Optical Braille Recognition Using Object Detection Neural Network

Ilya G. Ovodov
ELVEES RnD center, JSC
Zelenograd, Russia
iovodov@elvees.com

Abstract

Optical Braille recognition methods generally rely heavily on a Braille text's geometric structure. They run into problems if this structure is distorted. Thus, they find it difficult to cope with images of book pages taken with a smartphone.

We propose an optical Braille recognition method that uses an object detection convolutional neural network to detect whole Braille characters at once. The proposed algorithm is robust to deformations and perspective distortions of a Braille page displayed on an image. The algorithm is suitable for recognizing braille texts captured with a smartphone camera in domestic conditions. It can handle curved pages and images with perspective distortion. The proposed algorithm shows high performance and accuracy compared to existing methods.

Additionally, we produced a new dataset containing 240 photos of Braille texts with annotation for each Braille letter. Both the proposed algorithm and the dataset are available at GitHub.

1. Introduction

The embossed Braille alphabet was invented in 1824 and served as the primary way of writing and reading for blind people for many years. Recent advances in technology provide blind people with many new opportunities for receiving and transmitting information. Still, reading printed Braille and writing in Braille continues to be an important communication method for them. Moreover, Braille is often used for communication between blind and sighted people. In particular, the situation is quite common when sighted teachers work with blind students, and they have to deal with textbooks and student works written in Braille.

Braille is an alphabet made of dots embossed on paper for tactile recognition. Still, sighted people who deal with Braille texts usually do not have enough tactile skills to read the text with their fingers and have to read it with their eyes. For sighted people, Braille text looks like a lot of tiny white bulging points on a white background, so its visual recognition is very tedious. Reading double-side printed Braille is especially difficult. Text printed on a back of a page looks like dented dots. Tactilely these dots are almost invisible and do not interfere with the sensation of front side convex points. Still, visually they are hardly distinguishable from them (see Figure 1). So, reading such a text with the eyes is especially difficult. The use of technical Braille recognition tools, particularly optical recognition methods, can greatly facilitate this work. Thus, optical Braille recognition (OBR) methods have been developing since at least the 1980s [2].

Figure 1. Example of a page Taken with double-sided printing.

Each letter or other character is represented in a Braille text with several (1 to 6) bulging points located in a 2x3 grid. So, 63 different characters can be encoded. The distance between adjacent symbols is slightly larger than between two columns of points in one symbol. The character width, the spacing between characters, the spacing between lines, and the characters' places on a line are constant for each Braille document. Thus, the Braille characters' dots are located at nodes of a fixed grid (Figure 2). The use of this geometric structure is critical to most existing methods of Braille recognition. However, this significantly limits their applicability. These algorithms require either a scanner or special photographing conditions to ensure accurate alignment of a Braille page.

This work aims to provide the recognition method applicable to Braille text images obtained on a mobile phone camera in the domestic environment. The grid Braille char-

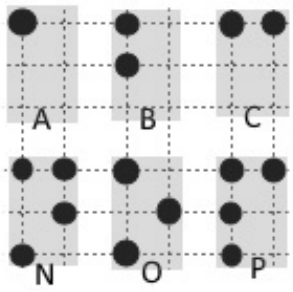


Figure 2. Schematic arrangement of Braille text.

acters are attached to can be significantly distorted due to: a) perspective distortions caused by the fact that the sheet is not perpendicular to the camera optical axis, b) the paper sheet curvature on open book spread images. Also, different areas of the sheet can have significantly different lighting (Figure 3).

To cope with these problems, we proposed an optical Braille recognition algorithm based on object detection convolutional neural networks. The proposed algorithm is robust to distortions of a Braille page image described above. So, the algorithm is suitable for recognizing braille texts captured with a smartphone camera in everyday conditions. The proposed algorithm has shown high performance and accuracy compared to existing methods.

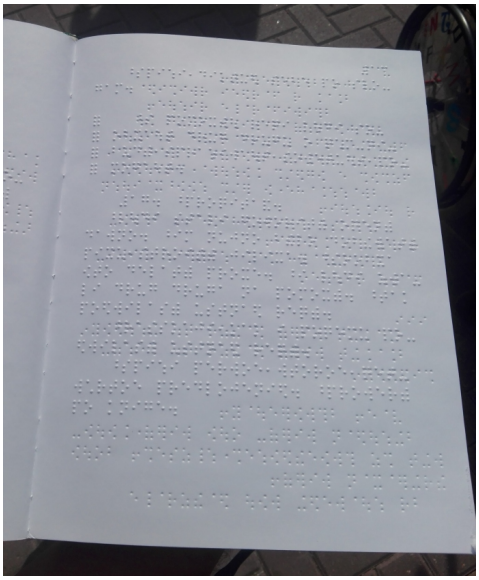
Additionally, we faced the lack of publicly available datasets that can be used to train a deep neural network and evaluate recognition algorithms' accuracy. The only available dataset is limited and does not cover the cases described above. We produced a new dataset containing 240 photos of Braille texts with annotation for each Braille letter.

2. Related work

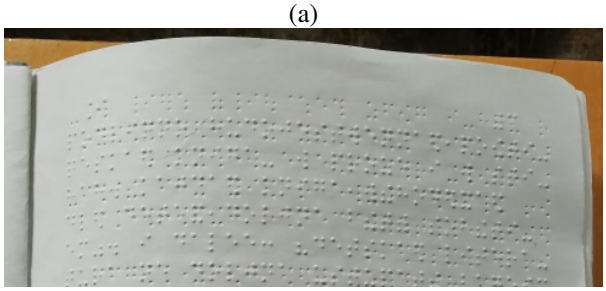
2.1. Optical braille recognition approaches

The main approach to optical braille recognition consists of sequentially performing Braille points search, characters grid restoration, including compensation for possible image rotation, points grouping into characters, and, finally, character decoding. The survey papers [21, 5] consider all algorithms following this sequence of steps.

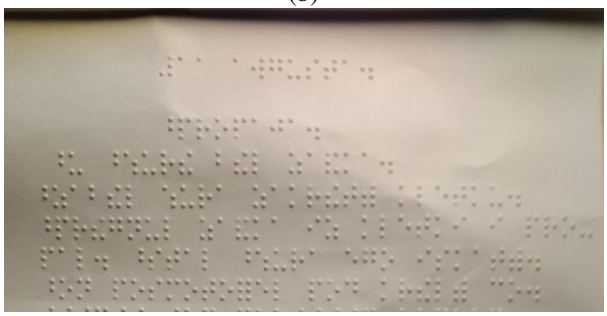
The simplest approach to point detection is thresholding. Zhang and Yoshino [24] use a dynamic local threshold. To detect dots on double-sided braille and distinguish between the front side and reverse side dots, Antonacopoulos and Bridson [1] use detection of light and dark areas of dots and segment image into areas of three classes: bright, dark, and background. They use static thresholds relative to the average brightness level in the vicinity of the point. Morgavi and Morando [16] use a simple neural network to



(a)



(b)



(c)

Figure 3. Examples of Braille text smartphone photos with perspective distortion (a), page curvature (b), and deformation (c).

find points. Venugopal-Wairagade [22] fulfills circle detection using Hough transform. Perera, Wanniarachchi [17] use HOG and SVM, R.Li *et al*. [10] - Haar detector and Adaboost., R.Li *et al*. [11] - Haar detector and Adaboost for primary dot detection and HOG, LBP, and SVM for final dot detection after grid restoration.

The next step is to restore the grid to which the points are anchored and possibly compensate for the sheet rotation. For this purpose can be used linear regression [17], Hough Transform [1, 6], or coordinates distribution density dur-

ing step-by-step rotation of the image can be investigated to find optimum rotation angle [11]. Sometimes, after the image de-skewing, points are searched again using information about their possible position [1, 11].

Although some methods assume the grid deformation, implying a change in the pitch between different lines ([11] and others), it is assumed that the grid lines are straight on the entire sheet and parallel.

Since the mentioned works essentially rely on snapping Braille points to the grid, they mostly work with images obtained with a scanner. In this case, the necessary image correction is reduced to de-skewing, which makes the grid lines vertical and horizontal. Only a few works declare the purpose of OBR on images from a smartphone ([24, 22]), but the methods described by them are still based on the presence of a rectangular grid on a sheet.

Although convolutional neural networks (CNN) made tremendous advances in image recognition in recent years, the use of deep learning and neural networks for OBR is sparse. There are only a few papers on the use of fully connected neural networks. Morgavi, Morando [16], and Ting Li *et al*. [13] use a simple neural network to find points, Subur *et al*. [20] - to find the symbol value using the points found by image segmentation. Kawabe *et al*. [6] use it for separating front and back points when recognizing two-sided braille. Only R.Li *et al*. [12] presented at CVPRW 2020 work based on segmentation neural network with modified UNet architecture. They used a neural network to determine areas occupied by Braille characters and recognize these characters. To determine the positions of individual characters based on segmentation results, subsequent post-processing is required.

2.2. Optical braille recognition methods accuracy evaluation

While various works provide different quantitative accuracy characteristics of proposed algorithms, sometimes quite high, comparing algorithms with each other encounters at least two obstacles:

- Until recently, there was no open dataset to compare different algorithms. Quality values provided in papers were measured on proprietary, not published datasets, so it is mostly impossible to compare published results of different works.

- Often works that use the general pipeline described above (i.e., points detection-grid restoration-grouping points into symbols and decoding), quality indicators are given only for the point recognition stage. It prevents comparing the performance of these algorithms with algorithms that do not have a separate point recognition stage, such as our algorithm.

The only publicly available DSBI dataset with braille text was published by Li *et al*. [10]. It contains 114 pages of scanned two-sided braille texts, divided into the train (26 pages) and test (88 pages) sets. All pages are carefully aligned during scanning. A grid of points for the front and back sides has been calculated, rotation required to bring the grid to a vertical-horizontal orientation has been calculated. The annotation is made by specifying the rotation angle, the coordinates of vertical and horizontal grid lines after rotation, and a list of braille characters referenced to this grid's nodes. All texts are in Chinese, but the Braille alphabet has the same structure in all languages, and this dataset can be used regardless of language.

Although this dataset does not look large enough and variable enough to provide full training of recognition algorithms (only 26 pages in the training set), it allows you to compare different approaches to the problem. The authors of [10] compared the accuracy of algorithms based on image segmentation (Antonacopoulos *et al*., [1]), Haar features and Adaboost (Viola & Jones [23]), and their algorithm (Li *et al*. [11]). However, they provide only the accuracy of point detection. Only in [12] they provided accuracy metrics of their algorithm, estimated not at the dot level, but at the character level.

2.3. Object detection convolutional neural nets

The use of convolutional neural networks (CNN) in computer vision and, in particular, in object detection has made tremendous strides in recent years. Convolutional networks were proposed by LeCun in 1989 [9], but their popularity has exploded since 2012 [8]. After classification problems, they were applied to solve the object detection problem: the simultaneous finding of rectangular areas containing objects in the image and classifying the objects contained in them. Initially, CNN-based solutions for object detection processed search for regions and classification of objects separately ([4] and others). Later methods that simultaneously search for regions and their classification (one-stage detectors) have achieved great success.

The one-stage detector principle for the object detection has become widespread with the advent of SSD ([15, 3]) and YOLO ([18, 19]) convolutional neural network architectures. One-stage detectors' key idea is that after a series of convolutions and size reductions, they produce a feature map. Each point of this map corresponds to a square region of the original image. Before training, several a priori sizes of desired objects are set for each cell of the feature map, called anchors. For each point of the feature map (i.e., for each square area in the original image), the ground truth annotation boxes intersecting with the anchors are considered, and the degree of intersection is calculated as *IOU* (intersection over union). The neural network learns to predict the necessary shift and resizing of the anchor (4 out-

puts per anchor), the degree of intersection (1 output per anchor), which is concerned as a confidence measure at the inference stage, indicating the presence of an object in the anchor area, and a class of the object (C output parameters where C is the number of classes). Thus, at each point of the feature map, the neural network learns to predict either $A \cdot (4 + 1 + C)$ output values, where A is the number of anchors, 4 outputs define coordinates of the bounding box ([18, 19]) or $A \cdot (4 + C)$ output values ([15, 3, 14]). In the latter case, confidence is included in the responses for all C classes so that for anchors whose areas do not correspond to objects, the responses for all C classes are small.

Simultaneous learning of location and class prediction is achieved using the loss function

$$L = L_{loc} + \lambda_{cls} \cdot L_{cls} \quad (1)$$

where L_{loc} is the loss function for location prediction errors, L_{cls} is the loss function for classification errors, λ_{cls} is the weighting factor.

The RetinaNet [14] further develops this approach proposing an improved loss function FocalLoss for L_{cls} component of the loss function, which provides more weight for more complex cases. See [14] for details. We used this implementation of object detection CNN in this work.

3. Our approach

3.1. The problem setting and the network architecture

Unlike conventional approaches, we do not separate stages of dots detection, grid restoration, and combining dots into characters. Instead, we find whole Braille symbols directly and simultaneously recognize them, using the object detection CNN described above. We assign each character a class label from 1 to 63 using formula $c = \sum_1^6 2^{i-1} p_i$ where p_i is 1 if the i-th point presents in the character and 0 otherwise.

We scale an input image to 100dpi resolution. Thus, the standard A4 page is scaled to approximately 864x1150 resolution. Braille characters are located with a horizontal spacing of about 25pt and line spacing about 40pt.

We use RetinaNet CNN architecture described in [14] with some minor modifications. The Optical Braille Recognition task differs in that it searches for a large number of approximately the same small size objects with a fixed width to height ratio. Therefore, we have simplified RetinaNet architecture to reduce the execution time, primarily NMS operations. Only one "output to class + box subnet" (see Figure 3 in [14]) was used at the layer level with feature map cells having 16x16 size. It guarantees that every Braille character is covered by at least one grid cell. We used only one anchor for each grid cell with a size close to the expected character size. These modifications have reduced calculation time by more than 5 times without substantial loss of recognition quality.

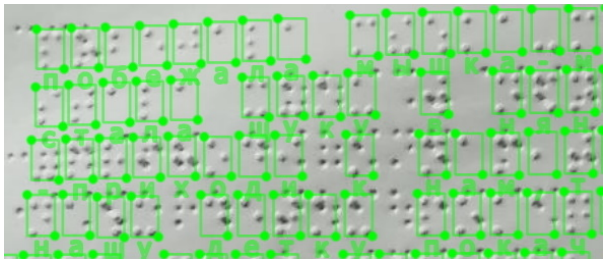


Figure 4. Sample annotations of Angelina Braille Images Dataset.

Also, we used a priory concern that Braille characters do not overlap and substantially reduced the IOU threshold used to filter overlapping detected bounding boxes using the NMS procedure. We used IOU threshold = 0.02 to allow only small overlapping due to detector imperfection.

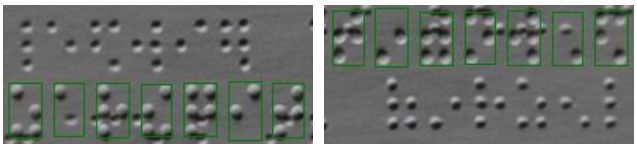
3.2. Angelina Braille Images Dataset

DSBI dataset [10] was the only publicly available dataset with labeled Braille text images, and we would like to express our deep appreciation to its creators. However, DSBI contains only scanned Braille images where Braille dots are aligned to a rectangular grid. It has a rather small number of images with limited diversity. Therefore, this dataset is not suitable for training the CNN that can properly recognize difficult images we aim to handle, neither for testing CNN designed to handle such difficult samples.

We prepared the new "Angelina Braille Images Dataset" containing 212 pages of double-sided braille books and 28 pages of student papers. These texts were photographed with various photo cameras or mobile phones under conditions close to the algorithms' intended work conditions. It includes curved pages in the open spread of the book and perspective distortions. Characters on the front side of the texts are labeled using the usual object detection problem method: for each character, a bounding box is defined, and a class from 1 to 63 is assigned, corresponding to the braille character inside the box. Sample images are shown in Figure 3, sample labels in Figure 4.

An annotation of the dataset was produced iteratively. At each iteration:

1. primary annotation was automatically generated;
2. automatically generated annotation bounding boxes were corrected manually to fit Braille text lines;
3. annotation Braille characters were converted into a plain text;



(a) (b)

Figure 5. Front side bulging dots (inside green rectangles) looks like reverse side bulged-in dots and vice versa if the image (a) is 180o rotated (b).

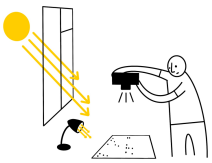


Figure 6. Light conditions Angelina Braille Images Dataset images were taken.

4. poetry texts were checked by comparison with ground truth texts of the same poems found from the Internet, considering that splitting the text into lines for poetry texts is fixed. At later iterations, non-poetry texts were checked by a spell-checker. All questionable cases were checked manually. As a result, when it is not clear if there is a Braille dot at some position or not, all ambiguous cases were resolved in favor of the grammatically correct option.

We noticed that bulging front side dots and bulged-in reverse side dots are distinguishable only when the falling light direction is known. Figure 5 shows that front side dots seems like reverse side dots and vice versa if the image is 180°rotated. We avoided this ambiguity by taking all images in light falling from the approximately top or top-left side of the page (Figure 6).

The dataset is divided into 191 training (80%) and 49 test (20%) images. Also, 44 images of various non-Braille texts from the Internet were added to the training set as negative examples.

Braille characters classes are labeled by corresponding Russian plain text letters and symbols. Still, these labels can be converted to braille dots using software tools provided with the dataset.

Our Angelina Braille Images Dataset is available at GitHub: https://github.com/IlyaOvodov/AngelinaDataset.

4. Experiments setup and results

4.1. Metrics used

We evaluated *precision*, *recall*, and *F1* metrics at a character and pixel level. Since the algorithm immediately detects symbols, it is natural to evaluate its precision at the symbol level. We used a calculation method that coincides with [12]. Detected Braille characters that intersect with ground truth characters with *IOU* (intersection over union)$\geq$ 0.5 and have a correct class are considered true positives (*TP*). Otherwise, they are considered as false positive (*FP*). So *FP* detections include both detections with either incorrect position or correct position but incorrect Braille characters assigned to them. Ground truth symbols that do not have corresponding TP detected Braille symbols are considered false negative (*FN*). The *precision*, *recall*, and *F1* metrics are defined as:

$$\begin{gathered} precision = TP/\left(TP + FP\right) \\ recall = TP/\left(TP + FN\right) \\ F1 = 2 \cdot precision \cdot recall/\left(precision + recall\right) \end{gathered} \tag{2}$$

We also evaluated per-dot metrics to compare our results with known Braille recognition methods based on dot detection. Since our algorithm does not detect individual points, we do it indirectly. All points of *TP* characters are considered as *TP* points. All points of ground truth characters that do hot have detection intersecting with them with $IOU \geq 0.5$ are considered *FN* points, and all points of detection characters that do not intersect with ground truth with $IOU \geq 0.5$ are *FP* points. For detections that intersect with ground truth characters with $IOU \geq 0.5$ but has character other than ground truth character, we compare the presence of point at each of 6 places of the g.t. and detected Braille characters. If some point present in both characters, it is considered as *TP*. Otherwise, it is considered as *FP* and *FN*, respectively. Then we evaluate Precision, Recall, and F1 at the dot level in a way described above (2).

4.2. Network training

We used the DSBI dataset [10] to compare the effectiveness of our algorithm with approaches published earlier [10, 11, 12]. Train-test split defined for DSBI dataset contains only 28 train images, which is too small for CNN training. We defined a train set as the first 74% and a test set as the last 26% pages of each Braille book in the DSBI dataset. It resulted in 84 and 30 images for train and test sets, respectively.

To evaluate our algorithm in more complex conditions, we combined the train sets from the DSBI dataset described above and our Angelina Braile Images Dataset. The evaluation was performed separately on the DSBI and Angelina Braile Images Dataset test sets.

We trained the neural network to handle images resized to 864-pixel width, which corresponds to approximately 100dpi. When training the neural network, to obtain a better resistance to different image scales and possible input distortions, train images were augmented as follows. Each

Method	Train dataset	Test dataset	Braille dot level			Character level			Perform., s/image
			Prec.	Recall	F1	Prec.	Recall	F1	
Segment [10]	DSBI [10]	DSBI	0.9172	0.9811	0.948				
Haar [10]			0.9765	0.9638	0.970				
Haar [11]			0.9838	0.9575	0.970				0.89
HOG,SVM [11]			0.9314	0.9869	0.958				15.02
SVM Grid [11]			0.9931	0.9997	0.996				1.22
TS-OBR [11]			0.9965	0.9997	0.998	0.9928	0.9996	0.9962	1.45
BraUNet [12]						0.9943	0.9988	0.9966	0.25
Our	DSBI	DSBI	0.9992	0.9995	**0.9994**	0.9977	0.9975	**0.9976**	**0.18**
	DSBI+Our		0.9984	0.9993	0.9989	0.9961	0.9964	0.9963	
	DSBI	Our	0.9812	0.9143	0.9466	0.9569	0.8980	0.9265	
	DSBI+Our	Our	0.9995	0.9986	**0.9991**	0.9985	0.9978	**0.9981**	

Table 1. Experimental results. Precesion, recall, F1 metric on a test dataset at dot and character levels, and processing performance.

image was scaled to a random width from 550 to 1150pix, which is $\pm 30\%$ of the required width. Then image was was compressed or stretched vertically by a random scale within $\pm 10\%$. Then, we rotated images at a random angle within $\pm 5°$. With a probability of 50%, we reflected the image along the vertical axis and changed each character label to the reflected character's label.

We normalized the image using the formula

$$x_c = \frac{I_c - m}{3 \max(s, 0.1 \cdot 255)} \tag{3}$$

where I_c is the intensity of the image channel in the range [0,255], c, m is the mean of I_c over the whole image, and s is the standard deviation of I_c over the image.

A random 416x416 image crop was used as CNN input. We trained the neural network for 500 epochs using Adam optimizer [7] with learning rate = 1e-4 and batch size = 24. Initially, the λ_{cls} factor in the loss function (1) was set to 1. In this case, L_{loc} component of the loss function prevails on L_{cls} resulting in more fast learning of character position than character classification. After 500 epochs, we set λ_{cls} to 100, making the contribution of both components approximately similar. We noticed that if the contribution of both components of the loss function is set equal from the very beginning, the learning process becomes unstable. Finally, we set $\lambda_{cls} = 1000$ and train the CNN using the "Reduce On Plateau" approach, i.e., reducing learning rate by factor 10 if the *F1* metric on the test set does not decrease for 500 epochs.

4.3. Results and discussion

Table 1 shows the results of the experiments.

When trained on the DSBI dataset, the new method gives *F1=0.9976* in a character-based test and *F1=0.9994* in a dot-based test. It outperforms other methods.

Investigation of characters that cause errors shows that the correct label is often questionable (Figure 7). The decision of whether detection or ground truth label is correct is, to a significant extent, subjective. Therefore a further comparison of algorithms with $F1 \geq 0.997$ on the DSBI dataset seems not informative. The reason is that the DSBI dataset was labeled without considering the semantic meanings of characters. So ambiguous cases were labeled arbitrary. The Angelina Dataset proposed in this paper was labeled using Braille characters' semantic meaning, so questionable cases were resolved in favor of an option that should present at the specific place from the grammatical point of view.

When trained on both DSBI and Angelina Braille Images Dataset, our method results in *F1=0.9981* on the Angelina test set and *F1=0.9963* on DSBI. It can be assumed that some accuracy decrease on this relatively homogeneous dataset is because CNN was trained on more diversified data, requiring more generalizing ability. As we can see, paper deformations and perspective distortion that present in photos in this dataset do not make recognition quality worse than on the DSBI dataset, where pages are flattened and scanned without distortion.

Processing of one A4 page with proposed method takes time 0.18 s/image on GPU NVIDIA 1080Ti. It outperforms other methods, including BraUNet [12], for which 0.25s/image is reported for the same hardware.

Source code and trained network weights of our algorithm are available at GitHub: https://github.com/IlyaOvodov/AngelinaReader.

Figure 7. Some characters in DSBI dataset being counted as errors. Red numbers – ground truth Braille character dots, blue – detected Braille character dots. It is not clear what one is correct.

We issued a public web service Angelina Braille Reader for recognition of Braille text images. It is available at https://angelina-reader.com.

Conclusion

In this paper, we proposed a new optical Braille recognition algorithm based on object detection convolutional neural network. The proposed algorithm has shown high accuracy and performance. This algorithm proved to be resistant to irregularities and perspective distortions of the depicted sheet with Braille text. Thus, it can recognize texts captured with a mobile phone in an everyday domestic environment.

This significantly distinguishes our algorithm from existing ones. Other methods need images with Braille text's geometric structure being undisturbed. Our algorithm's robustness makes it possible to use it as a basis for software service that can be used for the everyday needs of people who have to read Braille texts by their eyes. Such as teachers of blind students, parents of visually impaired children.

We faced that the only available dataset with annotated Braille texts does not contain enough difficult samples to evaluate Braille recognition algorithms' performance for the purposes described above. Both our algorithm and the best algorithms described before have almost 100% accuracy on this dataset. The error rate is comparable with the amount of ambiguous annotated characters.

We created and published a new Angelina Braille Images Dataset. It contains more difficult samples taken by smartphone and hand camera. Our dataset includes curved book spread pages, perspective distorted images, and other hard cases. Our algorithm has good performance on these more difficult images as well.

The proposed algorithm, the new Angelina Braille Images Dataset, and web service Angelina Reader for optical Braille recognition based on our algorithm are available on the Internet.

References

[1] Apostolos Antonacopoulos and David Bridson. A robust braille recognition system. In *International Workshop on Document Analysis Systems*, pages 533–545. Springer, 2004. 2, 3

[2] JP Dubus, M Benjelloun, V Devlaminck, F Wauquier, and P Altmayer. Image processing techniques to perform an autonomous system to translate relief braille into black-ink, called: Lectobraille. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1584–1585. IEEE, 1988. 1

[3] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 3, 4

[4] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 3

[5] Samer Isayed and Radwan Tahboub. A review of optical braille recognition. In *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*, pages 1–6. IEEE, 2015. 2

[6] Hiroyuki Kawabe, Yuko Shimomura, Hidetaka Nambo, and Shuichi Seto. Application of deep learning to classification of braille dot for restoration of old braille books. In *International Conference on Management Science and Engineering Management*, pages 913–926. Springer, 2018. 2, 3

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 3

[9] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 3

[10] Renqiang Li, Hong Liu, Xiangdong Wang, and Yueliang Qian. Dsbi: Double-sided braille image dataset and algorithm evaluation for braille dots detection. In *Proceedings of the 2018 the 2nd International Conference on Video and Image Processing*, pages 65–69, 2018. 2, 3, 4, 5, 6

[11] Renqiang Li, Hong Liu, Xiangdong Wang, and Yueliang Qian. Effective optical braille recognition based on two-stage learning for double-sided braille image. In *Pacific Rim International Conference on Artificial Intelligence*, pages 150–163. Springer, 2019. 2, 3, 5, 6

[12] Renqiang Li, Hong Liu, Xiangdong Wang, Jianxing Xu, and Yueliang Qian. Optical braille recognition based on semantic segmentation network with auxiliary learning strategy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 554–555, 2020. 3, 5, 6

[13] Ting Li, Xiaoqin Zeng, and Shoujing Xu. A deep learning method for braille recognition. In *Proceedings of the 2014 International Conference on Computational Intelligence and Communication Networks*, pages 1092–1095, 2014. 3

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 4

[15] 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*, pages 21–37. Springer, 2016. 3, 4

[16] GIOVANNA Morgavi and Mauro Morando. A neural network hybrid model for an optical braille recognitor. In *International Conference on Signal, Speech and Image Processing 2002 (ICOSSIP 2002)*. Citeseer, 2002. 2, 3

[17] TDSH Perera and WKIL Wanniarachchi. Optical braille recognition based on histogram of oriented gradient features and support-vector machine. *International Journal of Engineering Science*, 19192, 2018. 2

[18] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 3, 4

[19] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 3, 4

[20] Joko Subur, Tri Arief Sardjono, and Ronny Mardiyanto. Braille character recognition using find contour and artificial neural network. *JAVA Journal of Electrical and Electronics Engineering*, 14(1), 2016. 3

[21] V Udayashankara et al. A review on software algorithms for optical recognition of embossed braille characters. *International Journal of computer applications*, 81(3):25–35, 2013. 2

[22] Gayatri Venugopal-Wairagade. Braille recognition using a camera-enabled smartphone. *Int J Eng Manuf*, 4:32–39, 2016. 2, 3

[23] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001. 3

[24] Shanjun Zhang and Kazuyoshi Yoshino. A braille recognition system by the mobile phone with embedded camera. In *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, pages 223–223. IEEE, 2007. 2, 3