR	0.661	0.7	0.782	0.581	0.766
Q4	Q1, Q2, Q3, Bag, BagR	0.565	0.715	0.757	0.437	0.8

Table 2. Highest F-score models for each grouping

Evaluating the results. Quality of models was evaluated in terms of how well it separates images into [Visible; Invisible], considering that boundaries between Visible and Invisible can be multiple: a perfect model would split 100% of each label [Q1-Q4,Bag,BagR] into one category [Visible; Invisible] by using any threshold shown in red in graph 5 (product visibility increases in the direction of arrows).

Authors used F-score as the main metric to evaluate the models. F-score is a harmonic mean of precision and recall. F-score measures classifier quality more appropriately when classes are unbalanced (such as (Dal Pozzolo et al., 2015) or data in this research) than the most popular classification metrics: accuracy, precision, recall (sensitivity), specificity. On the other hand, F-score measure is comparable to accuracy, etc. when classes are balanced. Variation of F-score - F_β that gives different weights to precision vs. recall - is useful when cost of different error types (false positive vs. false negative) differ (not in scope if this research). Cross entropy, although relevant to measure classifier quality for unbalanced classes, gives higher weights to high-confidence mis-predictions, but in this research authors treat both high and low confidence mis-predictions the same.

In table 2 authors present F-score for the best threshold, as well as all the other thresholds. Next to F-score, less relevant metrics - accuracy, precision, recall, cross-entropy is presented.

In Figure 6 authors present the confusion matrix of the best F-score achieving model (Q1 vs. the rest). Although false negatives (8.1%) exceed false positives (4.5%), but unbalanced (real world) test set classes (32% Invisible) makes it likely.

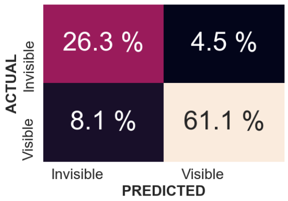

Fig. 6. Best model's (Q1 vs. the rest) confusion matrix

5 Conclusion

Authors investigated occluded object detection problem from the self checkout images and showed the best classification results by using a deep neural network based classifier of 7 convolutional and 3 dense layers. It is notable that performance degrades by removing any convolutional or deep layer, and no longer improves by adding layers. Regularization techniques were used and showed improvement in generalization of the model by adding them after convolutional (batch normalization) and dense (batch normalization and dropout) layers. As the positive impact of these aspects of pre-processing on validation accuracy: augmenting oversampled images improved by 3.4%; reducing image illumination differences using CLAHE improved by 1.4%.

The best separation of self checkout images is achieved between least visible objects of interest and the rest. Authors achieved 0.906 F-score categorizing self checkout images into less than 25% visibility of product of interest in comparison to the rest. However, according to descriptive statistics for further investigation of classification task it is worth investigating the models that achieve F-score values not less than 0.895 because both of the F-scores fall above the value of 3rd quartile. In addition to that both models similar values of precision and recall suggest no bias towards predicting either category.

References

Alexe, B., Deselaers, T., Ferrari, V. (2012). Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11), 2189–2202.

Ali, N., Zafar, B. (2018). MSRC-v2 image dataset, available at https://figshare.com/articles/dataset/MSRC-v2_image_dataset/6075788/2.

Beck, A. (2018). Self-Checkout in Retail: Measuring the Loss, available at https://www.researchgate.net/publication/330214157.

Cheng, M.-M., Liu, Y., Lin, W.-Y., Zhang, Z., Rosin, P. L., Torr, P. H. S. (2019). BING: Binarized normed gradients for objectness estimation at 300fps. *Computational Visual Media*, 5(1), 3–20.

Dal Pozzolo, A., Caelen, O., Johnson, R. A., Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2), 303–338.

He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, IEEE, pp. 770–778.

Ioffe, S., Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Bach, F. R., Blei, D. M., (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, JMLR.org, pp. 448–456.

Jeyabharathi, D., Suruliandi, A. (2013). Performance analysis of feature extraction and classification techniques in CBIR. In *2013 International Conference on Circuits, Power and Computing Technologies*, IEEE, pp. 1211–1214.

Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization, available at https://arxiv.org/abs/1412.6980.

Kodors, S. (2019). Detection of Man-Made Constructions using LiDAR Data and Decision Trees. *Baltic Journal of Modern Computing*, 7(2), 255–270.

Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F., Burges, C. J. C., Bottou, L., Weinberger, K. Q., (eds.), *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., pp. 1097–1105.

Li Fei-Fei, Fergus, R., Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4), 594–611.

Liutvinavičienė, J., Kurasova, O. (2018). Multi-level Massive Data Visualization: Methodology and Use Cases. *Baltic Journal of Modern Computing*, 6(4), 321–334.

Long, M., Zhu, H., Wang, J., Jordan, M. I. (2016). Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pp. 136–144.

Mowafy, M., Rezk, A., El-Bakry, H. (2018). An efficient classification model for unstructured text document. *American Journal of Computer Science and Information Technology*, 6(1), 16.

Nežerka, V., Trejbal, J. (2019). Assessment of aggregate-bitumen coverage using entropy-based image segmentation. *Road Materials and Pavement Design*, 21(8), 1–12.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81–106.

Rikters, M. (2019). Hybrid Machine Translation by Combining Output from Multiple Machine Translation Systems. *Baltic Journal of Modern Computing*, 7(3), 301–341.

Russell, B. C., Freeman, W. T., Efros, A. A., Sivic, J., Zisserman, A. (2006). Using Multiple Segmentations to Discover Objects and their Extent in Image Collections. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pp. 1605–1614.

Russell, B. C., Torralba, A., Murphy, K. P., Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3), 157–173.

Simonyan, K., Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. In Bengio, Y., LeCun, Y., (eds.), *3rd International Conference on Learning Representations*, Morgan Kaufmann Publishers Inc., pp. 1–14.

Singh, B., Li, H., Sharma, A., Davis, L. S. (2018). R-FCN-3000 at 30fps: Decoupling Detection and Classification. In Aneja, J., Deshpande, A., Schwing, A. G., (eds.), *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 1082–1090.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56), 1929–1958.

Wang, D., Zhao, Y., Rong, L., Wan, M., Shi, X., Wang, Y., Sheridan, J. T. (2019). Expanding the field-of-view and profile measurement of covered objects in continuous-wave terahertz reflective digital holography. *Optical Engineering*, 58(2), 1–7.

Yin, S., Qian, Y., Gong, M. (2017). Unsupervised hierarchical image segmentation through fuzzy entropy maximization. *Pattern Recognition*, 68, 245–259.

Zeiler, M. D., Fergus, R. (2014). Visualizing and understanding convolutional networks. In Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., (eds.), *European conference on computer vision*, Springer, pp. 818–833.

Zhang, Z., Chen, J., Wu, S. (2018). Query performance prediction and classification for information search systems. In Cai Y., Ishikawa Y., X. J., (ed.), *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, Springer, pp. 277–285.

Zivkovic, Z., Van Der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7), 773–780.

Zuiderveld, K. (1994). Contrast Limited Adaptive Histogram Equalization. In Heckbert, P. S., (ed.), *Graphics Gems*, Academic Press, pp. 474–485.

Received November 27, 2020 , revised February 10, 2021, accepted February 15, 2021