# Coarse Annotation Refinement for Segmentation of Dot-Matrix Batchcodes

Ning Jia, Christopher J. Holder, Stephen Bonner, Boguslaw Obara*
*Department of Computer Science*
*Durham University*
Durham, UK
ning.jia@durham.ac.uk, c.j.holder@durham.ac.uk, s.a.r.bonner@durham.ac.uk, boguslaw.obara@durham.ac.uk

***Abstract*—Deep Convolutional Neural Networks (CNN) have been extensively applied in various computer vision tasks. Although such approaches have demonstrated exceptionally high performance in various open challenges, adapting them to more specialised tasks can be non-trivial. In this paper we discuss our design and implementation of a batchcode detection system capable of accurate segmentation of batchcode regions within images of consumer products. A batchcode is a unique identifier printed on the packaging of many products that encodes useful information such as date and location of manufacture. Detection of batchcodes in images of products is a useful step in many processes, including quality control, supply chain tracking and counterfeit detection. Beginning with a unique dataset of product images and a set of crowdsourced coarse annotations that roughly correspond to the locations of batchcodes, we demonstrate that such annotations are insufficient for training a reliable model, and subsequently describe a novel label refinement process, which we call the Maximally Stable Global Region (MSGR) method, that we use to generate accurate ground-truth data suitable for training a robust neural network. We also show that detection accuracy can be further improved by applying MSGR to the output of the neural network. We evaluate our approach using a manually labelled test dataset of images of shampoo bottles, and demonstrate the efficacy of the proposed method for accurate real-time batchcode detection.**

***Index Terms*—Batchcode detection, Convolutional Neural Network, Semantic Segmentation, data labelling**

## I. Introduction

Many consumer goods are identified via a series of markings printed by the manufacturer when the product is on the production line. Markings such as the Universal Product Code (UPC), more commonly known as the Barcode, are used by retailers and manufactures to identify and track products throughout the retail process. In addition to the UPC, manufactures often print further codes on products that contain information relevant to the manufacturing process. These additional codes, which we will refer to as batchcodes throughout the rest of this paper, can contain information such as the specific facility where the product was manufactured, the date of the manufacture and other product related meta-data. Unlike UPC or Quick Response (QR) codes, batchcodes are generally not designed to be machine readable, and can vary greatly in shape, size, orientation and location on a given product. Furthermore, due to batchcodes often being printed using a series of small dots, automatic detection can be a challenging task.

* Corresponding Author

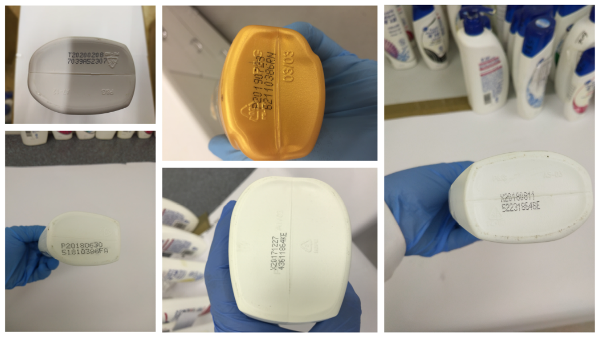
Fig. 1. Sample images of products with batchcodes. Apart from illumination and scale variabilities, there might be other code-like textures for example other text in the scene that affect detection accuracy.

Fig. 1 presents five sample images that show batchcodes printed on shampoo bottles. These batchcodes, which are printed directly onto the surface of the bottles, are composed of discrete black dots which form alphanumeric characters arranged in two lines. The bottles upon which they are printed can be white or coloured depending on the product type. Insomuch as being concerned with alphanumeric characters, batchcode detection can be regarded as a similar problem to scene text detection. On the other hand, batchcode detection is generally a single target detection task which is similar to barcode detection, while scene text detection requires finding all regions that contain text.

Prior to the implementation of deep Convolutional Neural Networks (CNN), Maximally Stable Extremal Regions (MSER) [1] based methods achieved leading performance on benchmarking datasets due to their robustness against low resolution/quality and background noise [2]. However, they failed to address the diversity of printing methods and typefaces and the problem of false detections in high-frequency background regions [3]. Initially CNNs were used as classifiers to prune the non-text proposals [3], however more recent region-proposal-based CNN approaches [4]–[6], designed for object detection, achieved the state-of-the-art performance on large scale public datasets designed for text scene detection [7]–[9]. We observe similar trends in barcode and QR Code detection for the implementation of MSER based methods [10] and region proposal models [11]. MSER is designed to return each connected component that remains stable over a range of changing threshold values [1], which is suitable for detecting barcodes or text since the proposed regions follow special patterns and can be easily clustered and recognised. However, when applied to the task of batchcode detection such approaches return each discrete dot along with regions of strong background noise, demonstrating that low-level feature detectors may be insufficient for robust batchcode detection. Inspired by these previous works we propose a novel algorithm that accurately detects a single batchcode region in a given image by optimising the Maximally Stable Global Region (MSGR). Due to the relatively small body of comparable prior work, we collected our own dataset for evaluation of our proposed framework. Thus, the primary contributions of our work are:

- We evaluate an integrated pipeline that covers both data annotation and model training, for the automated detection of batchcodes in images of products.
- We investigate issues that arise when crowdsourcing data annotations, and propose solutions to address them with a semi-supervised solution.
- We build upon existing MSER methods to propose our own approach for batchcode detection, and demonstrate state-of-the-art performance on our dataset.
- We evaluate several state-of-the-art deep CNN models and suggest their utilisation under different circumstances.

In the following sections we first discuss relevant prior work, then detail the dataset used for training and evaluation, and finally present the experiment analysis and conclusions.

## II. Related Work

As discussed in Section I, there are similarities and discrepancies between scene text detection, barcode detection and batchcode detection, so in this section we discuss prior work in these areas.

### A. Scene Text Detection and Barcode Detection

Work approaching text detection and recognition prior to 2015 can broadly be either sliding-window (i.e. moving a multi-scale window through the input image and using a pre-trained classifier to decide whether the current window contains text) or connected component methods (whereby text is separated from background using low-level image features) [12]. It is reported that the latter achieves better performance while requiring less time and computational power on benchmarking datasets, especially the MSER based methods which take advantage of the high contrast between text and background [2], [3], [12]. However, due to their low-level nature MSER tend to propose a large number of non-text regions, thus pruning of proposals gains additional focus. [3] is the first paper to introduce a CNN-based approach to the removal of non-text region proposals and achieves promising results, however the benchmarking datasets used are of a relatively small size ($\approx$500 images in total). Later works [7]–[9], [13] updated benchmarking scores on larger scale datasets (over 1000 images) using CNN based detection models such as Faster-RCNN [4] and SSD (Single Shot MultiBox Detector) [5], addressing the power of deep CNNs on large scale computer vision tasks. Meanwhile, due to the potential for scene text to be of arbitrary orientation and shape, which cannot be accurately described by horizontally aligned rectangular bounding boxes, [14] proposed to horizontally aligned rectangular semantic segmentation methods for arbitrarily shaped text detection, obtaining the highest recall and F-measure scores on benchmarking datasets.

As discussed in Section I, barcode and batchcode both encode product information, and while batchcode detection remains a relatively unexplored problem, automatic barcode detection has been widely studied since 2009, when Tekin and Coughlan proposed an algorithm for blind users to find barcode from video footage with audio guidance [15]. Since barcodes consist of parallel black bars, the high intensity difference between these bars causes a high response within the gradient image generated by edge detectors such as sobel kernels, which can be further isolated from background noise using morphological operations and thresholding methods [16]. In [10] Creusot and Munawar proposed to detect the barcode region by measuring the stability of the output from a thresholding function while the threshold value is shifted within a predetermined range then filtering candidate regions by their aspect ratio. More recently, CNN-based methods have outperformed these approaches: the framework proposed by [11], which combines two separate CNN models, one for object detection and another for angle calibration, reported leading performance on barcode and QR code detection. However, it is observed that the bounding-box output by the model does not fully contain the barcode region, and so additional processing steps are introduced to refine the accuracy of the proposed region. Meanwhile, [17] divides an input image into small patches and uses a fully connected network to make a binary decision as to whether a code exists within a given patch. As such, the neural network is not used in an end-to-end manner, and the framework is very task-specific due to the hand-crafted feature extraction process.

There are other relevant works, e.g. [18] presented a gabor filter based expiry date recognition system, where they simply thresholded the grayscale images containing the cropped expiry date, without specifying how to automatically detect it. [19] proposed to use bottom hat filter to extract dot matrix printed characters, we compared the proposed MSGR with their method in the experiment section and found that their method is more sensitive to background noise than ours.

### B. CNN models for Semantic Segmentation

Deep CNN models have demonstrated state-of-the-art performance over traditional methods at a wide range of computer vision tasks, albeit with a requirement for large quantities of data and corresponding labels for training. The use of CNN which automatically update model parameters through back-propagation of errors, removes the need to design and optimise. By utilising massively parallel accelerators, such as

Graphics Processing Units (GPU), CNN-based approaches can achieve real-time processing even for tasks that deal with large size images.

The approach of Fully Convolutional Network (FCN) [20] creates a downsized class map that is upsampled by reverse pooling operations to create a final output comprising a vector of class label confidence values for every pixel in the original input image. SegNet [21] and U-Net [22] propose similar approaches, with the additional step of reusing output from prior pooling operations to better inform corresponding upsampling operations and retain higher resolution spatial information that would otherwise be lost. These approaches demonstrate state-of-the-art segmentation and classification results in a wide variety of scenarios, including medical imagery. U-Net was initially motivated by segmentation of medical imagery, and is a more light-weight architecture than Segnet or FCN, comprising eighteen convolutional layers and four pooling and upsampling operations. The output of each upsampling operation is concatenated with the input of the corresponding pooling operation in order to retain spatial information that would otherwise be lost through downsampling. PSPNet [23] proposes to increase the empirical receptive field and improve scene parsing by replacing global pooling with a pyraymid pooling module. DeepLabV3 [24] adds parallel atrous convolution to control the resolution of CNN feature maps and achieves the highest mean Intersection over Union (IoU) score on public datasets for semantic segmentation.

Considering the advances of CNN in object detection and segmentation, we evaluate four state-of-the-art architectures that have been proposed in various pixel-wise image segmentation tasks, in order to train an end-to-end batchcode detection model. The four models we evaluate are: FCN [20], DeepLabV3 [24], U-Net [22] and PSPNet [23]. We chose not to include anchor-based methods (e.g. Faster R-CNN [4], Mask R-CNN [6]) in this work as they are not capable of marking objects with oblique boxes, and since we focus upon a single-target detection task, it is not necessary to utilise frameworks that integrated classification function. Since FCN, PSPNet and DeepLabV3 all reported their best performance when using a ResNet101 [25] as backbone, in this paper we use FCN_ResNet101, PSPNet_ResNet101 and DeepLabV3_ResNet101 to explore the upper bound of segmentation accuracy, but denote them as FCN, PSPNet and DeepLabV3 correspondingly for simplicity.

## III. Method

In this section we introduce the proposed framework for refining batchcode bounding box annotations that are coarsely annotated by crowd-source workers, and use them to train segmentation models to predict batchcode locations on unseen samples. The trained models can be used to automatically annotate batchcodes with a bounding box from a given image. There are two steps to our proposed approach:

- MSGR: locate batchcode by tweaking threshold value until the foreground region remains stable.
- Deep Segment: training deep segmentation CNNs for automatic batchcode detection with image-label pairs.

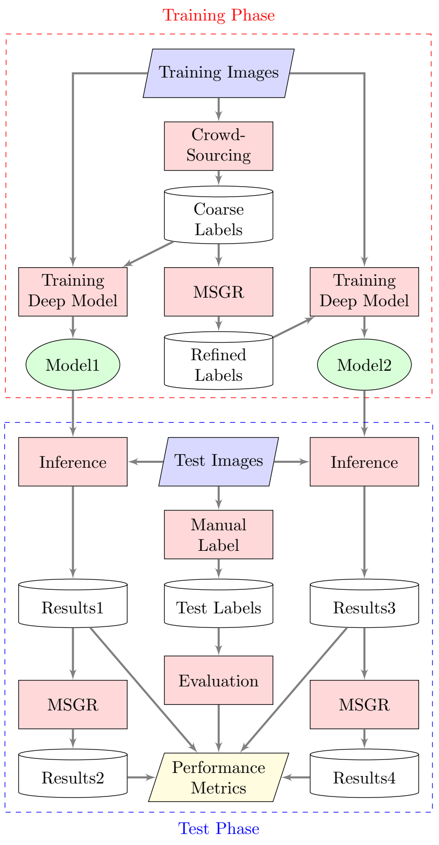

Fig. 2. The pipeline of the proposed framework. The Deep Segment models (Model1 and Model2) are trained with image-label pairs, and we use the proposed MSGR method to improve the accuracy of labelling the batchcode region. During test phase, we manually label the test data precisely to the boundary of the batchcode region, and evaluate the performance of the model outputs (Result1 and Result2), as well as the refined outputs after applying MSGR (Result3 and Result4).

Fig. 2 shows the full work flow of the framework, from raw image as input to model inference as output.

In this paper batchcode detection is regarded as a single target segmentation problem with large scale and shape variances. For the sake of simplicity in data labelling and storage, we use a bounding box represented by top left corner $\{x_1, y_1\}$ and bottom right corner $\{x_2, y_2\}$ to mark the rectangular[1] batchcode area with a bounding box. However, since most of the batchcodes are oblique to the horizontal or vertical axis, the bounding box is not an optimally compact fit to true batchcode region. In order to improve labelling accuracy, we also include rotation angle $\{\alpha | -90^\circ \leq \alpha \leq 90^\circ\}$ in the bounding box vector $\mathbf{b}$, i.e. $\mathbf{b} = \{x_1, y_1, x_2, y_2, \alpha\}$. To compute rotation angle $\alpha$, we exhaustively rotate the cropped thresholded batchcode and optimise for variance of columnwise sum, similar to the approach of [26].

[1]Strictly the batchcode region is trapezoid, using bounding box is a way to simplify the process. Experimentally it is also proved to be efficient.

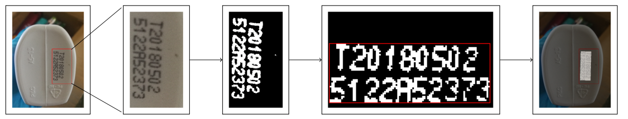

Fig. 3. The workflow of generating compact box from the coarse label using MSGR.

### A. Maximally Stable Global Region (MSGR)

**Algorithm 1** MSGR

```
1: procedure COLOUR SPACE TRANSFORM
2:     $I_{HED}(x,y,c) \leftarrow f_{HED}(I(x,y,c))$
3:     $t \leftarrow f_{Otsu}(I_{HED}(x,y,1))$
4: procedure MAXIMISE GLOBAL STABILITY
5:     $a_0 \leftarrow \infty$
6:     for $\lambda \in \{1, 1-\delta, 1-2\delta, ..., 0.5\}$ do
7:         $I_B(x,y) \leftarrow I_{HED}(x,y,1) > \lambda t$
8:         $a \leftarrow f_{dim}(I_B(x,y))$
9:         if $a - a_0 < 2$ then
10:            break
11:        else
12:            $a_0 \leftarrow a$
       return $I_B(x,y)$
```

Algorithm 1 illustrates how we combine a colour space transform and MSGR to find the batchcode region. $I(x,y,c), c \in \{1,2,3\}$ denotes the coarsely cropped image (in RGB colour space) as shown in Fig.3 $I(x,y,c)$. $I_{HED}(x,y,c)$ is the image in Haematoxylin-Eosin-DAB (HED) colour space [27], $f_{HED}$ denotes the colour space transform function from RGB to HED. Otsu's method $f_{Otsu}$ [28] is then applied to the first channel $I_{HED}(x,y,1)$ to obtain threshold $t$, that every pixel value above $t$ is set to 1 and the rest to 0 to obtain binary image $I_B(x,y)$ as shown in Fig.3 $I_B(x,y), \lambda=1$. We adjust a parameter $\{\lambda \in \mathbb{R} | 0 < \lambda \leq 1\}$ in descending order by $\delta$ for each iteration, where $\lambda$ represents the ratio applied to threshold to make the value gradually smaller(in our experiments we set parameter $\delta$ to $0.01$ for the balance of performance and speed), and multiply with $t$, until the sum of the bright area $a$ of the binary image remains stable, as shown in Fig.3 $I_B(x,y)$, $\lambda$ optimised). $f_{dim}$ denotes the method for calculating the dimensions of the foreground region of $I_B(x,y)$. Then we can wrap $I_B(x,y)$ with bounding box plus the rotation to obtain bounding box vector **b**.

Fig.3 demonstrates how images are processed through MSGR. Firstly, we obtain a coarse crop of the batchcode region, followed by thresholding, computing the angle of the batchcode and rotating the image so that it is aligned horizontally, retrieving a compact bounding box and generating a binary mask image where pixels inside the bounding box is 1, else 0, and finally rotating the binary mask to be aligned with the input image.

Generally, the batchcodes in our dataset are black ink marks printed on white or coloured bottles, and as such they should always have lower intensity values than those of the background. However, in reality there exists a significant amount of background noise that to nave thresholding methods would be indistinguishable from batchcode ink dots. In Fig.3 $I(x,y,c)$, there is a small dark region to the right of the batchcode batchcode, and as seen in Fig.3 $I_B(x,y), \lambda=1$ it is regarded as foreground after thresholding. In order to suppress noise of this type during batchcode extraction we adapt the MSER approach of [1] to find a foreground region that remains consistent while adjusting the threshold value given by Otsu's method in HED colour space. Instead of finding all the local regions, we consider the batchcode area as an integration and measure its area by maximising the stability across a range of thresholds in order to minimise the interference of noise, thus we name it Maximally Stable Global Regions. We adopt Otsu's thresholding method since it has excellent performance when foreground and background pixels are roughly equivalent [29], while in HED colour space the stability of the main colour region is not significantly influenced by additional colour regions, thus it is easier to remove dark regions that have slightly lighter colour than the batchcode inks. Empirically we also observed that HED outperforms other combinations in revealing batchcodes, such as HSV (Hue, Saturation, Value) [30] and CIELAB colour space [31]. The optimised bounding box is highlighted with transparent mask in Fig.3 (Refined label), with the original coarse bounding box shown in red for comparison.

Despite the above-mentioned improvements, there are two challenging scenarios that can cause the proposed algorithm to fail: 1) Unrelated text embossed on the bottle overlaps with the batchcode; 2) Additional unwanted ink near to the batchcode, for example pen, or accidental drops of ink from the printer. Fig. 4 demonstrates three cases: If the initial coarse bounding box includes any of these features, there is a chance that they will be incorrectly included in the batchcode bounding box. We remove these cases manually, and keep the good examples for training the segmentation neural network.

### B. Deep Segment

After obtaining the bounding box coordinates and rotation angles we produce binary masks for each image. We use these ground-truth masks, along with corresponding colour images, to train our selected deep segmentation models. We continue training until a converged validation loss curve is observed, and save the model weights for evaluation. A sigmoid function ensures network output is between 0 and 1, to which a binary threshold of 0.5 is applied followed by our MSGR algorithm to ensure that the final output region is optimally compact and accurate. We are expecting that the network outputs (the selected good samples) can be used for incrementally training the Deep Segment model. The hierarchical structure of CNN should be able to learn both low-level local features (pixel level features such as edge or texture details) and high-level global features (i.e. the batchcode region in our case), thus we assume that it is able to learn to predict a tight bounding box around

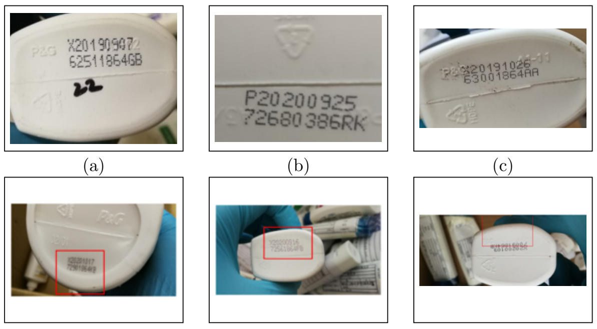

Fig. 4. The top row shows challenging samples where MSGR failed to generate a precise bounding box due to strong background noise. (a) handwritten text is visible on the bottle. (b) embossed regions of the bottle generate shadows whose pixel intensity is similar to that of the batchcode. (c) the batchcode is overlapped with the embossed bottle information, and there are many small black dots around the code. The bottom row are examples of coarsly labelled images marked by crowdsource workers. They are either off the target (partially cropped) or over cropped to the background region around batchcode.

the batchcode region from the refined labels, which helps eliminate the aforementioned noisy areas near the batchcode.

## IV. Experimental Evaluation

### A. Data

We evaluate our approach using our own dataset comprising 1479 images of shampoo bottles captured by mobile phone camera, which we split into training and test sets. The training set is initially annotated via the Amazon Mechanical Turk crowdsourcing platform (MTurk). Workers are instructed to draw an image-aligned rectangle containing the batchcode, however these boxes were found to rarely be accurate, as illustrated by the examples in Fig.4. Application of our MSGR technique, described in section 3.1, within these rectangles was used to obtain a more accurate bounding box for 1287 of the 1630 images originally sent to MTurk workers. During training, we randomly divide these images into training and validation sets using a ratio of 4:1. The test set consists of 192 images taken at a different time and location to the training set, which we manually annotate using batchcode-aligned bounding boxes.

### B. Experiments

As discussed in II, we compare four CNN architectures designed for semantic segmentation: FCN, DeepLabV3, U-Net and PSPNet. Models are trained with our training dataset using binary cross-entropy loss and Adam optimisation [32]. Input images are scaled to 512 x 512 pixels with padding used to compensate for differing aspect ratios, and random rotation is applied for data augmentation.

Mean loss across the validation set is used to judge model convergence, which we observe to happen at between 80 and 100 epochs for each model, at which point training ends. We report the performance of these four trained models on our test set of 192 manually annotated samples in I, where p, r and j refer to the mean precision, recall and Jaccard Index and fps refers to the number of images the network can process in one second.

First, we compare the impact of training label accuracy on model performance. Models trained with coarse labels sourced through MTurk generate Result1, while models trained with labels refined through MSGR generate Result2. When trained with coarse labels, U-Net, FCN and DeepLabV3 demonstrate very low precision and Jaccard Index, but high recall due to overly large segmented regions that include the batchcode as well as surrounding pixels. It is observed that by applying MSGR to the model output, precision and IoU are boosted by a large margin (35.33%(U-Net)-43.45%(FCN)), while recall is slightly reduced (2.44%(DeepLabV3)-4.59%(FCN)).

PSPNet struggles to make correct predictions across datasets, where most of the failed cases (IoU<0.8) are either missing part of the batchcode or segmenting regions of noise, thus it has both low precision and recall. MSGR only provides a moderate improvement, since it is not able to suppress hard noise, e.g. the high contrast edge of the plastic bottle, or other textures from the background, if they are included in the region segmented by the model.

Result3 and Result4 are generated by models trained using annotations refined through MSGR. DeepLabV3 achieves the best Jaccard Index, with FCN performing similarly. However, both are slow due to their use of a ResNet101 backbone, which is a very deep structure with many parameters compared with the more lightweight U-Net. PSPNet under-performs the same way as in Result1, indicating that its spatial pyramid pooling module may not perform well at tasks reliant on fine-grained textural features, such as small discrete dots. It is also observed again that the proposed MSGR method can further improve Jaccard Index by up to 2.25%(U-Net), albeit with a reduction in speed.

| **Model** | ***p(%)*** | ***r(%)*** | ***j(%)*** | fps |
|---|---|---|---|---|
| Result1 (Trained with Coarse Labels) | | | | |
| FCN | 41.26 | 98.50 | 40.71 | ≈12 |
| DeepLabV3 | 38.96 | 99.87 | 38.88 | ≈9 |
| U-Net | 38.77 | 99.44 | 38.23 | ≈23 |
| PSPNet | 66.69 | 61.33 | 42.51 | ≈10 |
| Result2 (Coarse Labels + MSGR) | | | | |
| FCN | 86.90 | 93.91 | 84.16 | ≈3.5 |
| DeepLabV3 | 83.87 | 97.43 | 82.11 | ≈3 |
| U-Net | 75.48 | 95.93 | 73.56 | ≈4.5 |
| PSPNet | 81.36 | 58.56 | 56.24 | ≈3 |
| Result3 (Trained with Refined Labels) | | | | |
| FCN | 93.29 | 97.08 | 90.53 | ≈12 |
| DeepLabV3 | 92.22 | 98.49 | **91.04** | ≈9 |
| U-Net | 92.54 | 94.86 | 88.25 | ≈**23** |
| PSPNet | 91.06 | 77.84 | 72.94 | ≈10 |
| Result4 (Refined Labels + MSGR) | | | | |
| FCN | 95.41 | 95.63 | **91.75** | ≈3.5 |
| DeepLabV3 | 93.83 | 95.56 | 91.63 | ≈3 |
| U-Net | 93.73 | 94.43 | 90.51 | ≈4.5 |
| PSPNet | 92.79 | 82.39 | 78.50 | ≈3 |

TABLE I
Performance comparison of four deep segment models trained with different data labels on the Testing dataset. The workflow of generating Result1-4 is illustrated in Fig.2.

We evaluate MSGR against the method proposed by [19] in its ability to refine CNN batchcode region segmentation output with bottom hat algorithm to fill holes, denoted as Bottom_hat in Table II. As mentioned in Section III MSGR is based on a colour space transform between RGB and HED, thresholded using Otsu's method. We also compare the results of using colour space transform alone, named Baseline in Table II, and the results show that when applying to FCN model trained with coarse labels (corresponding to FCN of Result2 in Table I, denoted as FCN_Result2), MSGR outperforms others by a significant margin.

| **Model** | *p(%)* | *r(%)* | *j(%)* | fps |
|---|---|---|---|---|
| Bottom_hat [19] | 77.02 | 97.86 | 75.72 | ≈3.5 |
| Baseline | 79.67 | 97.63 | 78.13 | ≈3.5 |
| FCN_Result2 | 86.90 | 93.91 | **84.16** | ≈3.5 |

TABLE II
PERFORMANCE OF MSGR COMPARED WITH OTHER THRESHOLDING METHODS.

## V. CONCLUSION

In this paper we have presented a new pipeline for the automatic detection of batchcodes in images of shampoo bottles. Beginning with a set of imperfect crowdsourced annotations, we demonstrated a novel approach to improving label quality by binarising the cropped batchcode region, detecting the orientation of the contained batchcode, then computing an optimally compact bounding box. In order to suppress noise when binarising the cropped image, we proposed a novel method named MSGR, which finds the maximum stability global region by tweaking the threshold value. Four state-of-the-art semantic segmentation CNN models are trained with the refined dataset and evaluated using a manually labelled dataset captured under different conditions. Experimental results indicate that our pipeline achieves good performance when detecting batchcode regions within real-world mobile images of products. This work demonstrates that for supervised machine learning tasks, accurate data labelling is vital for good model performance, and by automatically improving label quality we manage to achieve overall superior results.

## ACKNOWLEDGEMENT

This work was supported by the Procter & Gamble Company.

## REFERENCES

[1] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.

[2] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao, "Robust text detection in natural scene images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 970–983, 2013.

[3] Tong He, Weilin Huang, Yu Qiao, and Jian Yao, "Text-attentional convolutional neural network for scene text detection," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2529–2541, 2016.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*, 2016, pp. 21–37.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[7] Wenhao He, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu, "Multi-oriented and multi-lingual scene text detection with direct regression," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5406–5419, 2018.

[8] Jianqi Ma, Weiyuan Shao, Hao Ye, Li Wang, Hong Wang, Yingbin Zheng, and Xiangyang Xue, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018.

[9] Minghui Liao, Baoguang Shi, and Xiang Bai, "Textboxes++: A single-shot oriented scene text detector," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 3676–3690, 2018.

[10] C. Creusot and A. Munawar, "Real-time barcode detection in the wild," in *IEEE Winter Conference on Applications of Computer Vision*, 2015, pp. 239–245.

[11] Daniel Kold Hansen, Kamal Nasrollahi, Christoffer B Rasmussen, and Thomas B Moeslund, "Real-time barcode detection and classification using deep learning.," in *International Joint Conference on Computational Intelligence*, 2017, pp. 321–327.

[12] Qixiang Ye and David Doermann, "Text detection and recognition in imagery: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1480–1500, 2014.

[13] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu, "Textboxes: A fast text detector with a single deep neural network," in *AAAI Conference on Artificial Intelligence*, 2017.

[14] Cong Yao, Xiang Bai, Nong Sang, Xinyu Zhou, Shuchang Zhou, and Zhimin Cao, "Scene text detection via holistic, multi-channel prediction," *arXiv preprint arXiv:1606.09002*, 2016.

[15] Ender Tekin and James M. Coughlan, "An algorithm enabling blind users to find and read barcodes," *IEEE Workshop on Applications of Computer Vision*, pp. 1–8, 2009.

[16] Melinda Katona and László G Nyúl, "Efficient 1d and 2d barcode detection using mathematical morphology," in *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*, 2013, pp. 464–475.

[17] Péter Bodnár, Tamás Grósz, László Tóth, and László G. Nyúl, "Efficient visual code localization with neural networks," *Pattern Analysis and Applications*, vol. 21, no. 1, pp. 249–260, 2018.

[18] Ahmed Zaafouri, Mohamad Sayadi, and Farhat Fnaiech, "A vision approach for expiry date recognition using stretched gabor features," *International Arab Journal of Information Technology*, vol. 12, pp. 448–455, 2015.

[19] S. N. Patki, M. Joshi, and A. N. Kulkarni, "Dot matrix text recognition for industrial carton classification," in *International Conference on Industrial Instrumentation and Control*, 2015, pp. 777–782.

[20] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[21] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[23] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia, "Pyramid scene parsing network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2881–2890.

[24] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[26] Changming Sun and Deyi Si, "Skew and slant correction for document images using gradient direction," in *International Conference on Document Analysis and Recognition*, 1997, vol. 1, pp. 142–146 vol.1.

[27] Arnout C Ruifrok, Dennis A Johnston, et al., "Quantification of histochemical staining by color deconvolution," *Analytical and Quantitative Cytology and Histology*, vol. 23, no. 4, pp. 291–299, 2001.

[28] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[29] Mehmet Sezgin and Bülent Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–166, 2004.

[30] Dibya Jyoti Bora, Anil Kumar Gupta, and Fayaz Ahmad Khan, "Comparing the performance of L*A*B* and HSV color spaces with respect to color image segmentation," *arXiv preprint arXiv:1506.01472*, 2015.

[31] Vijai Singh and A.K. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques," *Information Processing in Agriculture*, vol. 4, no. 1, pp. 41 – 49, 2017.

[32] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2014.